Configuration du pilote DirXML Workflow Request Service

Guide d'administration

Révision 0.1 du document

Exclusion de garantie générale
Cette documentation ne doit pas être interprétée comme une promesse de développement, livraison ou commercialisation d'un produit de la part d'une société participante. Novell exclut toute garantie relative au contenu ou à l'utilisation de cette documentation. En particulier, Novell ne garantit pas que cette documentation est exhaustive ni exempte d'erreurs. Novell se réserve en outre le droit de réviser cette documentation à tout moment et sans préavis.

Présentation

Cette section présente le pilote DirXML® Workflow Request Service et fournit notamment les informations nécessaires pour le configurer.

Le pilote Workflow Request Service est conçu pour signaler à un ou plusieurs utilisateurs l'occurrence d'un événement de données et, dans certains cas, la nécessité d'une intervention de la part de l'utilisateur. Dans un scénario de provisioning de personnel, l'événement de données peut être la création d'un nouvel objet Utilisateur, et l'intervention de l'utilisateur peut consister notamment à assigner un numéro de bureau grâce à la saisie de données dans Novell® eDirectory™ ou dans une application. D'autres scénarios incluent l'envoi d'une notification à l'administrateur pour lui signaler la création d'un objet Utilisateur, la modification des données d'un objet par un utilisateur, etc.

Configurer le pilote Workflow Request Service consiste généralement à configurer deux sous-systèmes distincts mais associés : les règles et modèles de message électronique du canal Abonné et les modèles de serveur Web du canal Éditeur (et les règles, éventuellement).

En outre, les paramètres de pilote comme le nom du serveur SMTP, le numéro de port du serveur Web, etc., doivent être définis.

Modes de fonctionnement

Deux principaux modes de fonctionnement sont pris en charge :

- **Demande directe de données** : Un message électronique est envoyé pour demander à l'utilisateur de saisir des données dans eDirectory (ces données pourront être éventuellement utilisées par une autre application). Le destinataire répond en cliquant sur l'URL fournie dans le message. L'URL pointe sur le serveur Web qui s'exécute sur le canal Éditeur du pilote Workflow Request Service. L'utilisateur utilise ensuite les pages Web dynamiques générées par le serveur Web pour s'authentifier auprès de eDirectory et saisir les données requises.
- **Notification d'événement** : Un message électronique est envoyé à l'utilisateur sans utiliser le canal Éditeur. Il peut simplement s'agir de la notification d'une opération effectuée dans eDirectory ou d'une requête de données par un moyen autre que le serveur Web du canal Éditeur, par exemple Novell iManager, une autre application ou une interface personnalisée.

Exemple : Message électronique du canal Abonné, réponse du serveur Web du canal Éditeur

Exemple de scénario de provisioning de personnel dans lequel le responsable d'un nouvel employé lui assigne un numéro de bureau :

1. Un objet Utilisateur est créé dans eDirectory (par exemple, par le pilote DirXML du système des ressources humaines de la société).
2. Le canal Abonné du pilote Workflow Request Service envoie un message SMTP au responsable de l'utilisateur et à l'assistant du responsable. Ce message contient une URL qui désigne le serveur Web du canal Éditeur. L'URL contient par ailleurs des éléments de données qui identifient l'utilisateur et les personnes autorisées à soumettre les données requises.

3. Le responsable ou son assistant clique sur l'URL contenue dans le message électronique. Un formulaire HTML s'affiche dans un navigateur Web. Le responsable ou son assistant procède ensuite comme suit :

- Il sélectionne le DN de son objet Utilisateur eDirectory afin de s'identifier en tant qu'auteur de la réponse au message.
- Il saisit son mot de passe eDirectory.
- Il saisit le numéro de bureau du nouvel employé.
- Il clique sur le bouton Submit (Soumettre).

4. Le numéro de bureau du nouvel employé est envoyé à eDirectory via le canal Éditeur du pilote Workflow Request Service.

Exemple : Message électronique du canal Abonné, aucune réponse du canal Éditeur

Exemple de scénario dans lequel le responsable d'un nouvel employé lui assigne un ordinateur via un système de gestion des ressources :

1. Un objet Utilisateur est créé dans eDirectory (par exemple, par le pilote DirXML du système des ressources humaines de la société).
2. Le canal Abonné du pilote Workflow Request Service envoie un message SMTP au responsable de l'utilisateur et à l'assistant du responsable. Ce message contient des instructions qui permettront d'entrer les données dans le système de gestion des ressources.
3. Le responsable ou son assistant saisit les données dans le système de gestion des ressources.
4. (Facultatif) Les données d'identification de l'ordinateur sont transmises à eDirectory via le pilote DirXML du système de gestion des ressources.

Création de messages électroniques et de pages Web par le pilote Workflow Request Service

Les messages électroniques, les pages Web HTML et les documents XDS peuvent tous être considérés comme des *documents*. Le pilote Workflow Request Service crée des documents de manière dynamique, d'après les informations fournies au pilote.

Les *modèles* sont des documents XML qui contiennent les *textes standard* ou parties fixes d'un document ainsi que des *jetons de remplacement* qui indiquent l'emplacement des parties dynamiques (de remplacement) du document final construit.

Le canal Abonné et le canal Éditeur du pilote Workflow Request Service utilisent tous deux des modèles pour créer des documents. Le canal Abonné crée des messages électroniques et le canal Éditeur, des pages Web et des documents XDS.

La partie dynamique d'un document est fournie via des *données de remplacement*. Les données de remplacement du canal Abonné sont fournies par ses règles (comme la règle de transformation de commande). Celles du canal Éditeur proviennent des données HTTP (données d'URL et données HTTP POST) envoyées au serveur Web. Le pilote Workflow Request Service peut fournir automatiquement certaines données connues (l'adresse du serveur Web, par exemple).

Les modèles sont traités par des feuilles de style XSLT. Ces dernières sont distinctes des feuilles de style utilisées comme règles DirXML sur les canaux Abonné ou Éditeur.

Les données de remplacement sont fournies à la feuille de style XSLT sous forme de paramètre. Le résultat du traitement de la feuille de style est un document XML, HTML ou texte. Il sera utilisé comme corps d'un message électronique, comme page Web ou sera envoyé à DirXML sur le canal Éditeur.

Les données de remplacement sont acheminées du canal Abonné vers le canal Éditeur via l'URL qui figure dans le message électronique. L'URL comprend une section requête qui contient les éléments de données de remplacement.

Le pilote Workflow Request Service est livré avec des feuilles de style prédéfinies capables de traiter des modèles qui permettent de créer des documents de messagerie électronique, des documents HTML et XDS. D'autres feuilles de style personnalisées peuvent être créées afin de fournir, au besoin, d'autres options de traitement.

Une méthode avancée de création de documents est également disponible. Cette méthode utilise uniquement une feuille de style XSLT et des données de remplacement. Aucun modèle n'intervient dans la procédure. Toutefois, le présent document utilise la méthode avec modèle car elle est plus facile à configurer et à gérer et ne requiert pas de connaissance de la programmation XSLT.

Modèles

Cette section décrit les modèles de création de documents tels qu'ils sont utilisés dans le pilote Workflow Request Service.

Les modèles sont des documents XML qui sont traités par une feuille de style afin de générer un document final. Ce dernier peut être un document XML, HTML ou un document texte ordinaire (ou tout autre document qui peut être produit à l'aide de XSLT).

Les modèles permettent de générer le texte de messages électroniques sur le canal Abonné ainsi que des pages Web dynamiques et des documents XDS sur le canal Éditeur.

Les modèles contiennent du texte, des éléments et des jetons de remplacement. Les jetons de remplacement sont remplacés dans le document final par les données fournies à la feuille de style qui traite le modèle.

Plusieurs exemples de modèles (qui ont des fonctions diverses) sont présentés ci-dessous. Dans ces exemples, les jetons de remplacement correspondent aux chaînes de caractères comprises entre les deux signes $ et apparaissant en **gras**.

Les modèles peuvent également contenir des *éléments d'opération*. Ce sont des éléments de commande interprétés par la feuille de style de traitement des modèles. Les éléments d'opération sont décrits à l'*Annexe C*. Dans les exemples suivants, les éléments d'opération apparaissent également en **gras**.

Le modèle de l'exemple suivant permet de produire le corps d'un message électronique au format HTML :

```
<html xmlns:form="http://www.novell.com/dirxml/workflow/form">
<head></head>
<body>
Dear $manager$,<p/>
<p>
Nous vous informons du recrutement de <b>$given-name$ $surname$</b>
<p>
Vous devez lui assigner un numéro de bureau. Pour ce faire, cliquez <a href="$url$">
ici</a>.
</p>
<p>
En vous remerciant,<br/>
Service des Ressources Humaines
</p>
</body>
</html>
```

Le modèle de l'exemple suivant permet de produire le corps d'un message électronique au format texte ordinaire :

```
<form:text xmlns:form="http://www.novell.com/dirxml/workflow/form">

Cher $manager$,

Nous vous informons du recrutement de $given-name$ $surname$

Vous devez lui assigner un numéro de bureau. Pour ce faire, utilisez le lien suivant :

$url$

En vous remerciant,

Service des Ressources Humaines

</form:text>
```

L'élément <form:text> est obligatoire dans la mesure où les modèles doivent être des documents XML. L'élément <form:text> est supprimé lors du traitement du modèle.

Le modèle suivant permet de produire un formulaire HTML utilisé comme page Web pour l'entrée de données :

```
<html  xmlns:form="http://www.novell.com/dirxml/workflow/form">
  <head>
    <title>Entrez le numéro de bureau de $subject-name$</title>
  </head>
  <body>
    <link href="novdocmain.css" rel="style sheet" type="text/css"/>
    <br/><br/><br/><br/>
    <form class="myform" METHOD="POST" ACTION="$url-base$/process_template.xsl">
        <table cellpadding="5" cellspacing="10" border="1" align="center">
          <tr><td>
          <input TYPE="hidden" name="template" value="post_form.xml"/>
          <input TYPE="hidden" name="subject-name" value="$subject-name$"/>
          <input TYPE="hidden" name="association" value="$association$"/>
          <input TYPE="hidden" name="response-style sheet" value="process_template.xsl"/>
          <input TYPE="hidden" name="response-template" value="post_response.xml"/>
          <input TYPE="hidden" name="auth-style sheet" value="process_template.xsl"/>
          <input TYPE="hidden" name="auth-template" value="auth_response.xml"/>
          <input TYPE="hidden" name="protected-data" value="$protected-data$"/>
          Vous êtes :<br/>
          <form:if-single-item name="responder-dn">
            <input TYPE="hidden" name="responder-dn" value="$responder-dn$"/>
            $responder-dn$
          </form:if-single-item>
          <form:if-multiple-items name="responder-dn">
            <form:menu name="responder-dn"/>
          </form:if-multiple-items>
        </td></tr>
        <tr><td>
          Entrez votre mot de passe : <br/>
<input name="password" TYPE="password" SIZE="20" MAXLENGTH="40"/>
        </td></tr>
        <tr><td>
          Entrez le numéro de bureau de $subject-name$ :<br/>
          <input TYPE="text" NAME="room-number" SIZE="20" MAXLENGTH="20"
value="$query:roomNumber$"/>
        </td></tr>
        <tr><td>
          <input TYPE="submit" value="Submit"/> <input TYPE="reset" value="Clear"/>
        </td></tr>
      </table>
    </form>
  </body>
</html>
```

Le modèle suivant permet de produire un document XDS :

```
<nds>
  <input>
    <modify class-name="User" src-dn="not-applicable">
      <association>$association$</association>
      <modify-attr attr-name="roomNumber">
        <remove-all-values/>
        <add-value>
          <value>$room-number$</value>
        </add-value>
      </modify-attr>
    </modify>
  </input>
</nds>
```

Jetons de remplacement

Les éléments délimités par des signes $ dans les exemples de modèles ci-dessus sont des jetons de remplacement. Par exemple, $manager$ sera remplacé par les données de remplacement appelées « manager ».

Les jetons de remplacement peuvent apparaître dans du texte ou des valeurs d'attributs XML (notez la valeur href dans l'élément <a> du premier exemple ci-dessus).

Données de remplacement

Les données de remplacement sont des chaînes qui vont prendre la place des jetons de remplacement dans le document final généré à partir d'un modèle. Ces chaînes proviennent soit de données du canal Abonné, soit de données HTTP du canal Éditeur ou sont automatiquement fournies par le pilote. Il existe un autre type de données de remplacement : celles qui sont récupérées de eDirectory via DirXML (données de requête).

Les données de remplacement sont décrites en détail à l'*Annexe A*.

Données du canal Abonné : Les données de remplacement qui proviennent du canal Abonné sont de deux types. Le premier fournit les valeurs des jetons de remplacement contenus dans les modèles utilisés pour créer des messages électroniques. Le second est placé dans la partie requête d'une URL pour que les données puissent être utilisées sur le canal Éditeur lorsque l'URL est soumise au serveur Web du canal Éditeur.

Données HTTP : Les données de remplacement sont fournies au serveur Web du canal Éditeur soit sous la forme de données de chaîne de requête d'URL, soit sous la forme de données HTTP POST, soit sous les deux formes.

Données automatiques : Le pilote Workflow Request Service fournit des données automatiques. Ces données sont décrites à l'*Annexe B*.

