Novell Identity Manager

3.5.1

September 28, 2007

www.novell.com

IDENTITY MANAGER ユーザアプリケーション：管理ガイド

Novell®

保証と著作権

米国 Novell, Inc. およびノベル株式会社は、この文書の内容または使用について、いかなる保証、表明または約束も行っていません。また文書の商品性、および特定の目的への適合性について、いかなる黙示の保証も否認し、排除します。また、本書の内容は予告なく変更されることがあります。

米国 Novell, Inc. およびノベル株式会社は、すべてのノベル製ソフトウェアについて、いかなる保証、表明または約束も行っていません。またノベル製ソフトウェアの商品性、および特定の目的への適合性について、いかなる黙示の保証も否認し、排除します。米国 Novell, Inc. およびノベル株式会社は、ノベル製ソフトウェアの内容を変更する権利を常に留保します。

本契約の下で提供される製品または技術情報はすべて、米国の輸出規制および他国の商法の制限を受けます。お客様は、すべての輸出規制を遵守し、製品の輸出、再輸出、または輸入に必要なすべての許可または等級を取得するものとします。お客様は、現在の米国の輸出除外リストに掲載されている企業、および米国の輸出管理規定で指定された輸出禁止国またはテロリスト国に本製品を輸出または再輸出しないものとします。お客様は、取引対象製品を、禁止されている核兵器、ミサイル、または生物化学兵器を最終目的として使用しないものとします。ノベル製ソフトウェアの輸出については、「Novell International Trade Services (http://www.novell.com/company/policies/trade_services/)」の Web ページをご参照ください。弊社は、お客様が必要な輸出承認を取得しなかったことに対し如何なる責任も負わないものとします。

Copyright © 1997-2007 Novell, Inc. All rights reserved. 本ドキュメントの一部または全体を無断で複写・転載することは、その形態を問わず禁じます。

米国 Novell, Inc. は、本文書に記載されている製品に統合されている技術に関する知的所有権を保有します。これらの知的所有権は、http://www.novell.com/company/legal/patents/ (http://www.novell.com/company/legal/patents/) に記載されている 1 つ以上の米国特許、および米国およびその他の国における 1 つ以上の追加特許または出願中の特許を含む場合があります。

本ソフトウェアとそのドキュメントに対する権利、特許、著作権、およびそれに対して適用可能なその他すべての財産権は、あらゆる場合において、単独でおよび独占的に Novell とそのライセンス許諾者に留まるものであり、ユーザはこのような権利に矛盾する行為を一切取らないものとします。本ソフトウェアは著作権法および国際条約の条項によって保護されています。ユーザは、本ソフトウェアまたはそのドキュメントから著作権表示またはその他の登録商標権の表示を取り除かないものとし、本ソフトウェアまたはそのドキュメントのコピーあるいは抽出物すべての当該の表示を複製する必要があります。ユーザは \'96\'7b\'83\'5c フトウェアの所有権を取得することにはなりません。

Novell, Inc.
404 Wyman Street, Suite 500
Waltham, MA 02451
U.S.A.
www.novell.com

オンラインマニュアル: 本製品とその他の Novell 製品のオンラインヘルプにアクセスする場合や、アップデート版を入手する場合は、www.novell.com/documentation (http://www.novell.com/documentation) をご覧ください。

Novell の商標

Novell の商標については、商標とサービスマークの一覧 (http://www.novell.com/company/legal/trademarks/tmlist.html) を参照してください。

サードパーティ資料

サードパーティの商標は、それぞれの所有者に属します。

サードパーティの保証と著作権

Apache ソフトウェアライセンス、バージョン 1.1

Copyright (c) 2000 The Apache Software Foundation. All rights reserved.

ソースおよびバイナリの形式における再配布および使用は、変更の有無にかかわらず、次の条件を満たした場合に許可されます。

1.\'83\'5cースコードの再配布は、上記の著作権 \'95\'5c 示、諸条件の \'96\'7b リスト、および次の免責事項を保持する必要があります。

2. バイナリ形式での再配布では、配布物に付属するドキュメントおよびその他の資料に、上記の著作権表示、ここに示す条件、および下記の保証の否認が複写される必要があります。

3. 再配布にエンドユーザマニュアルが含まれている場合は、次の謝辞を入れる必要があります："This product includes software developed by the Apache Software Foundation (http://www.apache.org/)."(本製品には、Apache Software Foundation(http://www.apache.org/) が開発したソフトウェアが含まれています)。

本謝辞はソフトウェア自身に表示することもでき、通常サードパーティの謝辞が表示される場所であればどこにでも表示できます。

4. . 書面による事前の許可なしに、「Apache」および「Apache Software Foundation」という名称を、本ソフトウェアから派生した製品の保証または販売促進のために使用してはなりません。書面による許可については、apache@apache.org にお問い合わせください。

5. Apache Software Foundation の書面による事前の許可なしに、\'96\'7b 製品から派生した製品を「Apache」と呼んだり、製品名に「Apache」と記載したりすることはできません。

本ソフトウェアは「現状のまま」提供されるものであり、販売可能性に関する保証の黙示的保証を含む明示的または黙示的保証、および特定の用途に対する適合性はすべて放棄されます。いかなる場合においても、APACHE SOFTWARE FOUNDATION またはその貢献者は、直接的、間接的、付随的、特殊的、例示的、または結果的な損害 (代替商品またはサービスの調達、使用不 \'94\'5c、データの紛失、または利益の逸失、あるいは事業の中断を含むが、これらに限定されない) に対して、契約行為、厳格責任、不法行為 (不注意または別の方法を含む) を含め、責任の理論にかかわらず、たとえかかる損害の発生の可 \'94\'5c 性を知らされていた場合であっても、一切責任を負いません。

Autonomy

Copyright ©1996-2000 Autonomy, Inc.

Bouncy Castle

License Copyright (c) 2000 - 2004 The Legion Of The Bouncy Castle (http://www.bouncycastle.org)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions（ここで付与される許可は、本ソフトウェアと関連するドキュメントファイル (「ソフトウェア」) を取得した人物に対して無料で与えられるもので、ソフトウェアの使用、コピー、変更、結合、公開、配布、サブライセンス、コピーの販売も含めた制限のないソフトウェアの取り扱いが可能です。また、ソフトウェアを提供した人物に対して許可を与えることもできます。ただし、以上の許可は、次のじょう件を満たしている場合にのみ与えられます。）：

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.(上記の著作権情報とこの許可を与える際の条件をすべてのコピー、またはソフトウェアの大半に入れる必要があります。)

本ソフトウェアは、明示的または黙示的を問わず、販売可能性に関する保証、特定の用途に対する適合性、および権利侵害を含むがこれらに限定されないいかなる保証もなしに、「現状のまま」提供される

ものです。いかなる場合においても、著者または著作権保持者は、主張、損害、またはその他の責任に対し、\'96\'7b\'83\'5c フトウェアの使用または \'96\'7b\'83\'5c フトウェアとの関連、あるいは \'96\'7b\'83\'5c フトウェアを他の方法で扱ったことから生じる契約の訴訟、不法行為、またはその他においても、一切責任を負いません。

Castor Library

オリジナルのライセンスについては、http://www.castor.org/license.html を参照してください。

本プロジェクトのコードは、BSD と同様のライセンス [license.txt] にて提供されています :

Copyright 1999-2004 (C) Intalio Inc., and others. All Rights Reserved.

このソフトウェアおよび関連ドキュメント（「本ソフトウェア」）の再配布および使用は、変更の有無にかかわらず、次の条件を満たした場合に許可されます。

1.. ソフトウェアコードの再配布では、著作権の記述および表示を保持する必要があります。再配布においても、\'96\'7b ドキュメントのコピーを含める必要があります。

2. バイナリ形式での再配布では、配布物に付属するドキュメントおよびその他の資料に、上記の著作権表示、ここに示す条件、および下記の保証の否認が複写される必要があります。

3. . Intalio Inc. の書面による事前の許可なしに、「ExoLab」という名称を、本ソフトウェアから派生した製品の保証または販売促進のために使用してはなりません。書面による許可については、info@commat;exolab.org までお問い合わせください。

Intalio Inc. の書面による事前の許可なしに、本ソフトウェアから派生した製品を「Castor」と呼んだり、ソフトウェアの名称に「Castor」と記載したりすることはできません。Exolab、Castor、および Intalio は Intalio Inc. の商標です。

ExoLab プロジェクトに対する当然の賞賛は、(http://www.exolab.org/) を参照してください。

本ソフトウェアは INTALIO および貢献者によって「現状のまま」提供されるものであり、販売可能性に関する保証の黙示的保証を含む明示的または黙示的保証、および特定の用途に対する適合性はすべて放棄されます。いかなる場合においても、INTALIO またはその貢献者は、直接的、間接的、付随的、特殊的、例示的、または結果的な損害 (代替商品またはサービスの調達、使用不 \'94\'5c、データの紛失、または利益の逸失、あるいは事業の中断を含むが、これらに限定されない) に対して、契約行為、厳格責任、不法行為 (不注意または別の方法を含む) を含め、責任の理論にかかわらず、たとえかかる損害の発生の可 \'94\'5c 性を知らされていた場合であっても、一切責任を負いません。

Indiana University Extreme! Lab Software License

バージョン 1.1.1

Copyright (c) 2002 Extreme! Lab, Indiana University. All rights reserved.

ソースおよびバイナリの形式における再配布および使用は、変更の有無にかかわらず、次の条件を満たした場合に許可されます。

1.\'83\'5cースコードの再配布は、上記の著作権 \'95\'5c 示、諸条件の \'96\'7b リスト、および次の免責事項を保持する必要があります。

2. バイナリ形式での再配布では、配布物に付属するドキュメントおよびその他の資料に、上記の著作権表示、ここに示す条件、および下記の保証の否認が複写される必要があります。

3. 再配布にエンドユーザマニュアルが含まれている場合は、次の謝辞を入れる必要があります："This product includes software developed by the Indiana University Extreme!Lab (http://www.extreme.indiana.edu/)."(本製品には、Indiana University Extreme! Lab (http://www.extreme.indiana.edu/) が開発したソフトウェアが含まれています)。

本謝辞はソフトウェア自身に表示することもでき、通常サードパーティの謝辞が表示される場所であればどこにでも表示できます。

4. 書面による事前の許可なしに、「Indiana University」および「Indiana University Extreme! Lab」という名称を、本ソフトウェアから派生した製品の保証または販売促進のために使用してはなりません。書面による許可については、http://www.extreme.indiana.edu/ までお問い合わせください。

5. Indiana University の書面による事前の許可なしに、\'96\'7b 製品から派生した製品で「Indiana

University」という名称を使用したり、製品名に「Indiana University」と記載したりすることはできません。

このソフトウェアは、「現状のまま」提供されます。商品性、特定目的への適合性の保証を含む、明示または暗黙によるいかなる保証も行われません。いかなる場合においても、作者またはその貢献者は、直接的、間接的、付随的、特殊的、例示的、または結果的な損害 (代替商品またはサービスの調達、使用不 \'94\'5c、データの紛失、または利益の逸失、あるいは事業の中断を含むが、これらに限定されない) に対して、契約行為、厳格責任、不法行為 (不注意または別の方法を含む) を含め、責任の理論にかかわらず、たとえかかる損害の発生の可 \'94\'5c 性を知らされていた場合であっても、一切責任を負いません。

JDOM.JAR

Copyright (C) 2000-2002 Brett McLaughlin & Jason Hunter. All rights reserved.