Données de requête : Les jetons de remplacement qui commencent par `query`: sont considérés comme des demandes de données actuelles auprès de eDirectory. La partie du jeton qui suit `query`: est le nom d'un attribut d'objet eDirectory. L'objet sur lequel porte la requête est spécifié par l'un des éléments de données de remplacement suivants : `association`, `src-dn` ou `src-entry-id`. Ces éléments sont considérés dans l'ordre indiqué dans la phrase précédente.

Éléments d'opération contenus dans les modèles

Les éléments d'opération sont des éléments qualifiés par un espace de nom. Ils figurent dans le modèle et sont utilisé pour une commande simple ou pour créer des éléments HTML pour des formulaires HTML. L'espace de nom utilisé pour qualifier les éléments est http://www.novell.com/dirxml/workflow/form. Dans le présent document et dans les exemples de modèles fournis avec le pilote Workflow Request Service, le préfixe utilisé est `form`.

Les éléments qui apparaissent en **gras** dans les exemples ci-dessus sont des éléments d'opération.

Les éléments d'opération sont décrits en détail à l'*Annexe C*.

Messages électroniques du canal Abonné

Le canal Abonné du pilote Workflow Request Service est destiné à l'envoi de messages électroniques. Pour ce faire, le pilote prend en charge un élément XML personnalisé appelé <mail>. Les règles du canal Abonné construisent un élément <mail> en réponse à un événement eDirectory donné (comme la création d'un utilisateur). Un exemple d'élément <mail> est fourni ci-dessous :

```
<mail src-dn="\PERIN-TAO\novell\Provo\Joe">
  <to>JStanley@novell.com</to>
  <cc>carol@novell.com</cc>
  <reply-to>HR@novell.com</reply-to>
  <subject>Assignation de bureau nécessaire pour : Joe le stagiaire</subject>
  <message mime-type="text/html">
    <stylesheet>process_template.xsl</stylesheet>
    <template>html_msg_template.xml</template>
    <replacement-data>
      <item name="manager">JStanley</item>
      <item name="given-name">Joe</item>
      <item name="surname">le stagiaire</item>
      <url-data>
        <item name="file">process_template.xsl</item>
        <url-query>
          <item name="template">form_template.xml</item>
          <item name="responder-dn" protect="yes">\PERIN-TAO\big-org\phb</item>
          <item name="responder-dn" protect="yes">\PERIN-TAO\big-org\carol</item>
          <item name="subject-name">Joe le stagiaire</item>
        </url-query>
      </url-data>
    </replacement-data>
    <resource cid="css-1">novdocmain.css</resource>
  </message>
  <message mime-type="text/plain">
    <stylesheet>process_text_template.xsl</stylesheet>
    <template>txt_msg_template.xml</template>
    <replacement-data>
      <item name="manager">JStanley</item>
      <item name="given-name">Joe</item>
      <item name="surname">le stagiaire</item>
      <url-data>
        <item name="file">process_template.xsl</item>
        <url-query>
          <item name="template">form_template.xml</item>
          <item name="responder-dn" protect="yes">\PERIN-TAO\big-org\phb</item>
          <item name="responder-dn" protect="yes">\PERIN-TAO\big-org\carol</item>
          <item name="subject-name">Joe le stagiaire</item>
        </url-query>
      </url-data>
    </replacement-data>
  </message>
  <attachment>HR.gif</attachment>
</mail>
```

Le canal Abonné du pilote Workflow Request Service utilise les informations contenues dans l'élément <mail> pour construire un message électronique SMTP. Une URL peut être créée et insérée dans le message électronique pour permettre au destinataire du message d'y répondre. L'URL peut pointer sur le serveur Web du canal Éditeur ou sur un autre serveur Web.

L'élément <mail> et son contenu sont décrits en détail à l'*Annexe D*.

Serveur Web du canal Éditeur

Le canal Éditeur du pilote Workflow Request Service exécute un serveur Web configuré de manière à permettre aux utilisateurs d'entrer des données dans eDirectory via un navigateur Web. Le serveur Web fonctionne avec les messages électroniques envoyés par le canal Abonné du pilote Workflow Request Service.

Le serveur Web du canal Éditeur peut fournir des fichiers statiques et des contenus dynamiques. Les fichiers statiques comprennent par exemple les feuilles de style .css, les images, etc. Les contenus dynamiques sont par exemple les pages Web qui changent selon les données de remplacement contenues dans les données d'URL ou HTTP POST.

Le serveur Web du canal Éditeur est normalement configuré pour permettre à un utilisateur d'entrer des données dans eDirectory en réponse à un message électronique envoyé par le canal Abonné. Une interaction typique de l'utilisateur avec le serveur Web se ferait comme suit :

1. L'utilisateur soumet l'URL du message au serveur Web à partir d'un navigateur Web. L'URL précise la feuille de style, le modèle et les données de remplacement utilisés pour créer une page Web dynamique (qui contient généralement un formulaire HTML).
2. Le serveur Web crée une page HTML en traitant le modèle à l'aide de la feuille de style et des données de remplacement. La page HTML est renvoyée au navigateur Web de l'utilisateur comme étant la ressource désignée par l'URL.
3. Le navigateur affiche la page HTML et l'utilisateur entre les informations requises.
4. Le navigateur envoie une requête HTTP POST qui contient les informations entrées ainsi que d'autres informations issues de l'URL du message électronique. Le DN et le mot de passe de l'utilisateur qui répond au message doivent figurer dans les données POST.
5. Le serveur Web authentifie l'utilisateur à l'aide de son DN et de son mot de passe. Si l'authentification échoue, une page Web contenant un message d'échec est renvoyée comme résultat de la requête POST. Le message d'échec peut être construit à l'aide de la feuille de style et du modèle spécifiés dans les données POST. Si l'authentification réussit, le traitement se poursuit.
6. Le serveur Web construit un document XDS à l'aide de la feuille de style et du modèle spécifiés dans les données POST. Ce document est soumis à DirXML sur le canal Éditeur.
7. Le résultat de la soumission du document XDS, conjugué à la feuille de style et au modèle spécifiés dans les données POST, permet de construire une page Web qui indique à l'utilisateur le résultat de la soumission des données. La page Web est ensuite envoyée au navigateur en réponse à la requête POST.

Configuration

Cette section décrit la configuration des paramètres et des modèles du pilote Workflow Request Service.

Paramètres du pilote

Configuration du pilote

Cette section décrit les paramètres qui apparaissent sous « Configuration du pilote » dans l'interface utilisateur de l'objet Pilote.

Plusieurs de ces paramètres concernent en fait le serveur Web du canal Éditeur. Ils apparaissent à la section Configuration du pilote dans la mesure où le canal Abonné du pilote Workflow Request Service a également besoin d'y accéder.

DN de la base de documents

Ce paramètre est le DN eDirectory d'un objet Conteneur. Le pilote Workflow Request Service peut charger des documents XML (y compris des feuilles de style XSLT) à partir de eDirectory mais aussi à partir d'un disque. Si vous devez charger des documents XML à partir de eDirectory, ce paramètre identifie le conteneur racine à partir duquel les documents seront chargés.

Les documents chargés à partir de eDirectory résident dans la valeur d'attribut d'un objet eDirectory. Si aucun attribut n'est pas précisé, XmlData est utilisé par défaut. L'attribut peut être spécifié en ajoutant au nom de l'objet qui contient le document le caractère dièse (#) suivi du nom de l'attribut.

Par exemple, supposons que le DN de base de documents spécifié soit « novell\Workflow Documents » et que « Workflow Documents » contienne un conteneur appelé « templates ».

Si un objet Feuille de style DirXML nommé « e-mail _template » réside dans le répertoire « templates », les identificateurs de ressource suivants peuvent être utilisés pour désigner le document XML : « templates/e-mail _template » ou « templates/e-mail _template#XmlData ».

Les identificateurs de ressource peuvent être fournis sous la forme de données de remplacement, données d'URL ou données HTTP POST. Par exemple, l'élément suivant peut apparaître sous un élément <message> sur le canal Abonné :

```
<template>templates/e-mail _template#XmlData</template>
```

Répertoire des documents

Ce paramètre identifie un répertoire de système de fichiers qui sert de répertoire de base pour la localisation des ressources telles que les modèles, les feuilles de style XSLT et d'autres ressources fichier fournies par le serveur Web du canal Éditeur. Exemples de valeurs :

`c:\Novell\Nds\workflow_files`	Windows
`SYS:\SYSTEM\workflow_files`	NetWare
`/usr/lib/dirxml/workflow_files`	Unix

Utiliser le serveur HTTP (true|false)

Ce paramètre indique si le canal Éditeur doit exécuter ou non un serveur Web. La valeur *true* doit être affectée à paramètre si le serveur Web doit être exécuté. Dans le cas contraire, la valeur *false* doit être utilisée.

Afin d'économiser les ressources système, si le pilote Workflow Request Service doit uniquement être utilisé pour l'envoi de messages électroniques contenant une URL de réponse ou une URL qui pointe sur une autre application, le serveur HTTP ne doit pas être exécuté.

Adresse IP HTTP ou nom d'hôte

Ce paramètre vous permet de spécifier à quelle adresse, parmi les multiples adresses IP locales, le serveur Web du canal Éditeur écoutera les requêtes HTTP.

Si vous ne renseignez pas ce champ, le serveur Web du canal Éditeur écoutera à l'adresse IP par défaut. Pour les serveurs dotés d'une adresse IP unique, cela est suffisant. Si vous indiquez une adresse IP à points comme valeur du paramètre, le serveur Web du canal Éditeur écoutera les requêtes HTTP à l'adresse spécifiée.

Notez que la valeur spécifiée pour l'adresse IP HTTP ou le nom d'hôte est utilisée par le gestionnaire de messagerie du canal Abonné pour construire les URL si aucun nom d'hôte ni adresse n'est précisé dans l'élément de commande de courrier. Si le paramètre Utiliser le serveur HTTP (true|false) a pour valeur *false*, l'adresse IP ou le nom d'hôte HTTP peuvent être utilisés pour spécifier l'adresse ou le nom d'un serveur Web utilisable pour construire des URL pour les messages électroniques.

Port HTTP

Ce paramètre est un nombre entier qui indique le port TCP sur lequel le serveur Web du canal Éditeur doit écouter les requêtes entrantes. Si cette valeur n'est pas précisée, le numéro de port par défaut est 80 ou 443, selon que SSL est ou non utilisé pour les connexions au serveur Web.

Si le pilote Workflow Request Service s'exécute sur le serveur DirXML (c'est-à-dire, s'il n'est pas exécuté auprès du chargeur distant DirXML sur une machine distante), le numéro du port HTTP doit être différent de 80 et de 443 car iMonitor ou un autre processus utilise généralement les ports 80 et 443.

Nom du KMO

Ce paramètre, lorsqu'il est renseigné, indique le nom de l'objet Matériel clé (KMO) eDirectory qui contient le certificat et la clé de serveur utilisés pour SSL par le serveur Web du canal Éditeur.

La définition de ce paramètre oblige le serveur Web du canal Éditeur à utiliser SSL pour traiter les requêtes HTTP.

Il a priorité sur tout autre paramètre Keystore Java (voir ci-dessous).

L'utilisation de SSL est recommandée pour des raisons de sécurité dans la mesure où les mots de passe eDirectory sont transmis sous la forme de données HTTP POST lors de l'utilisation du serveur Web du canal Éditeur.

Nom du fichier Keystore

Ce paramètre, conjointement avec les paramètres `Mot de passe Keystore`, `Nom du certificat (alias de la clé)` et `Mot de passe du certificat (mot de passe de la clé)` permet de préciser le fichier Keystore Java qui contient le certificat et la clé utilisés pour SSL par le serveur Web du canal Éditeur.

La définition de ce paramètre oblige le serveur Web du canal Éditeur à utiliser SSL pour traiter les requêtes HTTP.

Si `Nom du KMO` est défini, ce paramètre et ses paramètres associés sont ignorés.

L'utilisation de SSL est recommandée pour des raisons de sécurité dans la mesure où les mots de passe eDirectory sont transmis sous la forme de données HTTP POST lors de l'utilisation du serveur Web du canal Éditeur.

Mot de passe Keystore

Ce paramètre précise le mot de passe du fichier Keystore Java spécifié par `Nom du fichier Keystore`.

Nom du certificat (alias de la clé)

Ce paramètre précise le nom du certificat à utiliser dans le fichier Keystore Java spécifié par `Nom du fichier Keystore`.

Mot de passe du certificat (mot de passe de la clé)

Ce paramètre précise le mot de passe correspondant au certificat spécifié à l'aide de `Nom du certificat (alias de la clé)`.

Configuration du canal Abonné

Serveur SMTP

Ce paramètre précise le nom du serveur SMTP utilisé par le canal Abonné pour l'envoi des messages électroniques.

Nom de compte SMTP

Si l'authentification est requise pour le serveur SMTP spécifié par « Serveur SMTP », ce paramètre précise le nom du compte à utiliser à ces fins. Le mot de passe utilisé est le « Mot de passe d'application » associé aux paramètres d'authentification du pilote.

Adresse « de » par défaut

Lorsqu'elle est précisée, cette adresse électronique est celle utilisée dans le champ SMTP « De » pour les messages électroniques envoyés par le canal Abonné. Lorsque cette adresse n'est pas indiquée, les éléments <mail> envoyés au canal Abonné doivent contenir un élément <from>.

Tout élément <from> contenu dans des éléments <mail> envoyés au canal Abonné a priorité sur ce paramètre.

Gestionnaires supplémentaires

Lorsqu'il est précisé, ce paramètre correspond à une liste de noms de classes Java séparés par des blancs. Chaque nom est celui d'une classe personnalisée qui implémente l'interface `com.novell.nds.dirxml.driver.workflow.CommandHandler` et traite un élément XDS personnalisé. (Le gestionnaire qui traite l'élément <mail> est un gestionnaire intégré.)

D'autres informations sur les gestionnaires personnalisés sont disponibles à l'Annexe F.

Configuration du canal Éditeur

Servlets supplémentaires

Lorsque ce paramètre est précisé, il correspond à une liste de noms de classes Java séparés par des blancs. Chaque nom est celui d'une classe personnalisée qui étend `javax.servlet.http.HttpServer`. Vous pouvez utiliser des servlets personnalisées pour étendre les fonctionnalités du serveur Web du canal Éditeur.

D'autres informations sur les servlets personnalisées sont disponibles à l'Annexe G.

Règles du canal Abonné

La configuration des règles du canal Abonné dépend des opérations que l'on souhaite effectuer à l'aide du pilote Workflow Request Service. Toutefois, certaines recommandations peuvent être utiles.

En général, placer la construction d'un élément <mail> à envoyer au canal Abonné dans la règle de transformation de commande est le meilleur moyen de procéder pour la raison suivante : l'essentiel du traitement du moteur DirXML est terminé au moment où les commandes sont transmises à la règle de transformation de commande. Cela signifie que les règles de création ont été traitées pour les événements d'ajout (ce qui permet par exemple d'opposer un veto sur les événements d'ajout d'objets qui ne possèdent pas tous les attributs nécessaires à la construction du message électronique). Cela signifie également que les événements de modification d'objets sans associations ont déjà été convertis en événements d'ajout.

La feuille de style XSLT qui construit le message électronique peut avoir besoin d'interroger eDirectory pour obtenir des informations supplémentaires.

Par exemple, si le message électronique est simplement un message de bienvenue destiné à un employé nouvellement embauché, la commande d'ajout peut contenir toutes les informations nécessaires : prénom, nom et adresse de messagerie Internet. Pour cela, précisez ces informations dans la règle de création en tant qu'attributs obligatoires. Ceci garantit que seules les commandes d'ajout qui contiennent les informations nécessaires seront transmises à la règle de transformation de commande.

Si toutefois le message électronique est un message adressé au responsable d'un employé, la feuille de style devra interroger eDirectory. Le DN du responsable s'obtient auprès de l'événement d'ajout de l'objet Utilisateur de l'employé, mais une requête doit être envoyée afin d'obtenir son adresse électronique car ces informations seront un attribut de l'objet Utilisateur du responsable.

En outre, si des notifications par courrier électronique sont générées suite au résultat des commandes de modification d'objets associés au pilote, des requêtes doivent être envoyées afin d'obtenir les informations non contenues dans la commande de modification.

Blocage des commandes pour les empêcher d'atteindre le canal Abonné

Si des messages électroniques doivent être générés à partir d'événements différents des événements d'ajout, les événements d'ajout doivent être autorisés à atteindre le canal Abonné associé aux objets à surveiller. Autoriser les événements d'ajout à atteindre le canal Abonné permet de créer une valeur d'association qui est renvoyée à DirXML par le canal Abonné.

Il est important que les objets eDirectory qui doivent être surveillés par les règles du pilote Workflow Request Service soient dotés d'une association à ce pilote. Seuls les objets dotés d'une association verront les événements de suppression, de réassignation de nom et de déplacement signalés au pilote. En outre, les événements de modification sur des objets sans association sont convertis en événements d'ajout après la transformation de l'événement sur le canal Abonné.

Toutes les autres commandes (modification, déplacement, réassignation de nom et suppression) doivent être verrouillées par la règle de transformation de commande afin de les empêcher d'atteindre le canal Abonné. Le canal Abonné traite uniquement les commandes <add> et <mail>. Pour les autres commandes, il renvoie une erreur.

Génération de messages électroniques

Des messages électroniques sont envoyés par le canal Abonné en réponse à la réception d'un élément <mail> qui décrit le message électronique à envoyer. Pour une description de l'élément <mail> et de son contenu, reportez-vous à l'*Annexe D*.

Des messages électroniques peuvent être générés en réponse à tout événement DirXML (ajout, modification, réassignation de nom, déplacement, suppression).

Les données de remplacement fournies avec les éléments <message> enfant d'un élément <mail> dépendront de deux facteurs principaux :

- Le modèle utilisé pour générer le corps du message. Les données de remplacement devant être utilisées par le modèle de message électronique apparaissent en tant qu'enfants de l'élément <replacement-data>.
- Les informations nécessaires aux modèles de page Web sur le canal Éditeur si le message électronique doit donner lieu à une réponse sur ce canal. Les données de remplacement devant être utilisées par les modèles de page Web apparaissent en tant qu'enfants de l'élément <url-query>, lui-même enfant de l'élément <url-data>, à son tour enfant de l'élément <replacement-data>.

Si le message électronique doit contenir une URL qui pointe sur le serveur Web du canal Éditeur et est utilisée pour solliciter des informations auprès d'un utilisateur, les données de remplacement doivent contenir au moins un élément `responder-dn`. Les valeurs des éléments « `responder-dn` » doivent correspondre aux DN des objets Utilisateur des utilisateurs auxquels le message est envoyé.

Si un jeton de remplacement de requête (voir *Données de remplacement* à la section *Présentation*) est utilisé dans le modèle, les données de remplacement correspondant à l'élément <message> doivent contenir un élément de données `src-dn`, `src-entry-id` ou `association` associé à la valeur appropriée. Une élément de données d'association peut être utilisé uniquement si l'objet eDirectory sur lequel porte la requête est déjà doté d'une association au pilote Workflow Request Service. L'association générée par le canal Abonné pour les objets sans association ne peut pas être utilisée car elle n'aura pas encore été créée dans l'objet eDirectory au moment de la requête.

L'élément <message> peut préciser le type MIME du corps du message. Si le type MIME est précisé, mais qu'aucune feuille de style n'est indiquée (c'est-à-dire s'il n'existe pas d'élément <stylesheet> enfant de l'élément <message>), l'un des deux noms de feuille de style par défaut est utilisé. Si le type MIME a pour valeur text/plain, le nom de la feuille de style par défaut est process_text_template.xsl. Si le type MIME a une valeur autre que text/plain , le nom de la feuille de style par défaut est process_template.xsl.

Modèles de messages électroniques du canal Abonné

Les modèles de messages électroniques sont des documents XML qui contiennent du texte standard et des jetons de remplacement. Ils permettent de générer le texte du corps d'un message. Pour des informations générales sur les modèles, reportez-vous à la section *Modèles* sous *Présentation*.

Les jetons de remplacement utilisés dans un modèle de message électronique déterminent les éléments <item> qui doivent être fournis en tant qu'enfants de l'élément <replacement-data> construit par la règle du canal Abonné qui construit l'élément <mail>. Par exemple, si le modèle de message électronique contient le jeton de remplacement $employee-name$, les données de remplacement doivent contenir un élément <item name="employee-name"> pour l'élément <message>. En l'absence du nom de l'employé (employee-name), le corps du message électronique résultant ne contiendra pas de texte à l'endroit occupé par le jeton de remplacement dans le modèle.

Les modèles de messages électroniques peuvent être utilisés pour générer des corps de messages au format texte ordinaire, HTML ou XML.

Si un modèle de message électronique génère un message au format texte ordinaire, il doit être traité par une feuille de style qui précise « texte ordinaire » comme type de sortie. Si la feuille de style ne précise pas qu'il s'agit de texte ordinaire, des caractères d'échappement XML indésirables seront générés. La feuille de style par défaut du pilote Workflow Request Service, process_text_template.xsl, est généralement utilisée pour traiter les modèles qui produisent des documents au format texte ordinaire.

Règles du canal Éditeur

Dans la plupart des implémentations du pilote Workflow Request Service, aucune règle de canal Éditeur n'est nécessaire. Ceci s'explique par le fait qu'il est possible de construire la page Web et les modèles XDS de manière à ce qu'ils produisent exactement le document XDS requis. De plus, ce document ne nécessite aucun traitement supplémentaire par des règles.

Si des règles sont obligatoires, elles seront spécifiques à chaque installation.

Modèles de pages Web du canal Éditeur

Les modèles de pages Web sont des documents XML qui contiennent du texte standard et des jetons de remplacement. Ils permettent de générer des documents de pages Web (généralement des documents HTML). Pour des informations générales sur les modèles, reportez-vous à la section *Modèles* sous *Présentation*.

Les jetons de remplacement utilisés dans les modèles de pages Web déterminent quelles données de remplacement sont fournies sous la forme de données de requête d'URL sur le canal Abonné. Sur le canal Éditeur, les données de remplacement sont obtenues à partir de la chaîne de requête d'URL correspondant aux requêtes HTTP GET, et à partir de la chaîne de requête d'URL et des données POST correspondant aux requêtes HTTP POST.

Considérez par exemple le flux de données de remplacement illustré ci-après, dans lequel les données sont transmises du canal Abonné, au message électronique avant d'atteindre le serveur Web du canal Éditeur :

> Le pilote Workflow Request Service est configuré de manière à demander au responsable d'un nouvel employé de lui assigner un numéro de bureau. L'envoi du message électronique au responsable est déclenché par la commande d'ajout (<add>) d'un nouvel objet Utilisateur traité par la règle de transformation de commande du canal Abonné.
>
> Lorsque le responsable clique sur l'URL qui figure dans le message, une page Web s'affiche dans son navigateur Web. Cette page doit indiquer pour qui le responsable doit entrer un numéro de bureau.
>
> Pour ce faire, l'élément <url-query> sur le canal Abonné contient un élément de données de remplacement qui identifie le nouvel employé par son nom :
>
> ```
> <item name="subject-name">Joe le stagiaire</item>
> ```
>
> Ainsi, la chaîne de requête d'URL contient obligatoirement (entre autres choses) « subject-name=Joe%20le%20stagiaire ». (%20 représente un espace codé URL.)
>
> Le navigateur Web du responsable soumet l'URL au serveur Web du canal Éditeur dès que le responsable clique sur l'URL dans le message. Le serveur Web construit l'élément de données de remplacement « subject-name » associé à la valeur « Joe le stagiaire ».
>
> Le modèle de page Web également spécifié par l'URL contient le jeton de remplacement $subject-name$. Lorsque le modèle de page Web est traité par la feuille de style pour construire la page Web, le jeton de remplacement est remplacé par « Joe le stagiaire », qui personnalise la page Web correspondant à l'employé dont la création de l'objet Utilisateur a causé l'envoi du message électronique.

Pour des informations supplémentaires sur une transaction complète entre le canal Abonné et le canal Éditeur, reportez-vous à l'*Annexe E*.

Modèles XDS du canal Éditeur

Les modèles XDS sont des documents XML qui contiennent du texte standard et des jetons de remplacement. Ils permettent de générer des documents XDS qui sont soumis à DirXML sur le canal Éditeur du pilote Workflow Request Service. Pour des informations générales sur les modèles, reportez-vous à la section *Modèles* sous *Présentation*.

Les jetons de remplacement utilisés dans les modèles XDS déterminent certaines données de remplacement qui sont fournies au serveur Web sous la forme de données de requête HTTP POST.

Considérez par exemple le modèle XDS suivant :

```
<nds>
  <input>
    <modify class-name="User" src-dn="not-applicable">
      <association>$association$</association>
      <modify-attr attr-name="roomNumber">
        <remove-all-values/>
        <add-value>
          <value>$room-number$</value>
        </add-value>
      </modify-attr>
    </modify>
  </input>
</nds>
```

Les jetons de remplacement utilisés dans le modèle indiquent que les données HTTP POST doivent fournir une valeur d'association et une valeur de numéro de bureau.

Normalement, la valeur d'association provient du canal Abonné. Le message électronique du canal Abonné place « association=*une certaine valeur* » dans la chaîne de requête de l'URL insérée dans le message. Le modèle de page Web utilisé pour générer la page Web lors de la soumission de l'URL au serveur Web place généralement la valeur d'association dans un élément INPUT masqué :

```
<INPUT TYPE="hidden" NAME="association" VALUE="$association$"/>
```

L'inclusion de la valeur d'association sous forme d'élément INPUT masqué entraîne la soumission de la paire « association=*une certaine valeur »* avec les données HTTP POST.

La valeur de numéro de bureau est entrée dans la page Web à l'aide d'un élément INPUT similaire au suivant :

```
<input TYPE="text" NAME="room-number" SIZE="20" MAXLENGTH="20"/>
```

Si le responsable entre 1234 et clique sur Soumettre, le navigateur Web envoie « room-number=1234 » avec les données HTTP POST.

Le serveur Web génère ensuite deux éléments de données de remplacement, <item name=“association”> et <item name=“room-number”>, qui sont utilisés pour le traitement du modèle XDS.

Le document XDS est généré en traitant le modèle XDS à l'aide de la feuille de style spécifiée dans les données POST. Ensuite, le document XDS est soumis à DirXML sur le canal Éditeur du pilote Workflow Request Service.