ソースおよびバイナリの形式における再配布および使用は、変更の有無にかかわらず、次の条件を満たした場合に許可されます。

1.\'83\'5cースコードの再配布は、上記の著作権 \'95\'5c 示、諸条件の \'96\'7b リスト、および次の免責事項を保持する必要があります。

2. バイナリ形式での再配布では、配布物に付属するドキュメントおよびその他の資料に、上記の著作権表示、ここに示す条件、および下記の保証の否認が複写される必要があります。

3. 「JDOM」という名称を、本ソフトウェアから派生した製品の保証または販売促進のために使用してはなりません。書面による許可については、license@jdom.org にお問い合わせください。

4. JDOM Project Management (pm@jdom.org) の書面による事前の許可なしに、本製品から派生した製品を「JDOM」と呼んだり、製品名に「JDOM」と記載したりすることはできません。

また、再配布するエンドユーザ向けマニュアルやソフトウェア自体に、次のような意味の文章を入れていただくことを、必須ではありませんがお願いいたします :「本製品には、JDOM プロジェクト (http://www.jdom.org/) により開発されたソフトウェアが含まれています。」

テキストで告知する代わりに、ロゴを使用することもできます (http://www.jdom.org/images/logos)。

本ソフトウェアは「現状のまま」提供されるものであり、販売可能性に関する保証の黙示的保証を含む明示的または黙示的保証、および特定の用途に対する適合性はすべて放棄されます。いかなる場合においても、JDOM の作者またはプロジェクトへの貢献者は、直接的、間接的、付随的、特殊的、例示的、または結果的な損害 (代替商品またはサービスの調達、使用不可、データの紛失、または利益の逸失、あるいは事業の中断を含むが、これらに限定されない) に対して、契約行為、厳格責任、不法行為 (不注意または別の方法を含む) を含め、責任の理論にかかわらず、たとえかかる損害の発生の可能性を知らされていた場合であっても、一切責任を負いません。

Phaos

本ソフトウェアは、部分的に SSLavaTM Toolkit (Copyright ©1996-1998 by Phaos Technology Corporation) から派生しています。All Rights Reserved. 顧客が Phaos\'83\'5c フトウェアの機 \'94\'5c にアクセスすることは禁じられています。

W3C

W3C® ソフトウェア注意事項とライセンス

本作業物 (README、および他の関連資料などのソフトウェアマニュアル類も含む) は、次のライセンスに基づいて著作権所有者から提供されています。本作業物の入手、使用またはコピー、あるいはその両方を行うことにより、ユーザ (使用権者) は、次の条件を読んで理解し、それらに従うことに同意します。

このソフトウェアとそのドキュメントのコピー、変更、および配布の許可は、変更の有無にかかわらず、いかなる目的でも無償で与えられます。この場合、ソフトウェアおよびドキュメント、あるいはその一部 (変更を含む) のすべてのコピーに、以下の内容を記載するものとします。

1. 再配布された作業物または派生作業物のユーザに見える場所に、この \'95\'5c 示の全文。

2. 既存の知的財産権の免責事項、通知、または条件すべて。これらがまったく存在しない場合、再配布または派生したコードの \'96\'7b 文内に、W3C の Software Short Notice を含める必要があります (ハイ

パーテキストを推奨、テキストも可)。

3. . 変更が行われた日付を含む、ファイルに対する変更または改変内容の表示 (コードが派生している場所への URI を提供することをお勧めします)。

本ソフトウェアおよびドキュメントは「現状のまま」提供されるもので、著作権保持者は、販売可能性に関する保証または特定の用途に対する適合性、あるいは本ソフトウェアまたはドキュメントの使用によって、サードパーティの特許、著作権、商標などの権利を侵害しないことを含むが、これらに限定されない表示または保証を、明示的または黙示的を問わず、一切行いません。

著作権保持者は、本ソフトウェアまたはドキュメントの使用から生じる直接的、間接的、特殊、または結果的な損害に対して一切責任を負いません。

具体的な書面による事前の許可なしに、著作権保持者の名称および商標を、本ソフトウェアに関する広告または広報に使用することはできません。\'96\'7b\'83\'5c フトウェアおよび関連ドキュメントの著作権に対する権利は、常に著作権保持者に帰します。

目次

このガイドについて

本ガイドでは、Novell Identity Manager ユーザアプリケーションの管理方法について説明します。このガイドは、次のパートに分かれています。

- 11 ページのパート I「概要」
- 29 ページのパート II「ユーザアプリケーション環境の設定」
- 79 ページのパート III「ユーザアプリケーションの管理」
- 207 ページのパート IV「ポートレットリファレンス」
- 283 ページのパート V「プロビジョニングワークフローの環境設定と管理」
- 499 ページのパート VII「付録」

Identity Manager に含まれる他の機能の管理 (すべてのパッケージに共通) については、『*Novell Identity Manager. 管理ガイド*』を参照してください。

対象読者

本ガイドは、Identity Manager ユーザアプリケーションの Identity セルフサービス機能、またはワークフローベースのプロビジョニング機能の設定、展開、および管理を担当するシステム管理者、設計者、およびコンサルタントを対象としています。

これらの機能についてのエンドユーザ用のドキュメントは、『*Identity Manager ユーザアプリケーション. ユーザガイド*』を参照してください。

フィードバック

本マニュアルおよびこの製品に含まれているその他のマニュアルについて、皆様のご意見やご要望をお寄せください。オンラインマニュアルの各ページの下部にあるユーザコメント機能を使用するか www.novell.com/documentation/feedback.html にアクセスしてコメントを記入してください。

マニュアルの更新

『*Identity Manager ユーザアプリケーション: 管理ガイド*』の最新バージョンについては、Identity Manager のドキュメント用 Web サイト (http://www.novell.com/documentation/idm35) を参照してください。

マニュアルの表記規則

Novell のマニュアルでは、「より大きい」記号 (>) を使用して手順内の操作と相互参照パス内の項目の順序を示します。

商標記号 (®、™ など) は、Novell の商標を示します。アスタリスク (*) は、サードパーティの商標を示します。

パス名の表記に円記号 (\) を使用するプラットフォームとスラッシュ (/) を使用するプラットフォームがありますが、このマニュアルでは円記号を使用します。Linux、UNIX など、スラッシュを使う必要があるプラットフォームを使用しているユーザは、必要に応じてスラッシュを使用してください。

概要

この節では、Identity Manager ユーザアプリケーションについて説明し、組織における ユーザアプリケーションの利用についてその計画立案を支援します。

- 13 ページの第 1 章「ユーザアプリケーションの紹介」

ユーザアプリケーションの紹介 1

この節は、Identity Manager ユーザアプリケーションの概要を説明しています。主なトピックは次のとおりです。

1.1 ユーザアプリケーションについて

Identity Manager ユーザアプリケーションは、Identity Manager の情報、リソース、機能を利用するための、ビジネスユーザ向けのビューです。ユーザアプリケーションはブラウザベースのアプリケーションで、さまざまなセルフサービスタスクを実行することができます。また、ユーザアプリケーションをプロビジョニングモジュールおよび Novell Audit® と連携させれば、プロビジョニング要求と承認を開始、管理できる、完全な終端間ソリューションになります。Identity Manager ユーザアプリケーションは安全でスケーラブルなだけでなく、管理も簡単に行えます。

ユーザアプリケーションを利用すれば、次のようなビジネスニーズに対処できます。

- ユーザへのセルフサービスの提供、新規ユーザの自己登録、匿名／ゲストユーザのアクセスを実現します。

  ユーザアプリケーションには、従業員の ID 情報を管理するための、一連のポートレットが用意されています。これらのポートレットをそのまま使用したり、カスタマイズして次のような ID 管理サービスを提供することができます。

  - ディレクトリオブジェクトの作成、またはワークフローを使ったオブジェクトの作成。
  - ID データの検索。
  - ユーザプロファイルと属性の表示、変更。

  詳細については、207 ページのパート IV「ポートレットリファレンス」を参照してください。
- 組織のポリシーに適合した社内リソースへのアクセスと、企業のセキュリティポリシーコンテキスト内でのプロビジョニングが保証されます。

  企業のセキュリティポリシーのガイドラインに従って、ID データへのユーザアクセスを許可することができます。

  詳細については、33 ページのセクション 2.2「セキュリティ」を参照してください。
- 社内のすべてのシステムに渡って、ユーザ情報の入力、更新、削除などの管理作業の手間を減らせます。

  カスタマイズしたワークフローを作成して配布した ID データを操作したり、必要に応じてワークフローを開始する条件を設定する Web ベースのインタフェースをユーザに提供できます。

詳細については、283 ページのパート V「プロビジョニングワークフローの環境設定と管理」を参照してください。

- ID、サービス、リソース、資産などの手動／自動プロビジョニングを管理し、複雑なワークフローをサポートできます。

  プロビジョニング要求を 1 人または複数の責任者にルーティングするワークフローを作成して、手動プロビジョニングを採用することができます。自動プロビジョニングの場合、アイデンティティボールト内で発生したイベントに応じてワークフローを自動的に開始するように設定することができます。

  詳細については、283 ページのパート V「プロビジョニングワークフローの環境設定と管理」を参照してください。

1.1.1 Identity セルフサービスについて

Identity は、ユーザアプリケーションの基盤となるものです。システム、アプリケーション、およびデータベースへのユーザアクセスを認証する基盤として、Identity が使用されています。各ユーザ固有の ID、および各ユーザの役割には、特定のアクセス権が付与されています。たとえば、マネージャの ID を持つユーザは、直属の部下の給与情報にアクセスできますが、社内の他の従業員の給与情報にはアクセスできません。

ユーザアプリケーションの *[識別セルフサービス]* タブでは、識別情報を表示して、操作を行うための便利な方法が提供されます。このタブでは、あるユーザが必要とする情報へのアクセス権を、必要となった時点でそのユーザに付与できるので、運用の機動性が向上します。たとえば、*[Identity セルフサービス]* タブを使用すると、以下の操作を実行できます。

- 各自のユーザアカウントを直接管理する
- 必要に応じて組織内の他のユーザやグループを検索する
- ユーザやグループの関係を図でわかりやすく表示する
- 関連付けられているアプリケーションを表示する

[Identity セルフサービス] タブの内容の設定は、ユーザアプリケーション管理者が担当します。一般的に、ビジネスユーザが参照できることや、作業できることは、アプリケーションの設定内容、各自のジョブ要件、およびオーソリティのレベルによって異なります。

1.1.2 ワークフローベースのプロビジョニングについて

Identity Manager ユーザアプリケーションの主な機能として、ワークフローベースのプロビジョニングが挙げられます。この機能を利用すれば、社内の保護されているリソースへのユーザアクセスを自動的に承認／撤回することができます。リソースには、ユーザアカウント、コンピュータ、データベースなどのデジタルエンティティが含まれます。

ユーザアプリケーションの *[要求と承認]* タブを利用すれば、簡単にリソースに対する要求を行えます。*プロビジョニング要求*とは、リソースを許可したり、取り消すことを目的にした、ユーザ／システムアクションです。プロビジョニング要求はユーザが直接開始することも (*[要求と承認]* タブを使用)、アイデンティティボールトで発生したイベントに応じて間接的に開始することもできます。

プロビジョニング要求に対して、組織内の 1 人以上の個人による承認が必要な場合、1 つまたは複数のワークフローが開始されます。ワークフローは、要求を満たすために必要な

承認を調整します。1 人の個人からの承認を必要とするプロビジョニング要求もあれば、複数の個人からの承認を必要とするプロビジョニング要求もあります。場合によっては、承認なしに実行できる要求もあります。プロビジョニング要求が正常に処理されると、*プロビジョニングされたリソース*が作成されます。プロビジョニングされたリソースは、Identity Manager のエンタイトルメントにマップされます。

デフォルトでは、ユーザアプリケーションの*［要求と承認］*タブには、プロビジョニング要求は表示されていません。プロビジョニング要求を設定する場合、ビジネスニーズをよく理解している設計者が、リソースをワークフローにバインドする*プロビジョニング要求定義*を作成します。設計者は、*順番に*処理を実行し、各段階で承認を行うワークフローを設定したり、*並行に*処理を実行するワークフローを作成することができます。並行に実行するワークフローの場合、複数のユーザが同時にワークフローの作業を行えます。

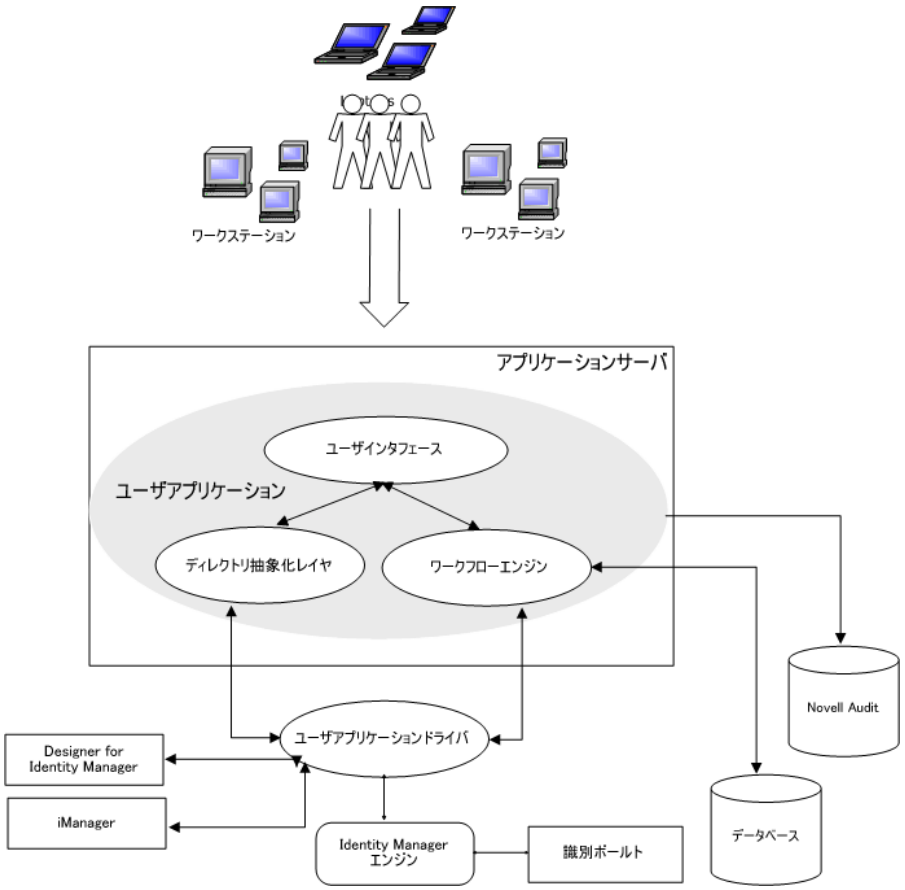
Identity Manager には、一連の Eclipse ベースのツールが用意されています。これらのツールを使って、ワークフロー内のデータ／フロー制御を設計することができます。さらに、Identity Manager には、既存のプロビジョニング要求の環境設定、処理中のワークフローの管理、チームおよびチームの権限の定義などの作業を行うための、一連の Web ベースのツールが用意されています。詳細については、25 ページのセクション 1.4「設計および設定用のツール」を参照してください。

プロビジョニングアプリケーション管理者は、ユーザアプリケーションのワークフローベースのプロビジョニングの管理を担当します。詳細については、21 ページのセクション 1.3「ユーザアプリケーションのユーザタイプ」を参照してください。

1.2 ユーザアプリケーションのアーキテクチャ

Identity Manager ユーザアプリケーションでは、独立した複数のコンポーネントが連携して動作しています。主要コンポーネントを図 1-1 に示します。

図 1-1 ユーザアプリケーションの主要コンポーネント



1.2.1 ユーザインタフェース

Identity Manager ユーザアプリケーションは、ブラウザベースの Java* アプリケーションを使用しています。これは、J2EE* 準拠のアプリケーションサーバ上の Java Web アプリケーションで動作する、JSR168 に準拠したポートレット、JavaServer* ページ、および JavaServer Faces から成り立っています。ユーザアプリケーションのフレームワークは、ウィンドウ状態、ポートレット初期設定、保持、キャッシュ、テーマ、ログなどを管理するコンテナサービスを提供し、セキュリティゲートキーパーとして動作します。これに対し、ユーザアプリケーションが実行されるアプリケーションサーバは、クラスタ化によるスケーラビリティ、JDBC* を経由したデータベースへのアクセス、および証明書ベースのセキュリティサポートなど、アプリケーション全体に対するさまざまなサービスを提供します。

1.2.2 ディレクトリ抽象化層

ディレクトリ抽象化層は、アイデンティティボールトデータの論理的なビューを提供しています。一連のエンティティと関連する属性は、ユーザアプリケーションでユーザに表示、変更、削除させるアイデンティティボールトオブジェクトに基づいて定義します。ディレクトリ抽象化層 :

- アイデンティティボールトに対して、すべてのユーザアプリケーションの LDAP クエリを実行します。これはアイデンティティボールトからプレゼンテーション層ロジッ

クを分離し、識別データに対する要求がすべてディレクトリ抽象化層を経由するようにします。

- ユーザアプリケーション経由で行われたデータ要求の制約とアクセス制御をチェックします。
- アイデンティティボールトから取得されたランタイム環境設定とエンティティ定義データをキャッシュに格納します。参照先 87 ページのセクション 5.1.1「キャッシュ管理」

ディレクトリ抽象化層データ定義の構造を定義するには、ディレクトリ抽象化層エディタプラグイン (iManager の Designer から使用可) を使用します。詳細は、『*Identity Manager ユーザアプリケーション: 設計ガイド*』のディレクトリ抽象化層エディタに関する項目を参照してください。

1.2.3 ワークフローエンジン

ワークフローエンジン (プロビジョニングモジュールで使用可) は、管理者が定義したワークフローのステップの管理と実行、およびステータス情報 (データベース内) の追跡を担当する一連の Java 実行形式ファイルです。必要な承認が付与された後、プロビジョニングシステムは、要求どおりにリソースのプロビジョニングを実行します。

ワークフロー実行時に、ワークフローエンジンは 1 つまたは複数の電子メールメッセージを送信して、ワークフローのステータスの変更を通知することができます。また、電子メールメッセージを送信して、代理人、委任、および可用性設定の更新をユーザに通知することもできます。

Identity Manager の Designer または iManager で電子メールテンプレートを編集し、そのテンプレートを電子メール通知に使用することができます。ランタイム時、ワークフローエンジンはディレクトリからテンプレートを取得し、タグを、通知に適したダイナミックテキストに置き換えます。

ワークフローエンジンの詳細と、プロビジョニングワークフローの環境設定、管理方法については、283 ページのパート V「プロビジョニングワークフローの環境設定と管理」を参照してください。

1.2.4 SOAP エンドポイント

ユーザアプリケーションには、サードパーティ製ソフトウェアアプリケーションがユーザアプリケーションサービスを活用できるように、次の SOAP エンドポイントが用意されています。

表 1-1 SOAP エンドポイント

SOAP エンドポイント	説明
プロビジョニング Web サービス	サードパーティ製ソフトウェアアプリケーションのアクセスをサポートするために、プロビジョニングワークフローエンジンには Web サービスエンドポイントが含まれています。このエンドポイントは、すべてのプロビジョニング機能 (例 :SOAP クライアントに新しい承認ワークフローの開始を許可したり、現在実行中のフローを表示する) を提供しています。
メトリクス Web サービス	ワークフローエンジンには、ワークフローメトリクスを収集する Web サービスも含まれています。ワークフローエンジンにメトリクス Web サービスを追加することにより、承認フロープロセスを監視することができます。また、ビジネスマネージャがプロセスを変更して最適化するために使用するインジケータとしても利用できます。
通知 Web サービス	Identity Manager プロビジョニングモジュールには、電子メール通知機能が用意されています。これを使って、ユーザにプロビジョニングシステムのステータスの変更や、ユーザが実行する必要があるタスクを知らせるメッセージを電子メールで送信することができます。サードパーティアクセスをサポートするために、通知機能には電子メールメッセージを 1 人または複数のユーザに送信する Web サービスエンドポイント機能が用意されています。
ディレクトリ抽象化層 (VDX)Web サービス	ディレクトリ抽象化層は、アイデンティティボールトデータの論理的なビューを提供しています。サードパーティソフトウェアアプリケーションからのアクセスをサポートするために、ディレクトリ抽象化層サービスには、VDXWeb サービスと呼ばれる Web サービスエンドポイントが用意されています。このエンドポイントを利用すれば、ディレクトリ抽象化層に定義されているエンティティに関連する属性にアクセスすることができます。また、グローバル検索と呼ばれる事前定義された検索を実行したり、特定のエンティティを検索することができます。

1.2.5 アプリケーションサーバ (J2EE 準拠)

アプリケーションサーバは、ユーザアプリケーション、ディレクトリ抽象化層、およびワークフローエンジンを実行するための、ランタイムフレームワークを提供しています。ユーザアプリケーションは、Java Web アプリケーションアーカイブ、つまり WAR ファイルとしてパッケージ化されています。WAR は、アプリケーションサーバに展開されます。

ユーザアプリケーションは、JBOSS および WebSphere 上で動作します。サポートするプラットフォームの一覧については、『*インストールガイド*』を参照してください。

1.2.6 Database

ユーザアプリケーションは、次の数種類の情報を保存するためデータベースに依存しています (デフォルトでは MySQL*、サポートされているデータベースについては、『インストールガイド』を参照してください)。

- ユーザアプリケーション環境設定データ : たとえば、Web ページ定義、ポートレットインスタンス登録、および初期設定値などです。
- プロビジョニングモジュールがインストールされている場合、ワークフロー状態の情報はデータベースに永続保存されます (実際のワークフロー定義は、アイデンティティボールト内のユーザアプリケーションドライバに保管されます)
- Novell Audit のログ。

1.2.7 ユーザアプリケーションドライバ

ユーザアプリケーションドライバは、ユーザアプリケーションを動作させるための重要な要素です。これは、次の処理を担当します。

- アプリケーション固有の環境設定データの保管。
- アイデンティティボールトで重要なデータの値が変更された場合に、ディレクトリ抽象化層に通知する。これにより、ディレクトリ抽象化層のキャッシュが更新されます。

プロビジョニングモジュールがインストールされている場合、ユーザアプリケーションドライバを次のように環境設定できます。

- アイデンティティボールト内のイベントを契機に、ワークフローを起動する。
- ワークフローのプロビジョニングアクティビティの成功／失敗をユーザアプリケーションデータベースに通知する。これにより、ユーザは要求の最終的なステータスを参照することができます。
- アイデンティティボールト内の属性値の変更に応じて、ワークフローを自動的に起動する。

ユーザアプリケーションドライバはランタイムコンポーネントであるだけでなく、ディレクトリオブジェクト (ユーザアプリケーションのランタイムの生成物で構成される) のストレージラッパーでもあります。

表 1-2 *ユーザアプリケーションドライバに保存されるアーティファクト*

アーティファクト	説明
ドライバセットオブジェクト	Identity Manager の各インストールでは、ドライバをドライバセットとしてグループ化する必要があります。1 つのディレクトリサーバで、一度に 1 つのドライバセットだけをアクティブにできます。セット内のドライバは、ドライバセット全体に影響を与えることなく個別にオンとオフを切り替えることができます。ユーザアプリケーションドライバも、他の Identity Manager ドライバと同じように、ドライバセット内に存在していなければなりません。ドライバセットはユーザアプリケーションによって自動作成されるわけではありません。ユーザがドライバセットを作成し、その中にユーザアプリケーションドライバを作成する必要があります。

アーティファクト	説明
ユーザアプリケーション	ユーザアプリケーションドライバおぶじぇくとは、さまざまなアーティファクトのコンテナです。ユーザアプリケーションドライバには、発行者および購読者チャネルオブジェクトとポリシーが実装されています。発行者チャネルがユーザアプリケーションによって使用されることはありませんが、カスタムで使用することもできます。
App Config オブジェクト	AppConfig オブジェクトはさまざまなユーザアプリケーション設定オブジェクトのコンテナです。 ◆ **RequestDefs:** プロビジョニング要求定義用コンテナここに保存される定義 (XML 形式) は、適切な権利を持つエンドユーザがユーザを使用してインスタンス化できる要求のクラスを表します。(プロビジョニングモードのみ) ◆ **WorkflowDefs:** ：ワークフローオブジェクトのコンテナで、デザインタイムの説明に加えてテンプレートや未使用のフローが含まれます。 ◆ **ResourceDefs:** プロビジョニングリソース定義のコンテナで、デザイン時の説明に加え、テンプレートや未使用のターゲットも含まれます。 ◆ **ServiceDefs:** サービス定義オブジェクトのコンテナで、ワークフローによって呼び出される Web サービスをラップします。 ◆ **DirectoryModel:** ユーザアプリケーションに表示される、アイデンティティボールト内の異なるコンテンツの種類を表すディレクトリ抽象化層オブジェクトです。 ◆ **AppDefs:** キャッシュ設定情報や電子メール通知プロパティなど、ランタイム環境を初期化する設定オブジェクトのコンテナです。 ◆ **ProxyDefs:** 代理人定義のコンテナです。 ◆ **DelegateeDefs:** 委任定義のコンテナです。

1.2.8 Designer for Identity Manager

Identity Manager の Designer は、ディレクトリ抽象化層オブジェクト、プロビジョニング要求、および関連するワークフローを定義するための、一連のプラグインを提供しています。詳細については、25 ページのセクション 1.4「設計および設定用のツール」を参照してください。

1.2.9 iManager

iManager には、プロビジョニング要求およびそれに関連付けられたワークフローの設定および管理に使用するための一連のプラグインが用意されています。これらのツールを使って、プロビジョニングチームとチームの権限を定義することもできます。詳細については、25 ページのセクション 1.4「設計および設定用のツール」を参照してください。

1.2.10 Identity Manager エンジン

Identity Manager エンジンは、アイデンティティボールトおよび接続されているシステム内のイベントを監視する、ランタイムフレームワークを提供しています。これは、アイデンティティボールトのポリシーとルートデータを強制します。Identity Manager ユーザアプリケーションは接続システムです。アイデンティティボールト、ユーザアプリケーショ