Configuration du niveau de trace

Le pilote Workflow Request Service produit des messages plus ou moins détaillés selon le niveau de trace défini :

Niveau	Description du message de trace
0	Pas de message de trace
1	Messages d'une seule ligne correspondant au suivi d'une opération de base
2	Aucun message supplémentaire (le moteur DirXML effectue le suivi des documents XML à ce niveau et aux niveaux supérieurs)
3	Aucun message supplémentaire
4	Messages liés à la construction du document à partir de modèles et de feuilles de style
5	Suivi des documents de données de remplacement

Annexe A – Données de remplacement

Les données de remplacement sont utilisées avec les documents XML qui servent de modèles pour la construction des messages électroniques, des pages Web et des documents XDS. Le remplacement s'effectue en fait lors du traitement du document modèle à l'aide d'une feuille de style XSLT qui exécute elle-même le remplacement dans le cadre de la construction du document final.

Les données de remplacement sont fournies au pilote Workflow Request Service par le biais de différents mécanismes sur les canaux Abonné et Éditeur.

Canal Abonné

- Les données de remplacement sont fournies avec l'élément <mail>.
- Une partie des données de remplacement fournies peuvent être des données d'URL. Si des données d'URL sont fournies, elles sont traitées et remplacées par des éléments de données automatiques (voir l'Annexe B).
- Si l'élément <mail> précise qu'une valeur d'association doit être construite (c'est-à-dire si l'élément <mail> possède un attribut *src-dn*), un élément de données automatique nommé « association » est ajouté aux données de remplacement.

Canal Éditeur

- Les données de remplacement sont fournies avec les données HTTP URL et HTTP POST.
- Les éléments de données de remplacement d'URL automatiques sont ajoutés aux données de remplacement avant leur utilisation dans le traitement du modèle.

Pendant le traitement du modèle, les données de remplacement se présentent sous la forme d'un document XML. Le document contenant les données de remplacement est transmis à la feuille de style qui traite le modèle sous la forme d'un paramètre nommé « replacement-data ». Si aucun modèle n'est utilisé, le document XML est traité directement par la feuille de style.

Sécurité des données

Les éléments de données sont acheminés du canal Abonné vers le canal Éditeur via une URL qui figure dans le message électronique envoyé par le canal Abonné. La modification de certains éléments de données de l'URL représente un risque pour la sécurité des données. Par exemple, si les valeurs « responder-dn » de l'URL fournie par le canal Abonné sont remplacées par le DN d'un autre utilisateur dans l'URL soumise au serveur Web du canal Éditeur, un utilisateur non autorisé pourra modifier des données dans eDirectory.

Pour garantir la correspondance exacte entre les données contenues dans l'URL soumise et celles fournies à l'origine par le canal Abonné, une section *données protégées* est fournie dans la chaîne de requête d'URL. Les données protégées sont des données qui ne peuvent pas être modifiées pour des raisons de sécurité. Elles varient selon la configuration, mais incluent toujours les éléments de données « responder-dn » ainsi que les éléments de données correspondant à tout objet eDirectory dont les valeurs doivent être modifiées.

Pour protéger les éléments de données ; il suffit de coder les valeurs d'origine et de placer les valeurs codées dans une chaîne de requête d'URL. Lorsque le serveur Web du canal Éditeur reçoit les valeurs codées, le canal Éditeur les décode et les compare aux éléments de données non codées fournis par une requête HTTP GET ou POST.

Si l'instance d'un élément de données figure parmi les données codées, une valeur d'élément de données non codées doit correspondre à l'une des valeurs d'élément de données codées. Si la valeur de l'élément de données non codées ne correspond à aucune valeur d'élément de données codées, la requête HTTP est rejetée par le serveur Web du canal Éditeur.

En outre, toute requête HTTP POST qui ne contient pas de données protégées est rejetée.

Exemple

Dans une requête HTTP POST, le serveur Web du canal Éditeur utilise les données POST non codées nommées « responder-dn » pour vérifier le mot de passe fourni par les données POST. Ceci permet d'authentifier l'utilisateur qui répond auprès de l'objet eDirectory de l'utilisateur.

Supposons que le contenu de l'élément <url-query> du canal Abonné précise deux éléments de données comme suit :

```
<item name="responder-dn" protect="yes">\PERIN-TAO\novell\phb</item>
<item name="responder-dn" protect="yes">\PERIN-TAO\novell\carol</item>
```

Les données protégées contenues dans l'URL générée par le canal Abonné contiendront deux valeurs « responder-dn ».

Supposons qu'un utilisateur malveillant obtienne l'URL générée et envoyée dans un message électronique. Il utilise ensuite cette URL pour obtenir le formulaire HTML qui permet aux utilisateurs de modifier les données d'un objet eDirectory.

Dans la requête HTTP POST soumise au serveur Web, l'utilisateur malveillant utilise son DN eDirectory (responder-dn=\PERIN-TAO\novell\wally) comme valeur « responder-dn » non codée. Il inclut également son propre mot de passe dans les données POST pour que l'authentification effectuée par le serveur Web réussisse.

Cependant, lorsque le serveur Web du canal Éditeur reçoit les données HTTP POST, il ne réussit pas à trouver « \PERIN-TAO\novell\wally » parmi les données protégées codées et rejette la requête POST.

Éléments XML

Les éléments qui composent un document de données de remplacement sont décrits ci-dessous. L'absence de description pour un élément indique qu'aucun attribut XML n'est autorisé pour cet élément.

<replacement-data>

L'élément <replacement-data> peut apparaître aux endroits suivants :

1) En tant qu'enfant de l'élément <message> sous un élément <mail> du canal Abonné. Le pilote Workflow Request Service convertit l'élément <replacement-data> fourni en élément <replacement-data> autonome utilisable dans le traitement des modèles. Le traitement suivant est effectué :

   a) Si une valeur d'association est créée pour l'élément <mail>, l'élément <item name="association"> est ajouté aux données de remplacement. La valeur de l'élément créé est la valeur d'association qui sera renvoyée à DirXML.

   b) Si l'élément <replacement-data> possède un élément <url-data> enfant, l'élément <url-data> est remplacé par plusieurs éléments <item> qui contiennent des données d'URL construites. Reportez-vous à <url-data> et <url-query>.

2) En tant qu'élément autonome de plus haut niveau d'un document de données de remplacement utilisé lors de la construction d'un document à l'aide d'une feuille de style sur le canal Abonné ou Éditeur.

<item>

L'élément <item> peut être un enfant de l'élément <replacement-data>, <url-data> ou <url-query>. Le contenu de l'élément <item> est le texte utilisé lors de la substitution des jetons de remplacement dans les modèles. Les éléments <item> sont toujours nommés à l'aide de l'attribut *name*.

Attributs <item>

name : La valeur de l'attribut *name* précise le nom sous lequel cet élément de données est désigné par les jetons de remplacement. Par exemple, si la valeur de l'attribut *name* est « manager », le jeton de remplacement $manager$ sera remplacé par la valeur contenue dans l'élément <item name="manager">. L'attribut *name* est obligatoire.

protect : Pour les éléments <item> qui sont des enfants d'éléments <url-query>, l'attribut *protect* précise si <item> sera ajouté à la section des données protégées de la chaîne de requête d'URL (voir <url-query>). Si l'attribut *protect* est présent, sa valeur doit être `yes`.

Noms <item> prédéfinis

Certains éléments <item> ont des significations prédéfinies pour le canal Abonné, le canal Éditeur ou pour les deux. Ces éléments sont décrits ici :

template : Le canal Éditeur traite la valeur de l'élément « template » comme le nom du document modèle à utiliser pour produire la réponse à une requête HTTP GET.

Lorsque <item name="template"> apparaît comme enfant de l'élément <url-query> sur le canal Abonné, la valeur est placée dans la requête d'URL pour préciser au serveur Web du canal Éditeur le nom du document modèle à utiliser pour répondre à la requête HTTP GET.

responder-dn : Le canal Éditeur utilise la valeur de l'élément « responder-dn » qui figure dans les données HTTP POST comme DN de l'objet eDirectory auprès duquel le mot de passe fourni dans les données HTTP POST est validé.

Le serveur Web rejette toute requête HTTP POST qui ne contient ni valeur « responder-dn », ni valeur « password ». En outre, si les données HTTP POST ne contiennent aucun élément de données protégées, la requête est rejetée.

Le canal Abonné fournit un ou plusieurs éléments <item name="responder-dn" protect="yes"> sous l'élément <url-query>. Comme les éléments « responder-dn » sont utilisés pour authentifier l'utilisateur, ils doivent être protégés.

password : Fourni par le serveur Web du canal Éditeur via les données HTTP POST. Le contenu de cet élément est le mot de passe qui est validé auprès de l'objet eDirectory spécifié par l'élément « responder-dn » contenu dans les données POST. L'élément « password » est normalement entré dans le formulaire HTML utilisé pour générer la requête HTTP POST.

Exemple :

```
<INPUT TYPE="password" NAME="password" SIZE="20" MAXLENGTH="40"/>
```

response-template : Fourni par le serveur Web via les données HTTP POST. Permet de générer la page Web utilisée en réponse à la requête POST. Cet élément est normalement spécifié par un élément INPUT masqué dans le formulaire HTML utilisé pour générer la requête HTTP POST.

Exemple :

```
<INPUT TYPE="hidden" NAME="response-template" VALUE="post_form.xml"/>
```

response-stylesheet : Fourni par le serveur Web via les données HTTP POST. Permet de générer la page Web utilisée en réponse à la requête POST. Cet élément est normalement spécifié par un élément INPUT masqué dans le formulaire HTML utilisé pour générer la requête HTTP POST.

Exemple :

```
<INPUT TYPE="hidden" NAME="response-stylesheet" VALUE="process_template.xsl"/>
```

auth-template : Fourni par le serveur Web via les données HTTP POST. Permet de générer la page Web utilisée en réponse à la requête POST si l'authentification de l'utilisateur échoue. Cet élément est normalement spécifié par un élément INPUT masqué dans le formulaire HTML utilisé pour générer la requête HTTP POST.

Exemple :

```
<INPUT TYPE="hidden" NAME="auth-template" VALUE="auth_response.xml"/>
```

auth-stylesheet : Fourni par le serveur Web via les données HTTP POST. Permet de générer la page Web utilisée en réponse à la requête POST si l'authentification de l'utilisateur échoue. Cet élément est normalement spécifié par un élément INPUT masqué dans le formulaire HTML utilisé pour générer la requête HTTP POST.

Exemple :

```
<INPUT TYPE="hidden" NAME="auth-stylesheet" VALUE="process_template.xsl"/>
```

protected-data : Cet élément de données contient les données codées construites par le canal Abonné. Sur le canal Abonné, il s'agit d'un élément de données fourni automatiquement.

Sur le canal Éditeur, l'élément de données « protected-data » est obtenu à partir de la chaîne de requête d'URL correspondant à une requête HTTP GET, et à partir des données POST correspondant à une requête HTTP POST.

L'élément « protected-data » est généralement transmis de la requête HTTP GET à la page Web utilisée pour générer la requête HTTP POST via un jeton de remplacement contenu dans le modèle utilisé pour construire la réponse à la requête HTTP GET.

Exemple :

```
<INPUT TYPE="hidden" NAME="protected-data" VALUE="$protected-data$"/>
```

<url-data>

L'élément <url-data> est un enfant de l'élément <replacement-data> qui se figure sous l'élément <message> sur le canal Abonné. Il contient les éléments <item> qui permettent de construire l'URL et les éléments de données associés fournis au modèle utilisé pour la construction du message électronique. Il contient également l'élément <url-query>.

Pour les besoins du pilote Workflow Request Service, les URL se composent de cinq parties :

1. Un protocole tel que http, https, ftp, etc.
2. Un hôte tel que www.novell.com ou 192.168.0.1
3. Un numéro de port. Deux-points, suivis d'un entier décimal. Par exemple, :80 ou :8180.
4. Un fichier ou un identificateur de ressource. Il s'agit généralement d'un nom de fichier ; peut inclure un chemin. Par exemple, stylesheets/process_template.xsl.
5. Une chaîne de requête. Il s'agit d'un ensemble de paires « name-value », séparées par des caractères &. Par exemple, template=form_template.xml&protected-data=AabABJKEL=

Noms <item> prédéfinis sous <url-data>

Les éléments <item> qui figurent sous <url-data> sont ignorés à l'exception de ceux listés ci-après. Tous sont facultatifs.

file : Précise la partie fichier de l'URL. Lorsqu'il est utilisé avec le serveur Web du canal Éditeur, l'élément « file » précise la feuille de style à utiliser pour construire la page HTML initiale renvoyée en réponse à l'URL. Lorsqu'il est utilisé avec un serveur autre que le serveur Web du canal Éditeur, l'élément « file » précise le nom de la ressource que désignera l'URL.

Si l'élément « file » n'apparaît pas, la partie fichier de l'URL est défaut process_template.xsl.

scheme : Élément facultatif qui se figure sous l'élément <url-data>. S'il est présent, il spécifie la partie protocole de l'URL (http ou ftp). En général, l'élément « scheme » est utilisé uniquement si l'URL doit pointer sur un serveur autre que le serveur Web du canal Éditeur.

Si cet élément n'apparaît pas, la partie protocole de l'URL est par défaut http ou https, selon la configuration du serveur Web du canal Éditeur.

host : Élément facultatif qui se figure sous l'élément <url-data>. S'il est présent, il précise la partie hôte de l'URL. En général, l'élément « host » est utilisé uniquement si l'URL doit pointer sur un serveur autre que le serveur Web du canal Éditeur.