ンのディレクトリ抽象化層、およびワークフローエンジン間の通信は、ユーザアプリケーションドライバを経由して行われます。

1.2.11 識別ボールト

アイデンティティボールトは、ユーザデータ (および他の ID データ)、および Identity Manager ドライバセットとユーザアプリケーションドライバのリポジトリです。ユーザアプリケーションはさまざまなアイデンティティボールトオブジェクトに依存するため、ユーザアプリケーションで要求されるカスタム LDAP オブジェクトとその属性に対応するように、eDirectory‘ スキーマを拡張する必要があります。スキーマの拡張は、ユーザアプリケーションのインストールの一環として自動的に行われます。カスタムオブジェクトと属性は、ユーザアプリケーションをインストールしてアクティブにした後に、デフォルト値で編成されます。

1.2.12 Novell Audit

Novell Audit は、さまざまな種類のデータ (ワークフローの手順で生成されたデータなど) を永続保存できる独立したログサーバです。詳細については、69 ページの第 3 章「ログのセットアップ」を参照してください。

1.3 ユーザアプリケーションのユーザタイプ

Identity Manager ユーザアプリケーションのユーザは、次のカテゴリに分類されます。

- 管理者
- ユーザ
- 設計者

1.3.1 管理者

ユーザアプリケーションには、さまざまな管理ユーザが定義されています。表 1-3 に定義されている管理ユーザは、インストール時に定義されます。

表 1-3 ユーザアプリケーションの管理ユーザ

User	説明
LDAP 管理者	アイデンティティボールトを環境設定できる権限を持つユーザです。これは、論理的な役割で、他のタイプの管理ユーザと共有することができます。 LDAP 管理者アカウントは、新規ユーザ、グループ、コンテナの作成など、通常はログインしたユーザが権限を持たないタスクを、ユーザアプリケーションが LDAP サーバ上で処理するための代理ユーザです。これは、アイデンティティボールトにバインドしてシステム LDAP 操作を行うために使われる資格情報 (ユーザ名とパスワード) を表しています。これらはユーザアプリケーション自身を実行するために必要な権限です。LDAP 管理者に必要なものを次に示します。 ◆ ユーザアプリケーションドライバと、そのオブジェクトにタイするスーパバイザ権。このためには、ドライバコンテナレベルで権限を設定し、それを継承可能にします。 ◆ ディレクトリ抽象化層ユーザエンティティ定義を通じて定義されたユーザに対するスーパバイザエントリ権。これには、objectClass、および DirXML-EntitlementRecipient、srvprvEntityAux、および srvprvUserAux 補助クラスに関連するすべての属性への属性書き込み権がなければなりません。 ◆ コンテナオブジェクト cn=DefaultNotificationCollection, cn=Security に対する読み込み権。このオブジェクトは、自動プロビジョニング電子メールに使用される電子メールサーバ設定に保持されます。これには、電子メールサーバ自体を認証する SecretStore 資格情報を含めることができます。

User	説明
ユーザアプリケーション管理者	ユーザアプリケーションの管理作業を実施できる権限を持つユーザです。このユーザは次の操作を行うことができます。 ◆ ユーザアプリケーションを管理するには、*［管理］*タブを使用します。 ◆ ワークフロータスク (ワークフローの有効化、無効化、停止など) を管理するには、iManager を使用します。 ◆ 新しいプロビジョニング要求の作成や電子メールテンプレートの管理を行うには、iManager または Designer を使用します。 ◆ Novell Audit のログデータに対するレポートの実行。 このユーザには、*［要求と承認］*タブに対する特別な権限はありません。 このユーザは、［管理］ページでアプリケーションレベルのアクセスを制御するため、特にディレクトリ権は必要ありません。ユーザアプリケーション管理者は［管理］ページのテーマを管理できますが、ユーザアプリケーションは LDAP 管理者の資格情報を使って、アイデンティティボールト内のテーマ選択を変更します。 **パスワードセルフサービス：** ユーザアプリケーション管理者が行う作業の 1 つに、ユーザアプリケーション用のパスワードセルフサービスの環境設定があります。パスワードセルフサービスの機能は、パスワード同期化ステータスです。ユーザアプリケーション管理者に対して他のユーザのパスワード同期化ステータスを参照できるようにする場合 (トラブルシューティングや他の目的で)、PasswordManagement グループを作成して、そのグループにユーザを割り当てることをおすすめします。PasswordManagement グループのメンバーには、他のユーザのパスワード同期ステータスを表示する権利が与えられます。このグループを作成する場合、次の条件を満たしていなければなりません。 ◆ 名前は「PasswordManagement」にする。 ◆ アイデンティティボールトに対する権限を与える。このグループには、パスワード同期ステータスを表示するユーザの eDirectory オブジェクト属性に対する読み込み権が必要です。
プロビジョニングアプリケーション管理者	ユーザアプリケーションに対するすべての管理権を与えずに、プロビジョニング管理タスクをビジネスユーザに委任できるようにすることを目的にしたユーザです。デフォルトでは、プロビジョニング管理者は［管理］ページにはアクセスできません。ただし、*［要求と承認］*タブに対しては、すべての権限を保有しています。たとえば、プロビジョニングアプリケーション管理者のログイン時には、すべてのユーザがそのチームメンバーとみなされるため、管理者がチームを選択する必要はありません。

iManager 管理者

上記のユーザと関連タスクに加えて、Identity Manager には iManager を使って次の作業を行う管理者が用意されています。

- 新しいプロビジョニング要求とワークフローを作成する。
- チームを定義する。
- 電子メールテンプレートを定義、管理する。
- ワークフロータスクを管理する (ワークフローの有効化、無効化、停止など)。

これらのタスクを実行するユーザを上記のような管理者にしたり、これらのタスクを実行する権限が与えられた別のユーザを管理者にすることができます。

iManager でワークフローオブジェクトを作成、編集するには、特定のユーザアプリケーションドライバの RequestDefs.AppConfig コンテナに対して次の権限が必要です。

- [Entry Rights] スーパバイザまたは作成。
- [All Attribute Rights] スーパバイザまたは書き込み。

ワークフローを開始するには、ユーザは特定のユーザアプリケーションドライバ (委任モデルを使用している場合は要求定義オブジェクト個別) の RequestDefs.AppConfig コンテナに対する参照 [Entry Rights] 権限がなければなりません。

1.3.2 設計者

設計者は Identity Manager の Designer を使って、ユーザアプリケーションをカスタマイズします。Designer は、企業の ＩＴ 開発者、コンサルタント、セールスエンジニア、システム設計者、システム管理者などの、ディレクトリ、データベース、自社の ＩＴ 環境を深く理解しており、ソリューションの設計者としての役割を果たす ＩＴ の専門家を対象にしたツールです。

Designer でワークフローオブジェクトを作成、編集するには、特定のユーザアプリケーションドライバの RequestDefs.AppConfig コンテナに対して次の権限が必要です。

- [Entry Rights] スーパバイザまたは作成。
- [All Attribute Rights] スーパバイザまたは書き込み。

ワークフローを開始するには、ユーザは特定のユーザアプリケーションドライバ (委任モデルを使用している場合は要求定義オブジェクト個別) の RequestDefs.AppConfig コンテナに対する参照 [Entry Rights] 権限がなければなりません。

1.3.3 ユーザ

ユーザは、ユーザアプリケーションの *[Identity セルフサービス]* タブと *[要求と承認]* タブ (プロビジョニングがインストールされている場合) を表示して操作を行う人々です。次の種類のユーザがあります。

- *認証ユーザ* (従業員、マネージャ、従業員やマネージャから委任された人、または代理人)。*委任ユーザ*は、他のユーザの代わりにタスクの作業を行うために、1 つまたは複数のタスクを委任された人です。*代理ユーザ*は、他のユーザの識別情報を一時的に引き継ぐことによって、そのユーザの役割を果たすエンドユーザです。元のユーザの権利はすべて代理ユーザに適用されます。元のユーザが担当していた仕事は、引き続きそのユーザが担当していることになります。
- *匿名またはゲストユーザ*匿名ユーザは、パブリック LDAP ゲストアカウント、またはアイデンティティボールトに設定されている特別なアカウントです。ユーザアプリケーション管理者は、[*Identity セルフサービス*] タブの一部の機能に対して匿名アクセスを有効にすることができます。また、ユーザアプリケーション管理者は、ユーザがリソースを要求するためのページを作成することもできます。匿名アクセスの設定方法については、28 ページの 表 1-8 を参照してください。

ユーザアプリケーションでユーザが利用できる機能は、ユーザアプリケーション管理者の設定内容によって異なります。次のように環境設定できます。

- 組織図ポートレットを使ってユーザオブジェクト間の階層関係を表示する。
- (ユーザが適切な権利を持つ) ユーザ情報の \'95\'5c 示と編集。

- 高度な検索条件を使用した、ユーザやリ \'83\'5cースの検索 (保存して再使用できます)。
- 忘れてしまったパスワードの回復。

プロビジョニングモジュールがインストールされている場合、ユーザに次の作業を行わせるようにユーザアプリケーションを環境設定できます。

- リ \'83\'5cースの要求 (事前定義された多数のワークフローの 1 つの開始)。
- 以前の要求のステータスの \'95\'5c 示。
- タスクの要求およびタスクリストの \'95\'5c 示 (リ \'83\'5cース、受信者、または他の特性ごと)。
- プロキシの割り当ての \'95\'5c 示。
- 委任の割り当ての \'95\'5c 示。
- 自分の可用性を指定する。
- 他のユーザに代わってタスクを要求するためにプロキシモードに入る。
- チームタスクの \'95\'5c 示やチームリ \'83\'5cースの要求など (\'83\'7d ネージャのみ)。

1.4 設計および設定用のツール

さまざまな管理者が次のツールを使って、Identity Manager ユーザアプリケーションの設計と環境設定を行えます。

表 1-4 *ユーザアプリケーションの設計と環境設定を行うためのツール*

ツール	目的
Designer for Identity Manager	Identity Manager の環境設定と展開を行うための、強力なグラフィカルツールセットです。次のプラグインは、ユーザアプリケーションの環境設定を支援することを目的に設計されています。 ◆ ディレクトリ抽象化層 : ユーザアプリケーションに必要なアイデンティティボールトオブジェクトを定義できます。 ◆ プロビジョニング要求定義エディタ : プロビジョニング要求定義用のワークフローを作成できます。また、ユーザが要求や電子メールを作成、承認するフォームをカスタマイズすることもできます。 ◆ プロビジョニングビュー : ディレクトリ抽象化層とプロビジョニング要求をユーザアプリケーションドライバにインポート、エクスポート、展開、移行できます。 詳細については、*『Identity Manager User Application: Design Guide』* を参照してください。

ツール	目的
iManager(プロビジョニングモジュールのみ)	Web ベースの管理コンソールです。次のプラグインは、ユーザアプリケーションの環境設定と管理を支援することを目的に設計されています。 ◆ プロビジョニング要求環境設定プラグイン : プロビジョニング要求定義をプロビジョニングされたリソースに関連付け、関連ワークフローのランタイム特性を指定し、定義を有効にできます。 ◆ ワークフロー管理プラグイン : ブラウザベースのインタフェースを使用して、ワークフロープロセスのステータスを表示したり、ワークフロー内のアクティビティを再割り当てしたり、応答がなく再開始できないワークフローを終了したりすることができます。 ◆ プロビジョニングチームプラグイン : チームの特徴を定義できます。チームはユーザのグループを表し、このチームに関連付けられたプロビジョニング要求と承認タスクを管理できるユーザが定義されています。チーム定義は、チームマネージャ、チームメンバー、およびチームオプションのリストで構成されています。 ◆ プロビジョニングチーム要求プラグイン : チームの要求権限を指定できます。チーム要求オブジェクトは、チームのドメイン内に該当する要求や、チームマネージャに与えられた権限を指定します。要求権限は、プロビジョニング要求やタスクでチームマネージャが実行できるアクションを指定します。 詳細については、283 ページのパート V「プロビジョニングワークフローの環境設定と管理」を参照してください。
ユーザアプリケーションの［管理］タブ	ユーザアプリケーションの環境設定、管理、カスタマイズを行うための、Web ベースの管理コンソールです。このタブには、次のページがあります。 ◆ アプリケーション環境設定 : キャッシュ、LDAP パラメータ、ログ、テーマ、パスワードモジュール設定などを設定できます。 ◆ ページ管理 : 新しいポートレットを作成したり、既存の Identity セルフサービスページをカスタマイズできます。 ◆ ポートレット管理 :Identity セルフサービスページで使用するポートレットを新しく作成したり、既存のポートレットをカスタマイズできます。 ◆ プロビジョニング : 委任、代理人、タスク、デジタル署名サービス、エンジンとクラスタの設定などの環境設定を行えます。 ◆ セキュリティ : プロビジョニング管理者およびユーザアプリケーション管理者の権限を定義できます。 詳細については、79 ページのパート III「ユーザアプリケーションの管理」を参照してください。

ツール	目的
lreport.exe (ログレポートツール) および iManager の監査およびログ機 \'94\'5c	事前定義された数多くのログレポート (Identity Manager 付属) を Crystal Reports *(.rpt) 形式で出力し、Novell Audit データベースに記録されたデータをフィルタリングできます。lreport.exe ログレポートツール (Windows* のみ) は、レポートを生成する方法の 1 つです。他の方法を使用してレポートを作成することもできます。詳細については、69 ページの第 3 章「ログのセットアップ」を参照してください。

1.5 次に行う作業

ここでは、Identity Manager ユーザアプリケーションの機能とアーキテクチャについて学習しました。これで、ユーザアプリケーションを自分のビジネスニーズに合わせてカスタマイズできます。通常は、次のようなカスタマイズを行います。

- ユーザインタフェースと Identity セルフサービス機能のカスタマイズ。詳細については、27 ページの 表 1-6 を参照してください。
- 要求と承認機能の設定 (プロビジョニングモジュールがインストールされている場合)。詳細については、28 ページの 表 1-8 を参照してください。
- 運用環境の設定。詳細については、28 ページの 表 1-7 を参照してください。

表 1-5 *ユーザインタフェースと Identity セルフサービス機能のカスタマイズ*

学習対象	参照先
ディレクトリ抽象化層の設定	*Identity Manager ユーザアプリケーション : 設計ガイド*
Identity セルフサービスのカスタマイズ	207 ページのパート IV「ポートレットリファレンス」
新しいページの追加とページセキュリティの設定	139 ページの第 6 章「ページの管理」
Identity ポートレットのカスタムインスタンスの作成	173 ページの第 7 章「ポートレットの管理」
ユーザアプリケーションのテーマやブランド設定の変更	106 ページのセクション 5.1.6「テーマ管理」
ユーザアプリケーションのユーザインタフェースのローカライズ	66 ページのセクション 2.8「テキストのローカライズ」
パスワードセルフサービスの有効化	119 ページのセクション 5.3「パスワード管理の環境設定」

表 1-6 *要求と承認機能の設定*

学習対象	参照先
プロビジョニング要求の作成	*Identity Manager ユーザアプリケーション : 設計ガイド*および 299 ページの第 17 章「プロビジョニング要求定義の設定」
要求と承認フォームのカスタマイズ	*Identity Manager ユーザアプリケーション : 設計ガイド*

学習対象	参照先
チームの定義	349 ページの第 19 章「プロビジョニングチームの環境設定」
電子メールテンプレートの定義	*Identity Manager ユーザアプリケーション : 設計ガイド*および 336 ページのセクション 18.4「電子メールテンプレートに関する作業」

表 1-7 *ユーザアプリケーション運用環境の設定*

学習対象	参照先
運用環境トポロジー	31 ページのセクション 2.1「トポロジ」
セキュリティの設定	33 ページのセクション 2.2「セキュリティ」
デジタル署名サポートの設定	37 ページのセクション 2.3「デジタル署名の環境設定」
パフォーマンスチューニング方針	48 ページのセクション 2.6「パフォー \'83\'7d ンスの調整」
クラスタの設定	54 ページのセクション 2.7「クラスタリング」
ログの設定	69 ページの第 3 章「ログのセットアップ」

表 1-8 *ゲストアクセス用ユーザアプリケーション環境設定 :*

学習対象	参照先
ゲスト／匿名アカウント	45 ページのセクション 2.4「ユーザアプリケーションへの匿名／ゲストアクセスを有効にする」
匿名ユーザの自己登録の許可	220 ページのセクション 11.4「作成ポートレットの自己登録の環境設定」
ディレクトリ検索への匿名アクセスの許可	280 ページのセクション 15.3「匿名アクセス用のリスト検索の環境設定」
マイプロファイルや組織図への匿名アクセスの許可	237 ページのセクション 12.6「詳細ポートレットの匿名アクセスの設定」および 265 ページのセクション 13.3「ゲストアクセス用の組織図の環境設定」
ワークフローへの匿名アクセスの許可	267 ページの第 14 章「リソース要求ポートレット」

ユーザアプリケーション環境の設定

これらの節では、Identity Manager ユーザアプリケーション環境のさまざまな要素を組織のニーズに合わせて設定する方法について説明します。

製品環境の設計

2

この節では、運用環境のセットアップに関する事項について解説します。サンドボックス環境やテスト環境、または他の運用前環境から運用環境へ移行する際の考慮事項について説明します。

このガイドの構成を次に示します。

2.1 トポロジ

各主要サブシステムは多数のインスタンスを持つことができ、さまざまな接続手段があります。可能なレイアウトがすべてサポートされるわけではありません。この節は、一部の設定方法が他の設定方法に比べて広く使われているかを説明する 3 つのサブセクションに分かれています。

2.1.1 最小設計

ユーザアプリケーションの最も簡単な論理構成は「すべてを 1 つずつ」インストールする方法です。1 つのアイデンティティボールトツリー、Identity Manager エンジンとドライバに 1 インスタンス、ユーザアプリケーションの 1 インスタンスを実行する Application Server の 1 インスタンスで構成されます。物理的な実装の観点から考えると、論理的にはこのすべてを 1 台のコンピュータで実行できます。しかし実際にはさまざまな理由 (セキュリティ、メンテナンス性、パフォーマンスなど) から、この実装はお勧めできません。実用的な実際のインストールで必要なコンピュータの台数を決めるときは、最低限次の点を考慮してください。

Novell Audit サーバ : このアプリケーションは、ランタイム時にユーザアプリケーション環境からのイベント情報の取得を担当します (他の多数の情報も取得することがあります)。社内の他のアプリケーションの永続保存ストアとして二重の役割を果たす場合もあります。さまざまな理由から、Identity Manager システムの他の主要部分 (アプリケーションサーバやアイデンティティボールトなど) を、Novell Audit サーバと同じコンピュータに置くことはお勧めできません。

識別ボールト : アイデンティティボールトは、非常に大量のトラフィックが発生するコンポーネントのため、高いパフォーマンスとスケーラビリティが求められます。アイデンティティボールトは専用のコンピュータに置くことを検討してください。アプリケーションサーバなどのトラフィックが多いシステムと同じコンピュータにアイデンティティボールトを置くことはお勧めできません。

データベース : サポートするデータベースのこのインスタンスが Novell® Audit データベースでもある場合、このインスタンスを専用のコンピュータで実行することもできます。ユーザアプリケーションは、このコンポーネントを次の目的で使用します。

- ポータル設定データの永続ストア
- 処理中のワークフローに関する状態情報の永続保存ストア (Provisioning Module がインストールされている場合)
- Novell Audit のログストア (オプション)

アプリケーションサーバ : パフォーマンスおよび容量上の理由から、このシステムは専用のコンピュータで実行してください。

これらの検討事項では、最小 3 台構成の環境を提案しています。

2.1.2 高可用性の設計

クラスタリングによる高可用性と高容量については、54 ページのセクション 2.7「クラスタリング」を参照してください。ここでは、次の点に留意してください。

- Identity Manager はマルチノードインストールおよび共有ストレージメカニズムを使用してアイデンティティボールト、エンジン、およびドライバの高可用性をサポートしています。詳細については、『*Identity Manager 管理ガイド*』の「高可用性」を参照してください。SUSE® Linux を使ってこのようなシステムを設定する手順の概要は、次の項目を参照してください。

  http://support.novell.com/cgi-bin/search/searchtid.cgi?/10093317.htm (http://support.novell.com/cgi-bin/search/searchtid.cgi?/10093317.htm)

- ユーザアプリケーションの高可用性は、JBoss クラスタリングにより実現できます。各ノードが 1 つのユーザアプリケーションインスタンスを実行するように JBoss クラスタを設定できます。これらのインスタンスは、すべて同等 (ピア) です。
- 自動フェイルオーバーがサポートされています。割り込みされたワークフローは、クラスタノードがなくなった後に再開できます。

詳細については、54 ページのセクション 2.7「クラスタリング」を参照してください。

2.1.3 設計上の制約

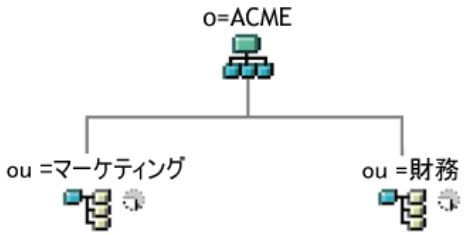
設計上、次の 2 つの重要な制約があります。

- ユーザアプリケーションインスタンスは、複数のユーザコンテナを処理する (検索、クエリ、およびユーザの追加などを行う) ことはできません。また、アプリケーションとコンテナの関係は永続的なものと見なされます。
- ユーザアプリケーションドライバを複数のユーザアプリケーションに関連付けることはできません。ただし、複数のユーザアプリケーションが同じ JBoss クラスタにある同等の複数のノードにインストールされている場合は例外です。つまり、ドライバとユーザアプリケーション間において 1 対多のマッピングはサポートされていません。

1 番目の制約により、ユーザアプリケーションの設計には高度なカプセル化が求められます。

たとえば、次のような組織 \'8d\'5c 造があるとします。

図 2-1 *サンプルの組織構造*



ユーザアプリケーションのインストール時、アイデンティティボールト内でユーザアプリケーションの検索対象となる最上位のユーザコンテナを指定するよう求められます。この場合、「ou=Marketing,o=ACME」や「ou=Finance,o=ACME」のように指定できます。両方を指定することはできません。ユーザアプリケーションの検索とクエリ (および管理者ログイン) はすべて、指定したコンテナのいずれかを検索範囲にして実行されます。

注 : 理論上は、o=ACME を検索範囲に指定すれば Marketing と Finance の両方を網羅できます。しかし大規模な組織では、(Marketing と Finance に関係する 2 つのコンテナだけではなく) 多数の ou コンテナが存在する可 \'94\'5c 性があるため、実用的ではありません。

もちろん、(リソースを共有しない) 2 つの独立したユーザアプリケーションのインストールを作成し、1 つをマーケティング用、もう 1 つを財務用として使用することもできます。各インストールは、それぞれ独自のデータベース、および適切に設定されたユーザアプリケーションドライバを持ちます。各ユーザアプリケーションは別々に管理され、独自のテーマを持つこともあります。

1 つのユーザアプリケーションインストールの同じ検索範囲に Marketing と Finance を設定する必要がある場合は、2 つの方法が考えられます。1 つの方法としては、2 つの同等ノードの上位に新しいコンテナオブジェクト (ou=MarketingAndFinance など) を挿入し、その新しいコンテナを検索範囲のルートとしてポイントする方法があります。もう 1 つの方法は、フィルタリングされたレプリカ (特殊なタイプの eDirectory‘ ツリー) を作成して元の ACME ツリーで必要な部分を組み合わせ、そのレプリカのルートコンテナでユーザアプリケーションをポイントする方法です。(フィルタリングされたレプリカの詳細については、『Novell eDirectory Administration Guide』を参照してください)。

特定のシステムレイアウトについて質問がある場合は、Novell の担当者までお問い合わせください。

2.2 セキュリティ

この項では、次のトピックについて説明します。

- 34 ページのセクション 2.2.1「セキュリティの概要」
- 35 ページのセクション 2.2.2「自己署名付き証明書」
- 35 ページのセクション 2.2.3「SSL を有効にする」
- 36 ページのセクション 2.2.4「SOAP セキュリティの有効化」