Si cet élément n'apparaît pas, la partie hôte de l'URL est par défaut l'adresse IP du serveur sur lequel s'exécute le pilote Workflow Request Service (c'est-à-dire l'adresse IP du serveur Web du canal Éditeur).

port : Élément facultatif qui se figure sous l'élément <url-data>. S'il est présent, il précise la partie port de l'URL. En général, l'élément « port » est utilisé uniquement si l'URL doit pointer sur un serveur autre que le serveur Web du canal Éditeur.

S'il n'apparaît pas, la partie port de l'URL est par défaut le port sur lequel s'exécute le serveur Web du canal Éditeur.

<url-query>

L'élément <url-query> est un enfant de l'élément <url-data>. Il contient des éléments <item> qui permettent de construire la partie requête de l'URL insérée dans le message électronique.

Chaque élément qui apparaît comme enfant de l'élément <url-query> est placé dans la chaîne de requête sous la forme name="value" où « name » représente la valeur de l'attribut *name* de l'élément <item> et « value » est le contenu de type chaîne de l'élément <item>.

Les éléments « item » qui apparaissent sous <url-query> peuvent inclure l'attribut *protect* défini sur « yes ». Dans ce cas, les noms et valeurs de ces éléments seront codés et placés dans une paire nom-valeur générée automatiquement dans la chaîne de requête d'URL. Le nom de la valeur générée est « protected-data » et sa valeur est la paire (ou les paires dans le cas d'attributs à valeurs multiples) nom-valeur codée au format Base64.

La protection des données garantit qu'elles ne pourront pas être modifiées lors de la soumission de l'URL au serveur Web du canal Éditeur. Par exemple, les éléments de données « responder-dn » doivent être protégés pour garantir que seuls les utilisateurs autorisés à répondre au message électronique pourront modifier les données eDirectory.

Si l'URL générée doit être utilisée avec le serveur Web du canal Éditeur, l'élément <url-query> doit contenir au moins un élément <item name="responder-dn" protect="yes"> sinon le serveur Web rejettera la requête HTTP POST finale.

Annexe B – Éléments de données de remplacement automatiques

Le pilote Workflow Request Service fournit automatiquement certains éléments de données de remplacement. Ils sont décrits dans cette section.

Données de remplacement automatiques du canal Abonné

Les éléments de données suivants sont automatiquement ajoutés aux documents de données de remplacement lors du traitement par le canal Abonné :

association : Un élément <item name=”association”> est ajouté au document de données de remplacement si l'élément <mail> possède un élément enfant <association> ou si le canal Abonné renvoie un élément <add-association>. Le contenu de l'élément <item> est la valeur d'association correspondant à l'objet eDirectory associé au message électronique en cours de traitement. Il est possible que la valeur d'association ne soit pas encore créée dans l'objet eDirectory ; par conséquent, elle ne pourra pas être utilisée dans les requêtes.

url : Le contenu de l'élément <item> est l'URL complète à utiliser dans le message électronique. Sur le canal Abonné, l'élément « url » est créé à partir des éléments « scheme », « host », « port », « file », qui figurent sous l'élément <url-data>, et des éléments qui figurent sous l'élément <url-query>. En l'absence de valeurs pour ces éléments, les valeurs par défaut sont utilisées. Ces valeurs sont déterminées à partir de la configuration du serveur Web du canal Éditeur.

url-base : Le contenu de l'élément <item> est la partie de l'URL générée qui ne contient ni l'identificateur de ressource (file) ni la chaîne de requête (query).

url-query : Le contenu de l'élément <item> est une chaîne de requête d'URL générée à partir des éléments <item> qui figurent sous l'élément <url-query>.

url-file : Le contenu de l'élément <item> est l'identificateur de ressource de l'URL.

protected-data : Le contenu de l'élément <item> est une forme codée des paires « name-value » obtenues à partir d'éléments <item> qui figurent sous l'élément <url-query>. Seuls les éléments <item> dont l'attribut *protect* a pour valeur « yes » sont ajoutés à la valeur de données protégées. Pour plus d'informations sur les données protégées, reportez-vous à l'Annexe A *Sécurité des données*.

Données de remplacement automatiques du canal Éditeur

Les éléments de données suivants sont automatiquement ajoutés aux documents de données de remplacement lors du traitement par le canal Éditeur :

post-status : Un élément <item name="post-status"> est créé et ajouté au document de données de remplacement par le serveur Web du canal Éditeur durant le traitement d'une requête HTTP POST. Une requête HTTP POST au serveur Web est une demande de soumission d'un document XDS à DirXML. DirXML renvoie un document d'état comme résultat de cette soumission. Le contenu de l'élément <item name="post-status"> est la valeur de l'attribut *level* de l'élément <status> renvoyé par DirXML comme résultat de la soumission à DirXML.

L'élément « post-status » est généralement utilisé dans la construction de la page Web renvoyée en réponse à la requête HTTP POST.

post-status-message : Un élément <item name="post-status-message"> est créé et ajouté au document de données de remplacement par le serveur Web du canal Éditeur durant le traitement d'une requête HTTP POST. Une requête HTTP POST au serveur Web est une demande de soumission d'un document XDS à DirXML. DirXML renvoie un document d'état comme résultat de cette soumission. Le contenu de l'élément <item name="post-status-message"> est le contenu de l'élément <status> renvoyé par DirXML en réponse à la soumission à DirXML. L'élément « post-status-message » sera créé uniquement si l'élément <status> renvoyé par DirXML possède un contenu.

L'élément « post-status-message » est généralement utilisé dans la construction de la page Web renvoyée en réponse à la requête HTTP POST.

url : Un élément <item name="url"> est créé et ajouté au document de données de remplacement par le serveur Web du canal Éditeur durant le traitement des requêtes HTTP GET et HTTP POST. L'élément <item> est ajouté avant l'utilisation du document de données de remplacement pour construire un document final. Le protocole, l'hôte et le port (« scheme », « host », « port ») de l'URL sont déterminés par la configuration du serveur Web.

url-base : Un élément <item name="url-base"> est créé et ajouté au document de données de remplacement par le serveur Web du canal Éditeur durant le traitement des requêtes HTTP GET et HTTP POST. L'élément <item> est ajouté avant l'utilisation du document de données de remplacement pour construire un document final. Le contenu de l'élément « url-base <item> » sur le canal Éditeur est identique à celui de l'élément « url <item> ».

Annexe C – Référence aux éléments d'opération contenus dans les modèles

Les éléments d'opération sont des éléments qualifiés par un espace de nom. Ils figurent dans le modèle et sont utilisés pour une commande simple ou pour créer des éléments HTML pour des formulaires HTML. L'espace de nom utilisé pour qualifier les éléments est http://www.novell.com/dirxml/workflow/form. Dans ce document et dans les exemples de modèles fournis avec le pilote Workflow Request Service, le préfixe utilisé est form.

Tout élément d'opération non traité dans cette annexe se voit éliminé du document final par la feuille de style de traitement de modèles (sauf cette la feuille de style est personnalisée). Ce comportement permet par exemple d'inclure dans un élément « form:text » les données destinées à la construction d'un message électronique au format texte ordinaire et de créer ainsi un modèle XML valide.

<form:input>

L'élément « form:input » permet de générer un ou plusieurs éléments HTML INPUT selon qu'un ou plusieurs éléments de données de remplacement sont présents. Le nombre d'éléments INPUT créés correspond au nombre d'éléments de données de remplacement dont le nom est spécifié par l'attribut « name » de l'élément « form:input ».

Attributs

Name : Précise le nom des éléments de données de remplacement utilisés pour créer les éléments INPUT. La valeur d'attribut sert de valeur à l'attribut *name* des éléments INPUT créés.

type ou **TYPE** : Précise la valeur de l'attribut *type* des éléments INPUT créés.

value : Si la valeur de l'attribut *value* est « yes », un attribut *value* est ajouté aux éléments INPUT créés dont la valeur est la valeur de type chaîne de l'élément de données de remplacement. Si la valeur de l'attribut *value* est différente de « yes », le contenu des éléments INPUT créés a pour valeur la valeur de type chaîne de l'élément de données de remplacement.

Exemple

```
<form:input name="responder-dn" TYPE="hidden" value="yes"/>
```

crée un ou plusieurs éléments INPUT similaires à

```
<INPUT name="responder-dn" TYPE="hidden" value="\PERIN-TAO\novell\phb"/>
```

<form:if-item-exists>

L'élément « form:if-item-exists » permet d'insérer conditionnellement des données dans le document final. Son contenu est traité uniquement si l'élément spécifié apparaît parmi les données de remplacement.

Attributs

Name : Précise le nom de l'élément de données de remplacement. Si plusieurs exemples de l'élément de données de remplacement existent, le contenu de l'élément « form:if-item-exists » est traité.

Exemple

```
<form:if-item-exists name="post-status-message">

      <tr>
        <td>
          Le message d'état était : $post-status-message$
        </td>
      </tr>
</form:if-item-exists>
```

L'exemple ci-dessus insère une ligne dans une table HTML uniquement si un élément de données de remplacement « post-status-message » existe.

<form:if-multiple-items>

L'élément « form:if-multiple-items » permet d'insérer conditionnellement des données dans le document final. Son contenu est traité uniquement si l'élément spécifié apparaît plusieurs fois parmi les données de remplacement.

Attributs

name : Précise le nom de l'élément de données de remplacement. Si plusieurs exemples de l'élément de données de remplacement existent, le contenu de l'élément « form:if-multiple-items » est traité.

Exemple

```
<form:if-multiple-items name="responder-dn">
    <form:menu name="responder-dn"/>
</form:if-multiple-items>
```

L'exemple ci-dessus créera un élément HTML SELECT (voir <form:menu>) s'il existe plusieurs éléments de données de remplacement nommés « responder-dn ».

<form:if-single-item>

L'élément « form:if-single-item » permet d'insérer conditionnellement des données dans le document final. Son contenu est traité uniquement si l'élément spécifié n'apparaît qu'une seule fois parmi les données de remplacement.

Attributs

name : Précise le nom de l'élément de données de remplacement. Si l'élément nommé ne figure qu'une seule fois parmi les données de remplacement, le contenu de l'élément « form:if-single-item » est traité.

Exemple

```
<form:if-single-item name="responder-dn">
  <input TYPE="hidden" name="responder-dn" value="$responder-dn$"/>
  $responder-dn$
</form:if-single-item>
```

L'exemple ci-dessus insère un élément HTML INPUT et un texte de remplacement dans le document final s'il existe exactement un seul élément nommé « responder-dn » parmi les données de remplacement.

<form:menu>

L'élément « form:menu » permet de générer un élément HTML SELECT associé à un ou plusieurs éléments OPTION enfant. Le premier élément OPTION enfant sera marqué comme étant sélectionné.

Attributs

name : Précise le nom de l'élément de données de remplacement. Si l'élément nommé figure parmi les données de remplacement, un élément HTML SELECT est créé dans le document final. Un élément HTML OPTION est créé en tant qu'enfant de l'élément SELECT pour chaque instance de l'élément de données qui figure parmi les données de remplacement.

Exemple

```
<form:menu name="responder-dn"/>
```

L'exemple ci-dessus produira des éléments HTML similaires aux suivants :

```
<SELECT name="responder-dn">
    <OPTION selected>\PERIN-TAO\big-org\php</OPTION>
    <OPTION>\PERIN-TAO\big-org\carol</OPTION>
</SELECT>
```

Annexe D – Référence à l'élément <mail>

L'élément <mail> et son contenu sont décrits en détail dans cette annexe. Si aucun attribut n'est listé pour un élément, cela signifie qu'aucun attribut n'a été défini pour cet élément.

<mail>

L'élément <mail> et son contenu décrivent les données nécessaires pour construire un message SMTP.

Attributs de l'élément <mail>

src-dn : Contient la valeur DN de l'objet eDirectory qui déclenche le message électronique. Obligatoire si les données de l'objet doivent être modifiées via le serveur Web du canal Éditeur en réponse au message électronique.

<to>

L'élément <to> est un enfant de l'élément <mail>. Un ou plusieurs éléments <to> contiennent les adresses électroniques des destinataires principaux du message SMTP. Au moins un élément <to> est obligatoire. Chaque élément <to> doit contenir une seule adresse électronique.

<cc>

L'élément <cc> est un enfant de l'élément <mail>. Plusieurs éléments <cc> contiennent les adresses électroniques des destinataires en CC du message SMTP. Aucun élément <cc> n'est obligatoire. Chaque élément <cc> doit contenir une seule adresse électronique.

<bcc>

L'élément <bcc> est un enfant de l'élément <mail>. Plusieurs éléments <bcc> contiennent les adresses électroniques des destinataires en CM du message SMTP. Aucun élément <bcc> n'est obligatoire. Chaque élément <bcc> doit contenir une seule adresse électronique.

<from>

L'élément <from> est un enfant de l'élément <mail>. Il contient l'adresse électronique de l'expéditeur du message électronique. L'élément <from> n'est pas obligatoire. En l'absence de l'élément <from>, l'adresse « De » par défaut fournie dans les paramètres du pilote Workflow Request Service est utilisée.

<reply-to>

L'élément <reply-to> est un enfant de l'élément <mail>. Il contient l'adresse électronique de la personne à laquelle les réponses au message SMTP seront adressées. L'élément <reply-to> n'est pas obligatoire.

<subject>

L'élément <subject> est un enfant de l'élément <mail>. Son contenu de type chaîne permet de définir le champ d'objet du message SMTP. L'élément <subject> n'est pas obligatoire mais recommandé, pour des raisons évidentes.