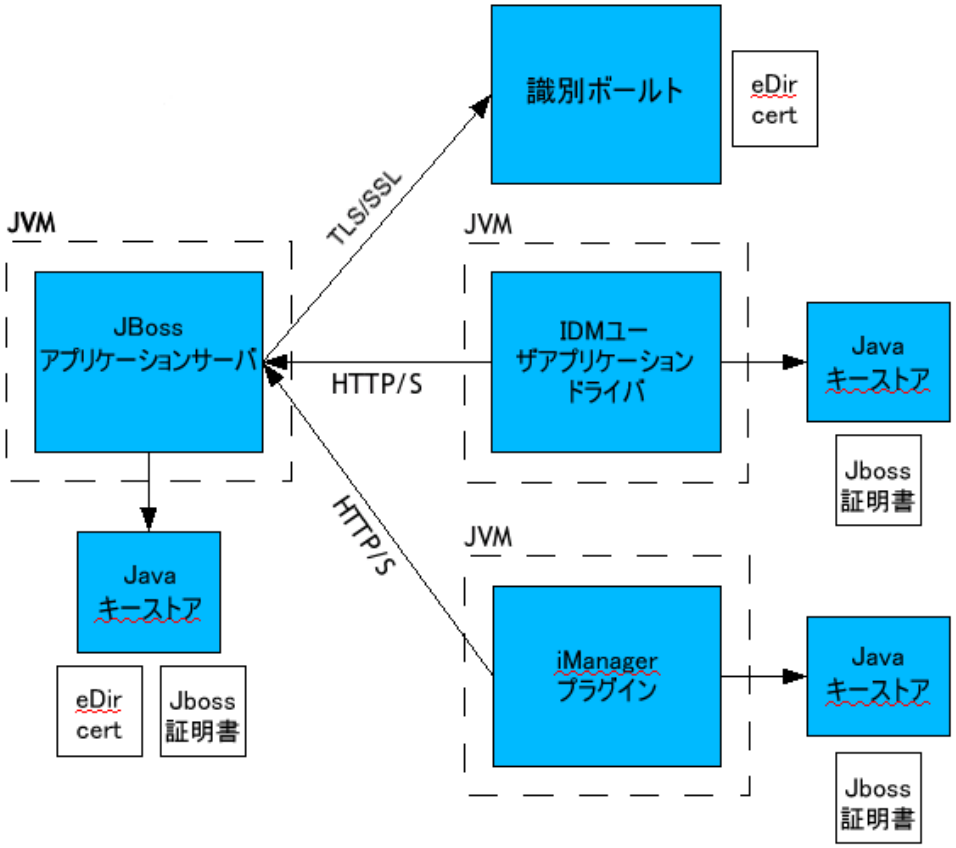
- 36 ページのセクション 2.2.5「相互認証」
- 36 ページのセクション 2.2.6「サードパーティ認証とシングルサインオン」
- 37 ページのセクション 2.2.7「重要なユーザアプリケーションデータの暗号化」

2.2.1 セキュリティの概要

運用前段階から運用段階に移行するときは、通常、システムのセキュリティ面を強化する必要があります。サンドボックステスト中、HTTP を使ってユーザアプリケーションドライバとアプリケーションサーバを接続することができます。また、ドライバとアプリケーションサーバ間の通信に、自己署名証明書 (一時的な手段として) を使用することもできます。運用環境では、会社の Verisign *(または他の信頼できるプロバイダ) の証明書に基づいたサーバ認証による安全な接続を使用する必要があります。

Identity Manager のユーザアプリケーション環境では、次の図のようにさまざまな部分で X.509 証明書が使用されます。

図 2-2 *Identity Manager ユーザアプリケーション環境*



デフォルトでは、ユーザアプリケーションとアイデンティティボールト間の通信はすべて、TLS (Transport Layer Security) により保護されます。アイデンティティボールト (eDirectory) 証明書は、インストール時に JBoss アプリケーションサーバキーストアへ自動的にインストールされます。特に指定しない限り、ユーザアプリケーションのインストーラは、eDirectory 証明書のコピーを JRE のデフォルト *cacerts* ストアに保存します。

WebSphere サーバキーストアへの証明書のインストールは、WebSphere ツールを使って手動で行う必要があります。

安全に通信するには、図のようにサーバ証明書を複数の場所に配置する必要があります。図中の *JBoss cert* ボックスが表示されている場所で、自己署名付き証明書を使用するか、Verisign などの認証局 (CA) によって発行された証明書を使用するか (代わりに) に応じて、異なる設定手順が必要です。

2.2.2 自己署名付き証明書

有名な信頼できる認証局 (Verisign など) が発行した証明書を使用する場合には、特別な設定手順は必要ありません。自己署名付き証明書を作成して使用する場合は、次の手順に従ってください。

1 次のようなコマンドライン構文を使用して、自己署名付き証明書のキーストアを作成します。dname にはご自分の Web サイトと組織を指定します。必要に応じて他の値も変更してください。

```
keytool -genkey -alias IDM -keyalg RSA -storepass changeit -
keystore jboss.jks -dname "cn=www.novell.com,o=Novell,s=MA,c=US" -
keypass changeit
```

証明書のほかに「jboss.jks」というファイルも作成します。

2 キーストアファイル (jboss.jks) を次の例のような JBoss ユーザアプリケーションディレクトリにコピーします。

```
cp jboss.jks ~/jboss-4.2.0.GA/WAR/conf
```

2.2.3 SSL を有効にする

ユーザアプリケーションは、認証に HTML フォームを使用します。その結果、ログイン時にユーザ資格情報がさらされてしまいます。これらの重要な情報を保護するために、SSL を有効にすることを強くお勧めします。

35 ページの 表 2-1 に、SSL 導入に関する説明の参照情報を示します。

表 2-1 *SSL 導入時の指針*

指針	次のトピックを参照してください。
SSL を使用するための Access Manager の設定	ユーザアプリケーションへのログインに Access Manager を使用する場合は、『*Novell Access Manager 3.0 SP1 セットアップガイド* (http://www.novell.com/documentation/novellaccessmanager/basicconfig/index.html?page=/documentation/novellaccessmanager/basicconfig/data/bookinfo.html)』を参照してください。
SSL を使用するためのアプリケーションサーバの設定	アプリケーションサーバメーカーが提供するドキュメントを参照してください。IDM ユーザアプリケーションで SSL の使用を設定する前に、アプリケーションサーバの SSL の使用を設定してください。

指針	次のトピックを参照してください。
SSL を使ってアイデンティティボールトに接続するための IDM ユーザアプリケーション (クライアントとして) の設定	インストール時に［セキュアな管理者接続］と［セキュアなユーザ接続］設定パラメータを設定する方法については、『IDM 3.5.1 インストールガイド (http://www.novell.com/documentation/idm35/pdfdoc/install/install.pdf)』の第 5 章にあるユーザアプリケーションのインストールに関する項目を参照してください。これらのパラメータは、configupdate ユーティリティを使って編集することもできます。

2.2.4 SOAP セキュリティの有効化

1 IDM.war で web.xml ファイルを検索し、テキストエディタで開きます。

2 ファイルの最後の方にある次のセクションをアンコメントします。

```
<security-constraint>
        <web-resource-collection>
                <web-resource-name>IDMProv</web-resource-name>
                <url-pattern>/*</url-pattern>
                <http-method>POST</http-method>
                <http-method>GET</http-method>
               <description>IDM Provisioning Edition</description>
                </web-resource-collection>
        <user-data-constraint>
                <transport-guarantee>CONFIDENTIAL</transport
guarantee>
        </user-data-constraint>
</security-constraint>
```

3 ファイルとアーカイブを保存して、JBoss を再起動します。

2.2.5 相互認証

Identity Manager ユーザアプリケーションは、クライアント証明書ベースの認証はサポートしていません。ただし、Novell® Access Manager を使用すれば、この機能を入手できます。詳細については、Novell の担当者までお問い合わせください。36 ページのセクション 2.2.6「サードパーティ認証とシングルサインオン」も参照してください。

2.2.6 サードパーティ認証とシングルサインオン

Identity Manager ユーザアプリケーションでは、Access Manager にログインできる任意のサードパーティ認証サービスを使用して、Access Manager を通じたシングルサインオンをサポートしています。この機能は、非パスワードベースの技術を使用して、Access Manager を通じてユーザアプリケーションにログインすることを有効にします。たとえば、スマートカードを使用したユーザ (クライアント) 証明書を介したログインです。

Access Manager は、ユーザを IDM アイデンティティボールト内の DN にマップします。Access Manager を使ってユーザアプリケーションにユーザがログインする際に、Access Manager は SAML アサーション (ユーザの DN を識別子として) を HTTP ヘッダに挿入し、そのリクエストをユーザアプリケーションに転送することができます。ユーザアプリケーションは SAML アサーションを使ってアイデンティティボールトとの LDAP 接続を確立します。この機能をサポートするための Access Manager の設定方法については、Access Manager のマニュアルを参照してください。

リクエストをやり取りするチャネルを保護するには、そのチャネルをファイアウォール内に置くか SSL 接続を使用してください。ユーザアプリケーション環境で SSL を設定する場合の参照情報については、35 ページの 表 2-1 を参照してください。

現時点でパスワードに基づいたシングルサインオン認証を許可するアクセサリポートレットは、ユーザアプリケーション認証に SAML アサーションを使用する場合、シングルサインオンをサポートしていません。

2.2.7 重要なユーザアプリケーションデータの暗号化

永続的に保管されるユーザアプリケーションに関連する重要な情報は、AES-128 アルゴリズムを使って暗号化されます。マスタキー自身も PBEWithSHA1AndDESede を使ったパスワードベースの暗号化により保護されます。パスワードがメモリに永続的に保管されることはありません。

次のような情報が暗号化されます (ただし、これ以外にも暗号化される情報があります)。

- LDAP 管理者のユーザパスワード
- LDAP ゲストユーザパスワード
- ＤＳＳ トラステッド CA キーストアパスワード
- DSS 署名キーキーストアパスワード
- DSS 署名キーエントリパスワード
- Novell Audit 署名キー

ただしクラスタ環境では、セッションのフェイルオーバーを有効にすると、ユーザセッション内の一部の重要なデータ (たとえば、ポートレットシングルサインオン用のログインパスワードなど) がセッションのレプリケーション時にネットワーク上に送信される可能性があります。そのため、機密データがネットワークスニファに見られる可能性があります。このような重要なデータを保護するには、次のいずれかの作業を行ってください。

- JGroups の暗号化を有効にします。JGroups の暗号化を有効にする方法の詳細については、JGroups Encrypt (http://wiki.jboss.org/wiki/Wiki.jsp?page=JGroupsENCRYPT) を参照してください。
- クラスタがファイアウォールの背後にあることを確認します。

2.3 デジタル署名の環境設定

この節では、Identity Manager ユーザアプリケーションに用意されているデジタル署名サポートを活用するための環境設定について説明します。

注 : デジタル署名機能用に Novell Certificate Server‘(Novell PKI インフラストラクチャ) を使用する場合は、eDirectory 8.8 以上を使用する必要があります。デジタル署名機能には、eDirectory 8.8 に付属する PKI 3.1 が必要です。

警告 : デジタル署名したドキュメントを保持するには、Novell Audit(または Sentinel) を使用する必要があります。デジタル署名ドキュメントはワークフローデータと一緒にユーザアプリケーションデータベースには保管されません。ログデータベースに保管されます。これらのドキュメントを保管するには、ログを有効にする必要があります。

この項では、次のトピックについて説明します。

- 38 ページのセクション 2.3.1「ユーザ証明書の設定」
- 42 ページのセクション 2.3.2「アプリケーションサーバの環境設定」
- 43 ページのセクション 2.3.3「ログの環境設定」
- 43 ページのセクション 2.3.4「ユーザアプリケーションの設定」
- 43 ページのセクション 2.3.5「プロビジョニング要求定義の環境設定」

2.3.1 ユーザ証明書の設定

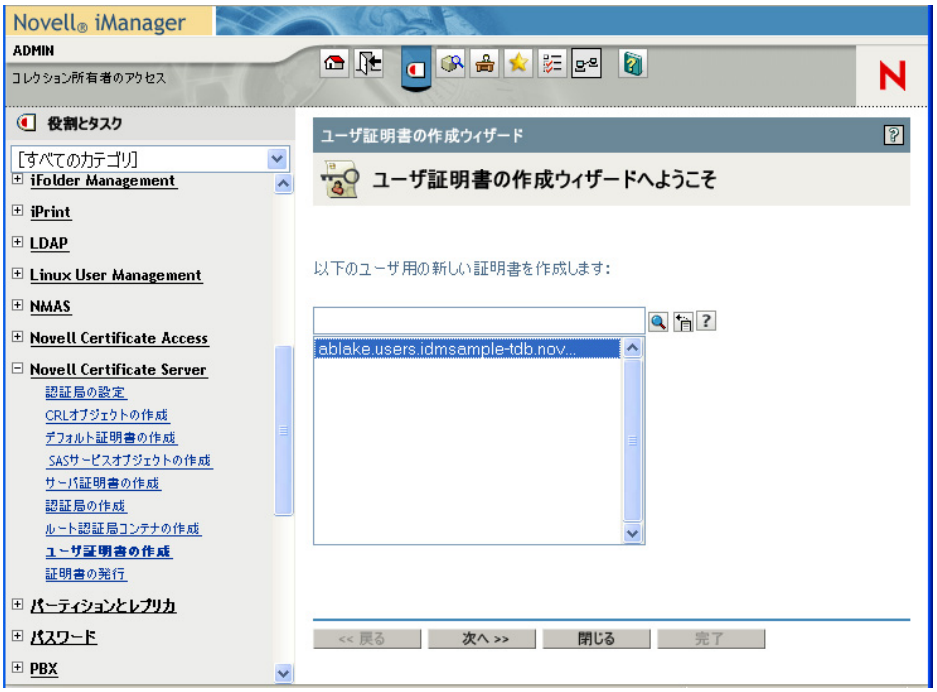
1 iManager を使ったユーザ証明書の作成

1a 管理者としてログインします。

1b *[Novell Certificate Server]* で *[Create User Certificate(ユーザ証明書の作成)]* を選択します。