<message>

L'élément <message> est un enfant de l'élément <mail>. Son contenu permet de construire le corps du message SMTP. Au moins un élément <message> est obligatoire. Plusieurs éléments <message> peuvent être fournis pour la construction d'un message SMTP proposant plusieurs représentations du corps du message (texte ordinaire et HTML, ou anglais et une autre langue).

Attributs de l'élément <message>

mime-type : Précise facultativement le type MIME du corps du message construit par l'élément <message> (par exemple text/plain ou text/html). En l'absence de l'attribut « mime-type », le pilote essaie de découvrir automatiquement le type MIME.

Les clients de messagerie peuvent utiliser le type MIME lorsqu'un message SMTP possède plusieurs représentations, ce qui permet de choisir la meilleure à afficher.

language : Précise facultativement la langue du corps du message construit par l'élément <message>. La valeur doit suivre la spécification SMTP. En l'absence de l'attribut de langue, aucune langue par défaut n'est fournie.

Les clients de messagerie peuvent utiliser la langue spécifiée lorsqu'un message SMTP possède plusieurs représentations, ce qui permet de choisir la meilleure à afficher.

<stylesheet>

L'élément <stylesheet> est un enfant de l'élément <message>. Le contenu de l'élément <stylesheet> est le nom de la feuille de style XSLT qui permet de construire le corps du message. En l'absence de l'élément <stylesheet>, process_template.xsl est utilisé comme feuille de style.

<template>

L'élément <template> est un enfant de l'élément <message>. Le contenu de l'élément <template> est le nom du document XML qui permet de construire le corps du message. En l'absence de l'élément <template>, le document de données de remplacement est traité par la feuille de style du message pour construire le corps du message.

<filename>

L'élément <filename> est un enfant de l'élément <attachment>. Le contenu de l'élément <filename> est un nom de fichier. La valeur de nom de fichier permet d'assigner un nom de fichier à une pièce jointe construite.

<replacement-data>

L'élément <replacement-data> est un enfant de l'élément <message>. Son contenu est utilisé comme paramètre de la feuille de style qui traite le modèle de message ou, en l'absence de modèle, est traité directement par la feuille de style du message. Le contenu de l'élément <replacement-data> est décrit en détail dans les annexes A et B.

<resource>

L'élément <resource> est un enfant de l'élément <message>. Son contenu est utilisé comme nom du fichier à incorporer dans le message SMTP en tant que ressource pour le corps du message. Par exemple, une feuille de style .css correspondant au corps d'un message HTML peut être fournie comme ressource.

Attributs de l'élément <resource>

cid : Précise l'ID de contenu utilisé pour désigner la ressource dans les URL qui figurent dans le corps du message. Par exemple, si une feuille de style .css est la ressource, la valeur cid peut être css-1. Dans le corps du message HTML, l'élément suivant peut servir à désigner la feuille de style .css :

```
<link href="cid:css-1" rel="style sheet" type="text/css">.
```

<attachment>

L'élément <attachment> est un enfant de l'élément <mail>. Il peut avoir le même contenu que <message> ou peut utiliser un nom de fichier comme contenu. Plusieurs éléments <attachment> peuvent apparaître en tant qu'enfants de l'élément <mail>.

Attributs de l'élément <attachment>

mime-type : Précise facultativement le type MIME de la pièce jointe. En l'absence de l'attribut « mime-type », le pilote essaie de découvrir automatiquement le type MIME.

language : Précise facultativement la langue de la pièce jointe. En l'absence de l'attribut de langue, aucune langue par défaut n'est fournie.

Annexe E – Flux de données

Cette section décrit dans le détail le flux de données généré par exemple dans la situation suivante : suite à l'embauche d'un nouvel employé, un message électronique est envoyé à son responsable. Ce message demande au responsable d'utiliser l'URL qu'il contient pour entrer un numéro de bureau pour l'employé.

La configuration du pilote Workflow Request Service pour le scénario de l'exemple est la suivante :

Configuration du canal Abonné

Filtre

Classe : Utilisateur

Attributs :

- Given Name
- manager
- Surname

Règles

Créer une règle :

attributs « Given Name », « manager » et « Surname » obligatoires.

Règle de transformation de commande

Convertit l'élément <add> en élément<mail>.

Configuration du canal Éditeur

Filtre

Classe : Utilisateur

Attributs :

- roomNumber

Règles

Aucune

Description détaillée

Dans la liste suivante, les éléments de données les plus importants qui circulent dans le processus sont « responder-dn » et « association ». L'élément « responder-dn » permet d'authentifier l'utilisateur qui entre les données via le serveur Web. L'élément « association » identifie l'objet eDirectory dont les données doivent être modifiées.

1) La société embauche un nouvel employé. Les données correspondantes sont entrées dans le système de gestions des ressources humaines (HR) de la société.

2) Le pilote DirXML associé au système de gestion ressources humaines crée un nouvel objet Utilisateur dans eDirectory. Les attributs de l'objet Utilisateur incluent Given Name, Surname et manager.

3) L'événement <add> (ajouter) suivant pour le nouvel objet Utilisateur est soumis au canal Abonné du pilote Workflow Request Service :

```
<nds dtdversion="1.1" ndsversion="8.6">
  <input>
    <add class-name="User" src-dn="\PERIN-TAO\novell\Provo\Joe" src-entry-id="281002"
timestamp="1023314433#2">
      <add-attr attr-name="Surname">
        <value type="string">le stagiaire</value>
      <add-attr>
      <add-attr attr-name="Given Name">
        <value type="string">Joe</value>
      <add-attr>
      <add-attr attr-name="manager">
        <value type="dn">\PERIN-TAO\novell\Provo\phb</value>
      <add-attr>
    </add>
  </input>
</nds>
```

   a) La règle de transformation de commande du canal Abonné utilise le DN du responsable pour envoyer à eDirectory la demande de l'adresse électronique du responsable et le DN de son assistant.

   b) Si le responsable a un assistant, la règle de transformation de la commande du canal Abonné envoie à eDirectory la demande de l'adresse électronique de l'assistant.

   c) La règle de transformation de la commande du canal Abonné construit un élément <mail> et remplace l'élément de commande <add> par l'élément <mail>. Les éléments de données de remplacement apparaissent en **gras**.

```
<nds dtdversion="1.1" ndsversion="8.6">
  <input>
    <mail src-dn="\PERIN-TAO\novell\Provo\Joe">
      <to>phb@company.com</to>
      <cc>carol@company.com</cc>
      <bcc>HR@company.com</bcc>
      <reply-to>HR@company.com</reply-to>
      <subject>Assignation de bureau nécessaire pour : Joe le stagiaire</subject>
      <message mime-type="text/html">
        <stylesheet>process_template.xsl</stylesheet>
        <template>html_msg_template.xml</template>
        <replacement-data>
          <item name="manager">JStanley</item>
          <item name="given-name">Joe</item>
          <item name="surname">le stagiaire</item>
          <url-data>
            <item name="file">process_template.xsl</item>
            <url-query>
              <item name="template">form_template.xml</item>
              <item name="responder-dn" protect="yes">\PERIN-TAO\novell\Provo\phb</item>
              <item name="responder-dn" protect="yes">\PERIN-TAO\novell\Provo\carol</item>
              <item name="subject-name">Joe le stagiaire</item>
            </url-query>
          </url-data>
        </replacement-data>
        <resource cid="css-1">novdocmain.css</resource>
      </message>
    </mail>
  </input>
</nds>
```


d) Le canal Abonné du pilote Workflow Request Service reçoit l'élément <mail> de DirXML.

e) Le canal Abonné génère une valeur d'association car l'élément <mail> a un attribut *src-dn*.

f) Le canal Abonné construit un document de données de remplacement à partir des données de l'élément <mail>. Ce document sera utilisé pour construire le message électronique. L'URL comprend divers éléments de données dans la partie requête (partie de l'URL qui suit le caractère ? et se trouve en **gras**). Le serveur Web du canal Éditeur utilise ces éléments de données lorsque l'URL est soumise au serveur Web en tant que requête HTTP GET.

```
<replacement-data>
  <item name="manager">JStanley</item>
  <item name="given-name">Joe</item>
  <item name="surname">le stagiaire</item>
  <item name="template">form_template.xml</item>
  <item name="responder-dn">\PERIN-TAO\novell\Provo\phb</item>
  <item name="responder-dn">\PERIN-TAO\novell\Provo\carol</item>
  <item name="subject-name">Joe le stagiaire</item>
  <item name="association">1671b2:ee4246a561:-7fff:192.168.0.1</item>
  <item name="url-base">https://192.168.0.1:8180</item>
  <item name="url-file">process_template.xsl</item>
  <item name="protected-data">
rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA
1lbmNvZGVkUGFyYW1zdAACW0JbABBlbmNyeXB0ZWRDb250ZW50cQB+AAFMAAlw
YXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWFsQWxncQB+AAJ4cH
VyAAJbQqzzF/gGCFTgAgAAeHAAAAAPMA0ECEIBRohGPjxEAgEKdXEAfgAEAAAA
uMSFqzHXwtMx8DkRCzkK1O46sEz1u51o3MDvHn+3+fE6SphHr3Hgjli4Jp3rUk
H7y6dXvcu7iq21Vs+9o6iZVzljTIJX/jjRrVZlR5JOuRNhk8JHFZ8FhgsmiIAH
/Fs61k4WmyEcmYfWmfqfBVeThr3Avwcim6ranS5Mm2U5i9Z/DBR13pIAobMpWY
kMaz4+G9e6oovBsiPdp6jSPzbFxcgALI2AMBh4hf9jnx7zOU9Uvd9qXtaE2rR0
AANQQkV0ABBQQkVXaXRoTUQ1QW5kREVT</item>
  <item name="url-query">template=form_template.xml&amp;responder-dn=%5CPERIN-
TAO%5Cnovell%5Cprovo%5Cphb&amp;responder-dn=%5CPERIN-
TAO%5Cnovell%5Cprovo%5Ccarol&amp;subject-
name=Joe+the+Intern&amp;association=1671b2%3Aee4246a561%3A-
7fff%3A192.168.0.1&amp;protected-
data=rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA1lbmNvZGVkUGFyYW1zdAAC
W0JbABBlbmNyeXB0ZWRDb250ZW50cQB%2BAAFMAAlwYXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWF
sQWxncQB%2BAAJ4cHVyAAJbQqzzF%2FgGCFTgAgAAeHAAAAAPMA0ECEIBRohGPjxEAgEKdXEAfgAEAAAAuMSFqzHX
wtMx8DkRCzkK1O46sEz1u51o3MDvHn%2B3%2BfE6SphHr3Hgjli4Jp3rUkH7y6dXvcu7iq21Vs%2B9o6iZVzljTIJ
X%2FjjRrVZlR5JouRNhk8JHFZ8FhgsmiIAH%2FFs61k4WmyEcmYfWmfqfBVeThr3Avwcim6ranS5Mm2U5i9Z%2FDB
R13pIAobMpWYkMaz4%2BG9e6oovBsiPdp6jSPzbFxcgALI2AMBh4hf9jnx7zOU9Uvd9qXtaE2rR0AANQQkV0ABBQQ
kVXaXRoTUQ1QW5kREVT</item>
  <item name="url">
https://192.168.0.1:8180/process_template.xsl?template=form_template.xml&amp;responder-
dn=%5CPERIN-TAO%5Cnovell%5CProvo%5Cphb&amp;responder-dn=%5CPERIN-
TAO%5Cnovell%5Cprovo%5Ccarol&amp;subject-
name=Joe+le+stagiaire&amp;association=1671b2%3Aee4246a561%3A-
7fff%3A192.168.0.1&amp;protected-
data=rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA1lbmNvZGVkUGFyYW1zdAAC
W0JbABBlbmNyeXB0ZWRDb250ZW50cQB%2BAAFMAAlwYXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWF
sQWxncQB%2BAAJ4cHVyAAJbQqzzF%2FgGCFTgAgAAeHAAAAAPMA0ECEIBRohGPjxEAgEKdXEAfgAEAAAAuMSFqzHX
wtMx8DkRCzkK1O46sEz1u51o3MDvHn%2B3%2BfE6SphHr3Hgjli4Jp3rUkH7y6dXvcu7iq21Vs%2B9o6iZVzljTIJ
X%2FjjRrVZlR5JouRNhk8JHFZ8FhgsmiIAH%2FFs61k4WmyEcmYfWmfqfBVeThr3Avwcim6ranS5Mm2U5i9Z%2FDB
R13pIAobMpWYkMaz4%2BG9e6oovBsiPdp6jSPzbFxcgALI2AMBh4hf9jnx7zOU9Uvd9qXtaE2rR0AANQQkV0ABBQQ
kVXaXRoTUQ1QW5kREVT</item>
</replacement-data>
```

g) Le canal Abonné traite le document html_msg_template.xml à l'aide de la feuille de style process_template.xsl. Le document de données de remplacement est transmis à la feuille de style sous forme de paramètre. Le document html_msg_template.xml suit. Notez les jetons de remplacement en gras. Ils seront remplacés par la valeur des éléments <item> correspondants contenus dans le document de données de remplacement.

```
<html xmlns:form="http://www.novell.com/dirxml/workflow/form">
  <head>
  </head>
  <body>
    <link href="cid:css-1" rel="style sheet" type="text/css"/>
    <p>
    Cher $manager$,
    </p>
    <p>
    Nous vous informons du recrutement de <b>$given-name$ $surname$</b>.
    </p>
    <p>
    Assignez un numéro de bureau à cette personne. Pour ce faire, cliquez <a href="$url$">
ici</a>.
    </p>
    <p>
    En vous remerciant,<br/>
    RH<br/>
    Service des Ressources Humaines
    </p>
  </body>
</html>
```

Le message électronique généré suit. Notez que les jetons de remplacement ont été remplacés par les valeurs des éléments <item> correspondants contenus dans le document de données de remplacement.

```
<html>
  <head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-8">
  </head>
  <body>
    <link href="cid:css-1" rel="style sheet" type="text/css">
    <p>
    Cher J Stanley,
    </p>
    <p>
    Nous vous informons du recrutement de <b>Joe le stagiaire</b>.
    </p>
    <p>
    Assignez un numéro de bureau à cette personne. Cliquez <a
href="https://192.168.0.1:8180/process_template.xsl?template=form_template.xml&responder-
dn=%5CPERIN-TAO%5Cnovell%5CProvo%5Cphb&responder-dn=%5CPERIN-
TAO%5Cnovell%5CProvo%5Ccarol&subject-
name=Joe+le+stagiaire&association=45f0e3%3Aee45e07709%3A-7fff%3A192.168.0.1&protected-
data=rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA1lbmNvZGVkUGFyYW1zdAAC
W0JbABBlbmNyeXB0ZWRDb250ZW50cQB%2BAAFMAAlwYXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWF
sQWxncQB%2BAAJ4cHVyAAJbQqzzF%2FgGCFTgAgAAeHAAAAAPMA0ECIr9Z1iG%2BO3BAgEKdXEAfgAEAAAAuMU%2F
SoFRkebvh2d5SqalF91ttjRY5lyyW5%2B%2FFIfOuDdYikYiDbOJb6607S0dPHjQzeVgu6ptIvGqaEQOEjBjDkY%2
Bi4VoVjUSXS3a8fiXB8moMdPtLJ%2FGyE8QiwbT4xbkQy48i02k99F2vGmlenRpSP6dD31kZl3dpJ0mGgq2yL%2Fe
FaynKyqnjkHLMexcqD8WlVooaRl1k2RPk5vDYvC8o2bn22OKKbOnSRM5YlPS0iWzxo0JVcnVVyt0AANQQkV0ABBQQ
kVXaXRoTUQ1QW5kREVT">ici</a> pour le faire.
    </p>
    <p>
    En vous remerciant,<br>
    RH<br>
    Service des Ressources Humaines
    </p>
  </body>
</html>
```

h) Le message électronique SMTP est envoyé au responsable et à son assistant.

i) Le canal Abonné renvoie un document XML contenant un élément <status> et un élément <add-association> à DirXML.

4) Le responsable ouvre le message électronique et clique sur le lien « Cliquez ici ».

5) Le navigateur Web du responsable soumet l'URL au serveur Web du canal Éditeur en tant que requête HTTP GET.

   a) Le serveur Web construit le document de données de remplacement suivant. La plupart des éléments de données proviennent de la partie requête de l'URL, à l'exception des éléments « url » et « url-base » générés automatiquement.

```
<replacement-data>
  <item name="association">45f0e3:ee45e07709:-7fff:192.168.0.1</item>
  <item name="protected-
data">rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA1lbmNvZGVkUGFyYW1zdAA
CW0JbABBlbmNyeXB0ZWRDb250ZW50cQB+AAFMAAlwYXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWFs
QWxncQB+AAJ4cHVyAAJbQqzzF/gGCFTgAgAAeHAAAAAPMA0ECIr9Z1iG+O3BAgEKdXEAfgAEAAAAuMU/SoFRkebvh
2d5SqalF91ttjRY5lyyW5+/FifOuDdYikYiDbOJb6607S0dPHjQzeVgu6ptIvGqaEQOEjBjDkY+i4VoVjUSXS3a8f
iXB8moMdPtLJ/GyE8QiwbT4xbkQy48i02k99F2vGmlenRpSP6dD31kZl3dpJ0mGgq2yL/eFaynKyqnjkHLMexcqD8
WlVooaRl1k2RPk5vDYvC8o2bn22OKKbOnSRM5YlPS0iWzxo0JVcnVVyt0AANQQkV0ABBQQkVXaXRoTUQ1QW5kREVT
</item>
  <item name="template">form_template.xml</item>
  <item name="responder-dn">\PERIN-TAO\novell\Provo\phb</item>
  <item name="responder-dn">\PERIN-TAO\novell\Provo\carol</item>
  <item name="subject-name">Joe le stagiaire</item>
  <item name="url-base">https://192.168.0.1:8180</item>
  <item name="url">https://192.168.0.1:8180</item>
</replacement-data>
```

Le serveur Web traite le document form_templates.xml à l'aide de la feuille de style process_template.xsl. Les jetons de remplacement et les éléments d'opération sont en **gras**. Notez que divers éléments de données sont placés dans des éléments INPUT masqués pour permettre leur transfert au serveur Web avec les données HTML POST.

Il existe en outre un jeton de remplacement $query:roomNumber$, qui récupérera la valeur actuelle de l'attribut « roomNumber » de l'employé (le cas échéant).

```
<html  xmlns:form="http://www.novell.com/dirxml/workflow/form">
  <head>
    <title>Entrez le numéro de bureau de $subject-name$</title>
  </head>
  <body>
    <link href="novdocmain.css" rel="style sheet" type="text/css"/>
    <br/><br/><br/><br/>
    <form class="myform" METHOD="POST" ACTION="$url-base$/process_template.xsl">
        <table cellpadding="5" cellspacing="10" border="1" align="center">
          <tr><td>
          <input TYPE="hidden" name="template" value="post_form.xml"/>
          <input TYPE="hidden" name="subject-name" value="$subject-name$"/>
          <input TYPE="hidden" name="association" value="$association$"/>
          <input TYPE="hidden" name="response-style sheet" value="process_template.xsl"/>
          <input TYPE="hidden" name="response-template" value="post_response.xml"/>
          <input TYPE="hidden" name="auth-style sheet" value="process_template.xsl"/>
          <input TYPE="hidden" name="auth-template" value="auth_response.xml"/>
          <input TYPE="hidden" name="protected-data" value="$protected-data$"/>
          <form:if-single-item name="responder-dn">
            Vous êtes :<br/>
            <input TYPE="hidden" name="responder-dn" value="$responder-dn$"/>
            $responder-dn$
          </form:if-single-item>
          <form:if-multiple-items name="responder-dn">
            Indiquez votre identité :<br/>
            <form:menu name="responder-dn"/>
          </form:if-multiple-items>
        </td></tr>
        <tr><td>
          Entrez votre mot de passe : <br/><input name="password" TYPE="password"
SIZE="20" MAXLENGTH="40"/>
        </td></tr>
        <tr><td>
          Entrez le numéro de bureau de $subject-name$ :<br/>
          <input TYPE="text" NAME="room-number" SIZE="20" MAXLENGTH="20"
value="$query:roomNumber$"/>
        </td></tr>
        <tr><td>
          <input TYPE="submit" value="Submit"/> <input TYPE="reset" value="Clear"/>
        </td></tr>
      </table>
    </form>
  </body>
</html>
```

Il en résulte la page HTML suivante :

```
<html>
  <head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Entrez le numéro de bureau pour Joe le stagiaire</title>
  </head>
  <body>
    <link href="novdocmain.css" rel="style sheet" type="text/css">
    <br><br><br><br>
<form class="myform" METHOD="POST"
ACTION="https://192.168.0.1:8180/process_template.xsl">
<table cellpadding="5" cellspacing="10" border="1" align="center">
<tr>
<td>
    <input TYPE="hidden" name="template" value="post_form.xml">
    <input TYPE="hidden" name="subject-name" value="Joe le stagiaire">
    <input TYPE="hidden" name="association" value="45f0e3:ee45e07709:-7fff:192.168.0.1">
    <input TYPE="hidden" name="response-style sheet" value="process_template.xsl">
    <input TYPE="hidden" name="response-template" value="post_response.xml">
    <input TYPE="hidden" name="auth-style sheet" value="process_template.xsl">
    <input TYPE="hidden" name="auth-template" value="auth_response.xml">
    <input TYPE="hidden" name="protected-data"
value="rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA1lbmNvZGVkUGFyYW1zdA
ACW0JbABBlbmNyeXB0ZWRDb250ZW50cQB+AAFMAAlwYXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWF
sQWxncQB+AAJ4cHVyAAJbQqzzF/gGCFTgAgAAeHAAAAAPMA0ECIr9Z1iG+O3BAgEKdXEAfgAEAAAAuMU/SoFRkebv
h2d5SqalF91ttjRY5lyyW5+/FIfOuDdYikYiDbOJb6607S0dPHjQzeVgu6ptIvGqaEQOEjBjDkY+i4VoVjUSXS3a8
fiXB8moMdPtLJ/GyE8QiwbT4xbkQy48i02k99F2vGmlenRpSP6dD31kZl3dpJ0mGgq2yL/eFaynKyqnjkHLMexcqD
8WlVooaRl1k2RPk5vDYvC8o2bn22OKKbOnSRM5YlPS0iWzxo0JVcnVVyt0AANQQkV0ABBQQkVXaXRoTUQ1QW5kREV
T">
    Indiquez votre identité :<br>
    <SELECT name="responder-dn">
        <OPTION selected>\PERIN-TAO\novell\Provo\phb</OPTION>
        <OPTION>\PERIN-TAO\novell\Provo\carol</OPTION>
    </SELECT>
</td>
</tr>
<tr>
<td>
    Entrez votre mot de passe : <br>
    <input name="password" TYPE="password" SIZE="20" MAXLENGTH="40">
</td>
</tr>
<tr>
<td>
    Entrez le numéro de bureau pour Joe le stagiaire :<br>
    <input TYPE="text" NAME="room-number" SIZE="20" MAXLENGTH="20" value="">
</td>
</tr>
<tr>
<td>
    <input TYPE="submit" value="Submit"> <input TYPE="reset" value="Clear">
</td>
</tr>
</table>
</form>
</body>
</html>
```

b) Le responsable sélectionne son DN eDirectory à partir du menu de la page Web, entre son mot de passe ainsi que le numéro de bureau du nouvel employé et clique sur Soumettre.

c) Le navigateur Web soumet une requête HTTP POST au serveur Web.

d) Le serveur Web construit le document de données de remplacement suivant à partir des données POST. Notez les données qui étaient dans les divers éléments <INPUT> masqués. Les données entrées par le responsable dans le formulaire sont en **gras**.

```
<replacement-data>
  <item name="room-number">casier 1234</item>
  <item name="template">post_form.xml</item>
  <item name="response-template">post_response.xml</item>
  <item name="auth-template">auth_response.xml</item>
  <item name="association">45f0e3:ee45e07709:-7fff:192.168.0.1</item>
  <item name="password" is-sensitive="true"><!—content suppressed →</item>
  <item name="protected-
data">rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA1lbmNvZGVkUGFyYW1zdAA
CW0JbABBlbmNyeXB0ZWRDb250ZW50cQB+AAFMAAlwYXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWFs
QWxncQB+AAJ4cHVyAAJbQqzzF/gGCFTgAgAAeHAAAAAPMA0ECIr9Z1iG+O3BAgEKdXEAfgAEAAAAuMU/SoFRkebvh
2d5SqalF91ttjRY5lyyW5+/FifOuDdYikYiDbOJb6607S0dPHjQzeVgu6ptIvGqaEQOEjBjDkY+i4VoVjUSXS3a8f
iXB8moMdPtLJ/GyE8QiwbT4xbkQy48i02k99F2vGmlenRpSP6dD31kZl3dpJ0mGgq2yL/eFaynKyqnjkHLMexcqD8
WlVooaRl1k2RPk5vDYvC8o2bn22OKKbOnSRM5YlPS0iWzxo0JVcnVVyt0AANQQkV0ABBQQkVXaXRoTUQ1QW5kREVT
</item>
  <item name="responder-dn">\PERIN-TAO\novell\Provo\phb</item>
  <item name="auth-style sheet">process_template.xsl</item>
  <item name="response-style sheet">process_template.xsl</item>
  <item name="subject-name">Joe le stagiaire</item>
  <item name="url-base">https://192.168.0.1:8180</item>
  <item name="url">https://192.168.0.1:8180</item>
</replacement-data>
```

e) Le serveur Web vérifie si la valeur de l'élément « responder-dn » correspond à une valeur « responder-dn » contenue dans les données protégées. Si elle ne correspond pas, le serveur Web annule la demande. Si elle correspond, le traitement se poursuit.

f) Le serveur Web soumet une requête XDS <check-object-password> à DirXML sur le canal Éditeur pour authentifier l'utilisateur qui soumet la requête HTTP POST.

```
<nds dtdversion="1.0" ndsversion="8.6">
  <source>
    <product build="20020606_0824" instance="Pilote Workflow Request Service"
version="1.1a">Pilote DirXML Workflow Request Service</product>
    <contact>Novell, Inc.</contact>
  </source>
  <input>
    <check-object-password dest-dn="\PERIN-TAO\novell\Provo\phb" event-id="chkpwd">
      <password><!-- content suppressed --></password>
    </check-object-password>
  </input>
</nds>
```

g) DirXML renvoie <status level=”success”>. Si l'état renvoyé par DirXML est différent de Réussi, les modèles spécifiés par l'élément de données « auth_template » et la feuille de style précisée par l'élément de données « auth_stylesheet » sont utilisés pour construire la page Web qui sera renvoyée en réponse à la requête POST.