1c 証明書を作成するユーザを選択して、*[次へ]* をクリックします。

オブジェクトセレクタまたはオブジェクト履歴を使ってユーザを選択できます。



1d サーバを選択して、証明書のニックネームを作成します。作成方法として *[カスタム]* を指定して、*[次へ]* をクリックします。

1e 要件に応じて、1024 ビットまたは 2048 ビットのキーサイズを死しますキータイプを *[署名]* に設定します。他の設定はそのままにして、*[次へ]* をクリックします。

1f デフォルトの環境設定を使用する場合、証明書パラメータはそのままにして *[次へ]* をクリックします。

証明書取り消しリスト (CRL) サポートを有効にする場合は、［カスタム］を選択して［CRL signing(CRL 署名)］チェックボックスを選択します。

CRL 環境設定の詳細は、Novell Certificate Server のマニュアルを参照してください。

1g ［*完了*］をクリックします。

1h ログアウトします。

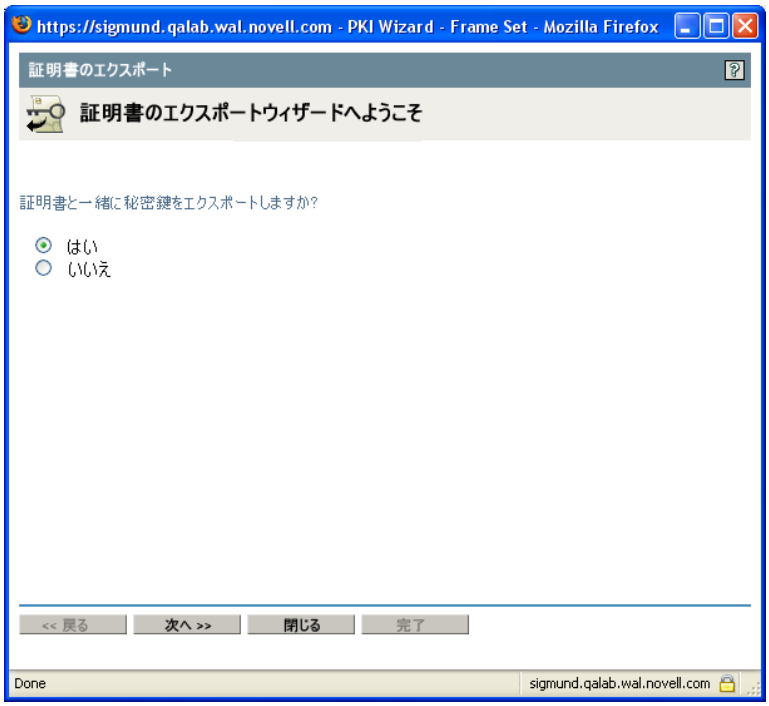
2 ユーザ証明書を、秘密鍵を含む PFX ファイルとして保存します。

2a 証明書をエクスポートするユーザとしてログインします。

2b *［Novell Certificate Access(Novell 証明書アクセス)］*で、*［View My Certificates (自分の証明書を表示)］*を選択します。

2c 証明書を選択して、*［エクスポート］*ボタンをクリックします。

2d 証明書のエクスポートウィザードで、*［はい］*をクリックして証明書と秘密鍵をエクスポートします。次に、［*次へ*］をクリックします。



2e 秘密鍵を保護するパスワードを入力して、*［次へ］*をクリックします。

2f カードリーダーがない場合は、*［Export the certificate into the browser(証明書をブラウザにエクスポート)］*を選択します。そうでない場合は、*［Save exported certificate to a file(エクスポートしたファイルを証明書に保存)］*リンクをクリックします。

後でブラウザにインポートすることもできます。別のブラウザにインポートする場合は、*［Save exported certificate to a file(エクスポートしたファイルを証明書に保存)］*をクリックします。

2g ファイルを開かないで保存する場合は、*［Save to Disk(ディスクに保存)］*をクリックします。

2h *［閉じる］*をクリックします。

3 スマートカードを使用している場合は、スマートカードリーダドライバをインストールしてください。

4 証明書情報をスマートカードに転送するために必要なソフトウェアをインストールします。たとえば、cryptovision(cv act sc/interface) が提供するスマートカードミドルウェアソフトウェアを入手したり、製品の評価版やマニュアルをダウンロードする場合は、http://www.cryptovision.com/idmdigsig.html を参照してください。

注 : cryptovision ミドルウェアソフトウェアは、バージョン 3.3 以上をインストールする必要があります。スマートカードに証明書情報を転送するには、管理ソフトウェアが必要です。Linux* では、cryptovision ソフトウェアはサポートされていません。

---

5 スマートカードにキーペア (証明書) をインポートする

スマートカードの代わりにブラウザ証明書サポートの使用を計画している場合は、上記の手順 3 ～ 5 をスキップすることができます。証明書は、iManager またはブラウザ証明書

管理ユーザインタフェースを使ってブラウザにインポートできます。cryptovision アプレットは、Windows 上で動作する Internet Explorer と Firefox* のみをサポートしています。

2.3.2 アプリケーションサーバの環境設定

アプリケーションサーバを環境設定するには、次の手順に従ってください。

1. JBoss を利用している場合は JBOSS_HOME/server/IDM/lib ディレクトリに、WebSphere を利用している場合は /IBM/WebSphere/AppServer/lib/ext ディレクトリに、次の JAR をコピーします。
   - dom.jar
   - xmldigsig.jar
   - xmlsec.jar

   dom.jar、xmldsig.jar、および xmlsec.jar は、http://java.sun.com からダウンロードできます。これらの JAR は、Web Services Developer Pack に同梱されています。

   cryptovision の場合は、SafXVerifier.jar も必要です。SafXVerifier.jar のダウンロードについては、http://www.cryptovision.com/idmdigsig.html (http://www.cryptovision.com/idmdigsig.html) を参照してください。
2. JBoss に展開するには、xmlsigner.war を JBOSS_HOME/server/IDM/deploy ディレクトリにコピーします。

   WebSphere に展開するには、WebSphere 管理コンソールを使って xmlsigner.war を展開します。

   xmlsigner.war のダウンロードの詳細は、http://www.cryptovision.com/idmdigsig.html (http://www.cryptovision.com/idmdigsig.html) を参照してください。
3. ルート認証局およびすべての中間証明書をエクスポートし (iManager を使用)、それを keytool コマンドを使ってシステムのローカル設定に指定されているキーストアにインポートします。

   JBoss の場合の例を次に示します。

   ```
   keytool -import -trustcacerts -file certFile
   ```

   certFile は、証明書ファイルへの完全修飾パスです。

   WebSphere の場合の例を次に示します。

   ```
   keytool -import -trustcacerts -file servercert.der -alias
   myserveralias -keystore trust.p12 -storetype PKCS12
   ```

   Novell Certificate Server をご利用の場合は、ルート認証局をエクスポートする必要はありません。
4. configupdate スクリプト (Windows の場合 configupdate.bat、Linux/Solaris の場合 configupdate.sh) を実行して、ユーザアプリケーション環境設定ユーティリティを開始します。
5. *[詳細オプションの表示]* をクリックします。
6. *[トラステッドキーストア]* で、*[トラステッドストアパス]* に証明書ファイルのパスを入力します。*[キーストアパスワード]* フィールドに、自分のパスワードを入力します。デフォルトのパスワードは、「changeit」です。

   トラステッドキーストアには、有効なデジタル署名に使用するすべてのトラステッド署名者の証明書が含まれます。

注 : JBoss の場合、Novell Certificate Server を使用している場合は、*［eDirectory 証明書］* の *［キーストアパス］* フィールドにある文字列 (例 :C:\Program Files\Java\jdk1.5.0_08\jre\lib\security\cacerts) を単にコピーして、それを *［トラステッドキーストア］* の *［トラステッドキーストアパス］* に貼り付けることができます。また、*［キーストアパスワード］* をコピーして、*［トラステッドストアパスワード］* フィールドに貼り付けることもできます。

WebSphere の場合、たとえばこの文字列は、「/opt/IBM/WebSphere/AppServer/profiles/AppSrv01/ config/cells/citgoNode01Cell/nodes/MyServerNode01/trust.p12」のようになります。

7 OCSP を使用している場合は、*［その他］* の *［OCSP URI］* フィールドに、OCSP の URI を入力します。この値は、信頼されている証明書のステータスをオンラインで更新する場合に使用されます。URI は、OCSP(Online Certificate Status Protocol) サーバのアクセスポイントを示します。

JBoss アプリケーションサーバ設定の詳細は、メーカーが提供する次のような資料をを参照してください。

- *JBoss Enterprise Application Platform 4.2.0 Getting Started Guide* (https://www.redhat.com/docs/manuals/jboss/jboss-eap-4.2/doc/Getting_Started.pdf)
- *JBoss Enterprise Application Platform 4.2.0 Configuration Guide* (https://www.redhat.com/docs/manuals/jboss/jboss-eap-4.2/doc/Server_Configuration_Guide.pdf)

2.3.3 ログの環境設定

デジタル署名のログを有効にするには、ログプラットフォームエージェントを設定する必要があります。Novell Audit または Sentinel にイベントをレポートするクライアントはすべて、プラットフォームエージェントが必要です。プラットフォームエージェントは、logevent 環境設定ファイルで設定できます。このファイルでは、プラットフォームエージェントが Novell Audit サーバと通信するために必要な情報を設定します。

重要 : デジタル署名を含むイベントを記録する場合、記録するデータを十分に処理できる値を LogMaxBigData に指定する必要があります。

ログの環境設定の詳細は、69 ページの第 3 章「ログのセットアップ」を参照してください。

2.3.4 ユーザアプリケーションの設定

ユーザアプリケーションのデジタル署名サポートを設定するには、ユーザアプリケーションの *［管理］* タブにある *［デジタル署名サービス］* ページを使用してください。詳細については、194 ページのセクション 8.3「デジタル署名サービスの環境設定」を参照してください。

2.3.5 プロビジョニング要求定義の環境設定

Identity Manager の Designer または iManager を使って、プロビジョニング要求定義のデジタル署名サポートを設定することができます。デジタル署名サポートの基本要件は、Designer を使用する場合でも、iManager を使用する場合でも同じです。

プロビジョニング要求定義でデジタル署名をサポートするように設定するには、次の事項が必要です。

1. プロビジョニング要求を開始するためにデジタル署名が必要かどうかを指定する。
2. ワークフロー内の各承認ステップでデジタル署名が必要かどうかを指定する。各承認ステップには複数の送信リンクがある場合もあるため、各リンクに対してデジタル署名が必要かどうかを指定する必要があります。

要求の開始または承認ステップの実行にデジタル署名が必要かどうかを指定したら、デジタル署名を必要とする各要求／承認ステップに対して次の事項を指定する必要があります。

表 2-2 *デジタル署名設定*

設定	説明
デジタル署名タイプ	デジタル署名がタイプとしてデータを使用するのか、またはフォームを使用するのかを指定します。 ◆ **データ：** ユーザ合意として、XML 署名を使用することを示します。［データ］を選択した場合、XML データが監査ログに記録されます。ユーザは署名を送信する前に、XML データをプレビューすることができます。 ◆ **フォーム：** 生成するデジタル署名宣言を含む PDF ドキュメントを使用することを示します。このドキュメントがユーザ合意として使用されます。ユーザは、要求の送信や承認前に、生成された PDF ドキュメントをプレビューできます。［フォーム］を選択した場合、PDF ドキュメント (XML 内にカプセル化) が監査ログに記録されます。 **警告：** デジタル署名したドキュメントを保持するには、Novell Audit(または Sentinel) を使用する必要があります。デジタル署名ドキュメントはワークフローデータと一緒にユーザアプリケーションデータベースには保管されません。ログデータベースに保管されます。これらのドキュメントを保管するには、ログを有効にする必要があります。
デジタル署名宣言	ユーザの署名を確認する、デジタル署名確認文字列を指定します。