h) Le serveur Web traite le modèle post_form.xml à l'aide de la feuille de style process_template.xsl pour générer un document XDS. Les jetons de remplacement sont en **gras**.

```
<nds>
  <input>
    <modify class-name="User" src-dn="not-applicable" event-id=”wfmod”>
      <association>$association$</association>
      <modify-attr attr-name="roomNumber">
        <remove-all-values/>
        <add-value>
          <value>$room-number$</value>
        </add-value>
      </modify-attr>
    </modify>
  </input>
</nds>
```

i) Le canal Éditeur soumet le document XDS créé à DirXML.

```
<nds>
  <input>
    <modify class-name="User" src-dn="not-applicable" event-id=”wfmod”>
      <association>45f0e3:ee45e07709:-7fff:192.168.0.1</association>
      <modify-attr attr-name="roomNumber">
        <remove-all-values/>
        <add-value>
          <value>casier 1234</value>
        </add-value>
      </modify-attr>
    </modify>
  </input>
</nds>
```

j) DirXML renvoie un document de résultat.

```
<nds dtdversion="1.1" ndsversion="8.6">
  <source>
    <product version="1.1a">DirXML</product>
    <contact>Novell, Inc.</contact>
  </source>
  <output>
    <status event-id="wfmod" level="success"></status>
  </output>
</nds>
```

k) Le serveur Web ajoute l'élément de données de remplacement « post-status » (et éventuellement l'élément de données de remplacement « post-status-message ») au document de données de remplacement. L'élément de données ajouté apparaît en **gras** :

```
<replacement-data>
  <item name="room-number">casier 1234</item>
  <item name="template">post_form.xml</item>
  <item name="response-template">post_response.xml</item>
  <item name="auth-template">auth_response.xml</item>
  <item name="association">45f0e3:ee45e07709:-7fff:192.168.0.1</item>
  <item name="password" is-sensitive=”true”><!—content suppressed →</item>
  <item name="protected-
data">rO0ABXNyABlqYXZheC5jcnlwdG8uU2VhbGVkT2JqZWN0PjY9psO3VHACAARbAA1lbmNvZGVkUGFyYW1zdAA
CW0JbABBlbmNyeXB0ZWRDb250ZW50cQB+AAFMAAlwYXJhbXNBbGd0ABJMamF2YS9sYW5nL1N0cmluZztMAAdzZWFs
QWxncQB+AAJ4cHVyAAJbQqzzF/gGCFTgAgAAeHAAAAAPMA0ECIr9Z1iG+O3BAgEKdXEAfgAEAAAAuMU/SoFRkebvh
2d5SqalF91ttjRY5lyyW5+/FifOuDdYikYiDbOJb6607S0dPHjQzeVgu6ptIvGqaEQOEjBjDkY+i4VoVjUSXS3a8f
iXB8moMdPtLJ/GyE8QiwbT4xbkQy48i02k99F2vGmlenRpSP6dD31kZl3dpJ0mGgq2yL/eFaynKyqnjkHLMexcqD8
WlVooaRl1k2RPk5vDYvC8o2bn22OKKbOnSRM5YlPS0iWzxo0JVcnVVyt0AANQQkV0ABBQQkVXaXRoTUQ1QW5kREVT
</item>
  <item name="responder-dn">\PERIN-TAO\novell\Provo\phb</item>
  <item name="auth-style sheet">process_template.xsl</item>
  <item name="response-style sheet">process_template.xsl</item>
  <item name="subject-name">Joe le stagiaire</item>
  <item name="url-base">https://192.168.0.1:8180</item>
  <item name="url">https://192.168.0.1:8180</item>
  <status event-id="" level="success"></status>
  <item name="post-status">success</item>
</replacement-data>
```

l) Le serveur Web traite le modèle post_response.xml à l'aide de la feuille de style process_template.xsl. Les jetons de remplacement et les éléments d'opération sont en **gras**.

```
<htm xmlns:form="http://www.novell.com/dirxml/workflow/form">
  <head>
    <title>Résultat de l'opération post pour $subject-name$</title>
  </head>
  <body>
    <link href="novdocmain.css" rel="style sheet" type="text/css"/>
    <br/><br/><br/><br/>
    <table class="formtable" cellpadding="5" cellspacing="20" border="1" align="center">
      <tr>
        <td>
          DirXML a signalé l'état = $post-status$
        </td>
      </tr>
<form:if-item-exists name="post-status-message">
      <tr>
        <td>
          Le message d'état était : $post-status-message$
        </td>
      </tr>
</form:if-item-exists>
    </table>
  </body>
</html>
```

m) La page Web générée est renvoyée en réponse à la requête HTTP POST. Notez que la deuxième ligne de la table est absente car l'élément « post-status-message » désigné par l'élément <form:if-item-exists> ne figure pas dans le document de données de remplacement.

```
<html>
  <head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Résultat de l'opération post pour Joe le stagiaire</title>
  </head>
  <body>
    <link href="novdocmain.css" rel="style sheet" type="text/css">
    <br><br><br><br>
    <table class="formtable" cellpadding="5" cellspacing="20" border="1" align="center">
      <tr>
        <td>
          DirXML a signalé l'état = réussi
        </td>
      </tr>
    </table>
  </body>
</html>
```

Annexe F – Gestionnaires d'éléments personnalisés pour le canal Abonné

Le pilote Workflow propose un mécanisme d'extension pour l'envoi de notifications utilisateur par des méthodes autres que SMTP *(Simplified Mail Transport Protocol)*. Par exemple, un client pourrait avoir besoin d'envoyer des notifications via l'interface MAPI *(Messaging Application Programming Interface)* au lieu de SMTP.

Pour utiliser un autre mécanisme que SMTP pour l'envoi de notifications, vous devez écrire une classe Java qui traite un élément XML personnalisé soumis sur le canal Abonné du pilote Workflow.

Le gestionnaire d'éléments personnalisés Java doit mettre en œuvre l'interface Java com.novell.nds.dirxml.driver.workflow.CommandHandler. Le nom de la classe d'éléments personnalisés est précisé sous Gestionnaires supplémentaires dans la page des paramètres de configuration du canal Abonné.

Lorsque le canal Abonné du pilote Workflow rencontre un élément de commande, il consulte sa table de gestionnaires. Lorsqu'il trouve un gestionnaire qui traite l'élément de commande, cet élément est transmis au gestionnaire. Le gestionnaire exécute ensuite le traitement demandé.

Il existe deux gestionnaires intégrés d'éléments de commande dans le pilote Workflow : un gestionnaire pour les éléments <mail> et un autre pour les éléments <add>.

La définition de l'élément de commande personnalisé incombe à l'auteur du gestionnaire personnalisé. Pour concevoir l'élément de commande personnalisé, il est recommandé de partir de la conception de l'élément <mail>.

Les éléments personnalisés sont créés par des règles sur le canal Abonné, tout comme l'élément <mail>.

La documentation correspondant à com.novell.nds.dirxml.driver.workflow.CommandHandler et la documentation correspondant à de nombreuses classes d'utilitaires et de supports se trouvent dans la documentation Java qui accompagne le pilote Workflow. La documentation Java (javadocs) se trouve dans le fichier workflow_driver_docs.zip dans l'image de distribution.

Construction d'URL utilisables avec le serveur Web du canal Éditeur

Pour utiliser sans problème le serveur Web du canal Éditeur du pilote Workflow, utilisez des classes d'utilitaires pour construire l'URL à inclure au message de notification. La classe com.novell.nds.dirxml.driver.workflow.URLData exécute cette tâche.

L'exemple de code contenu dans SampleCommandHandler.java illustre ce processus.

Construction de documents de messages à l'aide de feuilles de style et de documents de modèles

Il est pratique d'utiliser la même méthode de construction des documents que le gestionnaire SMTP, à savoir, une combinaison de feuilles de style, documents de modèles et données de remplacement. Pour ce faire, vous devez obtenir les feuilles de style et documents de modèles et appeler par programmation le processeur de feuille de style.

L'exemple de code contenu dans SampleCommandHandler.java illustre ce processus.

SampleCommandHandler.java

Le code source de l'exemple de gestionnaire de commande personnalisé est fourni avec le pilote Workflow. Il se trouve dans le fichier workflow_driver_docs.zip dans l'image de distribution.

Le gestionnaire est mis en œuvre dans la classe com.novell.nds.dirxml.driver.workflow.samples.SampleCommandHandler.

L'exemple de gestionnaire génère simplement un document à l'aide de feuilles de style et de modèles et écrit le document résultant dans un fichier.

Compilation de la classe SampleCommandHandler

Pour compiler la classe SampleCommandHandler, vous pouvez utiliser tout compilateur Java 2. Vous devez placer nxsl.jar, dirxml.jar, collections.jar et WorkflowRequestServiceBase.jar dans le chemin de classe du compilateur Java.

Test de la classe SampleCommandHandler

Commencez par importer l'exemple de configuration Room Number correspondant au pilote Workflow.

Compilez la classe SampleCommandHandler et placez le fichier de classe résultant dans un fichier .jar. Placez le fichier .jar dans le répertoire de fichiers DirXML .jar approprié à la plate-forme sur laquelle vous exécutez le pilote.

Ajoutez l'élément XML suivant sous l'élément <subscriber-options> situé à la section Paramètres XML du pilote dans la page des propriétés du pilote :

```
<output-path display-name=”Sample Output Path”></output-path>
```

Éditez les paramètres du pilote. Sous Exemple de chemin de sortie, placez un chemin d'accès à un répertoire dans lequel SampleCommandHandler écrira les documents créés. Sous Gestionnaires supplémentaires, ajoutez la chaîne com.novell.nds.dirxml.driver.workflow.samples.SampleCommandHandler.

Remplacez la règle de transformation de commande du canal Abonné par CommandXform.xsl, qui peut se trouver dans le même répertoire que le fichier SampleCommandHandler.java.

Créez un objet Utilisateur et ajoutez une référence au responsable à l'objet Utilisateur. Si le responsable possède une valeur d'adresse électronique, un élément de commande <sample> est envoyé au canal Abonné et SampleCommandHandler écrit un fichier à l'emplacement spécifié ci-dessus.

Annexe G – Servlets personnalisées pour le canal Éditeur

Le pilote Workflow propose un mécanisme d'extension par lequel vous pouvez ajouter des fonctionnalités au serveur Web du canal Éditeur. Les servlets personnalisées peuvent être chargées par le canal Éditeur. Pour cela, il suffit de préciser le nom de la ou des classes de servlets sous *Servlets supplémentaires* dans la configuration du pilote.

Utilisation du canal Éditeur

Si une servlet personnalisée doit soumettre des données à DirXML, elle devra utiliser le canal Éditeur du pilote Workflow. Les classes com.novell.nds.dirxml.driver.workflow.ServletRegistrar et com.novell.nds.dirxml.driver.workflow.PublisherData sont fournies pour faciliter cette opération. L'exemple de code contenu dans SampleServlet.java illustre ce processus.

Authentification

Une servlet personnalisée doit authentifier les utilisateurs qui soumettent des informations. L'exemple de code contenu dans SampleServlet.java illustre ce processus. Notez toutefois que le type d'authentification effectuée à l'aide de l'élément <check-object-password> ne vérifie pas les droits NDS. Toute modification soumise sur le canal Éditeur est autorisée si l'objet Pilote dispose de tels droits, que l'utilisateur ait ou non le droit d'apporter des modifications.

Si vous utilisez une URL générée par un gestionnaire de commande sur le canal Abonné, vous devez utiliser la classe com.novell.nds.dirxml.driver.workflow.URLData pour valider l'URL afin de garantir que l'élément de données « responder-dn » n'a pas été modifié. Pour des informations à ce sujet, reportez-vous à la documentation Java (javadocs).

SampleServlet.java

Le code source d'un exemple de servlet est fourni avec le pilote Workflow. Il se trouve dans le fichier workflow_driver_docs.zip dans l'image de distribution.

La servlet est implémentée dans la classe com.novell.nds.dirxml.driver.workflow.samples.SampleServlet.

Cette servlet accepte une requête HTTP GET pour toute ressource finissant par *.sample.* La chaîne de requête de l'URL HTTP doit contenir une donnée dest-dn, une donnée attr-name et une donnée value.

La servlet authentifie l'utilisateur, puis soumet la requête de modification à DirXML via le canal Éditeur du pilote Workflow.

Compilation de la classe SampleServlet

Pour compiler la classe SampleServlet, vous pouvez utiliser tout compilateur Java 2. Vous devez placer nxsl.jar, dirxml.jar, collections.jar et WorkflowRequestServiceBase.jar dans le chemin de classe du compilateur Java.

Test de la classe SampleServlet

Commencez par importer l'exemple de configuration Room Number correspondant au pilote Workflow.

Compilez la classe SampleServlet et placez le fichier de classe résultant dans un fichier .jar. Placez le fichier .jar dans le répertoire de fichiers DirXML .jar approprié à la plate-forme sur laquelle vous exécutez le pilote.

Éditez les paramètres du pilote. Sous *Servlets supplémentaires,* ajoutez la chaîne com.novell.nds.dirxml.driver.workflow.samples.SampleServlet.

Ajoutez le Numéro de téléphone au filtre du canal Éditeur.

Soumettez l'URL suivante dans un navigateur (en supposant que le navigateur s'exécute sur la même machine que le pilote Workflow) :

https:localhost:8180/1.sample?dest-dn=username.container&attr-name=Telephone%20Number&value=555-1212

Remplacez *username.container* par le DN d'un utilisateur de votre arborescence.