Designer を使ったプロビジョニング要求定義の設定方法の詳細は、『*Identity Manager ユーザアプリケーション : 設計ガイド*』を参照してください。iManager を使ったプロビジョニング要求定義の設定方法の詳細は、299 ページの第 17 章「プロビジョニング要求定義の設定」を参照してください。

2.4 ユーザアプリケーションへの匿名／ゲストアクセスを有効にする

ユーザアプリケーションの Identity セルフサービス機能への匿名／ゲストアクセスを有効にするには、表 2-3 で説明している手順に従ってください。

表 2-3 *匿名アクセスの設定*

タスク	参照先
匿名アクセスに使用するゲストアカウントを決定します。	詳細については、45 ページの 「ゲストアカウントの作成」を参照してください。
ゲストユーザに対して、適切なアイデンティティボールト権限を割り当てます。	認証を受けない Web アプリケーションユーザに公開する機能に基づいて権限を定義します。検索、詳細、組織図、作成などのポートレットを公開することができます。また、ワークフローの開始を許可することもできます。このような場合、eDirectory のバインドと LDAP 操作の実行には、ゲストユーザアカウントが使用されます。
Identity セルフサービスタスクを実行する場合は、ゲストアクセス専用の新しいページとポートレットを作成します。	詳細については、207 ページのパート IV 「ポートレットリファレンス」を参照してください。
リソース要求を実行するには、リソース要求ポートレットを使用します。	詳細については、267 ページの第 14 章「リソース要求ポートレット」を参照してください。

2.4.1 ゲストアカウントの作成

ユーザアプリケーションでユーザ／ゲストアクセスをサポートするには、次の 2 種類の方法があります。以下の操作を行えます。

- 専用のユーザアカウントを作成する。匿名ユーザのアクティビティに必要な許可を設定します。このユーザがユーザコンテナ内のユーザの場合、ツリー検索時にこのゲストアカウントが返されます。これを防止するために、ゲストユーザはユーザコンテナ外に置いてください。
- eDirectory の [Public] オブジェクトに対応するパブリック LDAP ゲストアカウントを使用する。[Public] に対するデフォルトのアクセスは、ツリー全体に対する参照権です。このユーザに利用させるタスクに応じて、適切な許可を設定してください。匿名ユーザに利用させたくないタスクがある場合は、この方法は適切でない場合もあります。

ユーザアプリケーションでは、1 つのタイプの匿名ユーザしか指定できません。またこのユーザは、インストール時に指定する必要があります。インストールオプションを次に示します。

- **パブリック匿名アカウントの使用：** これは、LDAP ゲストアカウントを使用します。
- **LDAP ゲスト：** これは、専用のユーザアカウントです。

インストール完了後は、configupdate ユーティリティを使って、インストールの設定内容を変更することができます。

2.5 「パスワードを忘れた場合」の環境設定

ユーザアプリケーションには、パスワードを忘れた場合に備えてパスワードセルフサービス機能が用意されています。このサービスには、次の機能があります。

- 本人確認の回答メッセージを表示する
- パスワードのヒントを表示する
- パスワードの変更を許可する

企業ファイアウォール内のユーザは、「パスワードを忘れた場合」サービスを、ユーザアプリケーション WAR を通じてデフォルトで利用できます。

また、別個の「パスワードを忘れた場合」管理 WAR(IDMPwdMgt.WAR) を設定し、それをファイアウォール内外のシステムに展開することもできます。この WAR をファイアウォール外に展開することにより、リモートユーザに対してセキュリティ層を追加しながら「パスワードを忘れた場合」セルフサービスを提供できます。「パスワードを忘れた場合」WAR は、*外部パスワード WAR* と呼ばれることもあります。外部パスワード WAR の設定方法は、表 2-4 を参照してください。

IDMPwdMgt.WAR には、「パスワードを忘れた場合」セルフサービスソフトウェアとデフォルトのユーザアプリケーションテーマのみが含まれています。

表 2-4 *外部パスワード WAR を有効にする手順*

タスク	説明
ユーザアプリケーションをインストールします。インストール時には、ユーザアプリケーション環境設定パラメータの指定を要求するメッセージが表示されます。外部パスワード WAR を有効にするには、次の項目を指定します。 ◆ *外部パスワード WAR の使用* ◆ *パスワードを忘れた場合のリンク* ◆ *パスワードを忘れた場合の返信リンク* インストール後に configupdate ツールを使って、環境設定の内容を変更することもできます。	*[外部パスワード WAR の使用]* を指定すると、指定したインストールディレクトリに IDMPwdMgt.WAR が生成、インストールされます。 *[パスワードを忘れた場合のリンク]* には、外部パスワード WAR の場所を指定します。アプリケーションサーバのホストとセキュアポートを入れて指定します。たとえば、http://localhost:8080/ExternalPwd/jsps/pwdmgt/ForgotPassword.jsf のように指定します。インストールプログラムは、指定された場所に基づいて IDMPwdMgt.WAR の名前を変更します。 [*パスワードを忘れた場合の返信リンク*] には、外部パスワード WAR が (Web サービスを使って)、ユーザアプリケーションを呼び戻すために使用するパスを指定します。たとえば「https://*idmhost:sslport*/idm」のように指定します。 リンクの場所を変更する場合は、*[管理]* タブから設定を変更します。

タスク	説明
外部パスワード WAR をアプリケーションサーバに展開します。	外部パスワード WAR をアプリケーションサーバに展開する前に、アプリケーションサーバが SSL をサポートするように設定されていることを確認してください。参照先 35 ページのセクション 2.2.3「SSL を有効にする」. また、次のことをご確認ください。 ◆ 外部パスワードWARをファイアウォール外に展開する場合、アプリケーションサーバホスト間での通信を可能にするために、ファイアウォールの SSL ポートを開けてください。 ◆ 外部パスワード WAR を提供するアプリケーションサーバには、コアユーザアプリケーションを提供するアプリケーションサーバのサーバ証明書が必要になります。外部パスワード WAR を提供するアプリケーションサーバが使用する、JRE のキーストア (cacerts) にサーバ証明書をインポートするには、keytool インポートコマンドを使用します。keytool コマンドの構文を次に示します。 `keytool -import -file certname.cer -keystore cacerts -storepass changeit -alias uacerts`
外部パスワード WAR のテーマをカスタマイズしますか？	詳細については、112 ページの 「外部パスワード WAR 用テーマのカスタマイズ」を参照してください。

外部パスワード WAR の場所は、

configuration.AppDefs.AppConfig.driver.driverset に

<property>
<key>com.novell.pwdmgmt.login.PREF_FORGOT_PSWD_LINK_KEY</key>
<value>http://localhost:8080/ExternalPwd/jsps/pwdmgt/ForgotPassword.jsf</value> として保存されます。

戻り先の場所は、

configuration.AppDefs.AppConfig.driver.driverset に

<property>
<key>com.novell.pwdmgmt.login.PREF_FORGOT_PSWD_RETURN_LINK_KEY</key>
<value>https://localhost:8443/IDMProv</value>
</property> として保存されます。

戻り先の場所は、External WAR/WEB-INF/faces-managed-beans.xml の userAppURL プロパティに保存されます。次に例を示します。

<property-name>userAppURL</property-name>
<property-class>java.lang.String</property-class>
<value>https://localhost:8443/IDMProv</value>

2.5.1 外部パスワード WAR へのアクセス

ユーザは次のように指定して、ブラウザから直接外部パスワード WAR の *[パスワードを忘れた場合]* ページに移動できます。

http://localhost:8080/ *ExternalPwd*/jsps/pwdmgt/ForgotPassword.jsf 。

直接アクセスされた場合、外部パスワード WAR は、WEB-INF\faces-managed-beans.xml で次のエントリをチェックします。

```
<property-name>userAppURL</property-name>
<property-class>java.lang.String</property-class>
<value>https://151.155.254.69:8443/IDM</value>
```

外部パスワード WAR は、ユーザアプリケーション WAR 内の「パスワードを忘れた場合」機能を処理する Web サービスの呼び出しに、userAppURL エントリを使用します。

[パスワードを忘れた場合] ページにアクセスするには、*[ログイン]* ページで *[パスワードを忘れた場合]* リンクをクリックします。ユーザアプリケーションは、*[パスワードを忘れた場合のリンク]* に指定された値に基づいて、ユーザを外部パスワード WAR にリダイレクトします。外部パスワード WAR は、*[パスワードを忘れた場合の返信リンク]* に指定された値を使ってユーザアプリケーションを呼び戻します。

2.6 パフォー \'83\'7d ンスの調整

パフォーマンスの調整は複雑な課題です。Identity Manager ユーザアプリケーションは、さまざまな対話を行う幅広いテクノロジに依存しています。パフォーマンスの低下を招くような設定シナリオやユーザ対話シナリオをすべて予測することは不可能です。ただし、いくつかのサブシステムはベストプラクティスを実践することで、パフォー \'83\'7d ンスを向上できます。

詳細については、次の節を参照してください。

- 48 ページのセクション 2.6.1「ログ」
- 50 ページのセクション 2.6.2「識別ボールト」
- 51 ページのセクション 2.6.3「JVM」
- 51 ページのセクション 2.6.4「セッションタイムアウト値」
- 52 ページのセクション 2.6.5「JBoss のチューニング」
- 52 ページのセクション 2.6.6「ユーザアプリケーションのアイデンティティボールトへの接続へのセキュアソケットの使用」

2.6.1 ログ

ユーザアプリケーションでは、Novell Audit によるログと、オープンソースの Apache *log4j* フレームワークによるログが可能です。デフォルトでは Novell Audit によるログは無効になっています。これに対し、log4j によるファイルとコンソールのログはデフォルトで有効になっています。

注 : ログが可能なイベントの種類、およびログの有効 / 無効の切り替えについては、69 ページの第 3 章「ログのセットアップ」を参照してください。

---

log4j の環境設定は、次のファイルに保存されています :

- jboss-log4j.xml、このファイルはインストールディレクトリにあります (JBoss アプリケーションサーバを使用している場合)。
- log4j.xml、このファイルはユーザアプリケーション WAR 内にあります (JBoss 以外のアプリケーションサーバを使用している場合)。

jboss-log4j.xml ファイルの下部にある、次のエントリを探してください。

```
<root>
    <priority value="INFO" />
      <appender-ref ref="CONSOLE" />
      <appender-ref ref="FILE" />
  </root>
```

root に値を割り当てると、レベルが明示的に割り当てられていないログアペンダはすべて root レベル (この場合は INFO) を継承します。たとえば、デフォルトでは FILE アペンダにはしきい値レベルが割り当てられていないため、root のしきい値レベルを引き継ぎます。

log4j で使用されるログレベルは DEBUG、INFO、WARN、ERROR、および FATAL で、これは org.apache.log4j.Level クラスで定義されています。これらの設定を適切に使用しなかった場合、パフォー \'83\'7d ンスの面で問題が発生する可 \'94\'5c 性があります。

経験則から言うと、INFO や DEBUG は特定の問題をデバッグするときだけ使用してください。

ルートに含まれるアペンダに特定のしきい値レベルが設定されていない場合、デバッグを行うとき以外はしきい値を ERROR、WARN、または FATAL に設定する必要があります。

ログレベルが高いときのパフォーマンスは、メッセージの冗長性とはほとんど関係なく、log4j ではコンソールとファイルのログが同時書き込みに関与しているという単純な事実に影響されます。AsyncAppender クラスを使用できますが、このクラスを使用してもパフォーマンスの向上が保証されるわけではありません。この問題 (Apache log4j の周知の問題で、Identity Manager の問題ではありません) については、http://logging.apache.org/log4j/docs/api-1.2.8/org/apache/log4j/performance/Logging.html (http://logging.apache.org/log4j/docs/api-1.2.8/org/apache/log4j/performance/Logging.html) を参照してください。

ユーザアプリケーションのログ設定ファイルのデフォルトのレベルである INFO(前述) は、多くの環境で問題になりませんが、パフォーマンスが重要な環境では先ほどの jboss-log4j.xml のエントリを次のように変更する必要があります。

```
<root>
      <priority value="ERROR"/>
      <appender-ref ref="FILE"/>
</root>
```

つまり、CONSOLE を削除し、ログレベルを ERROR に設定します。完全にテストおよびデバッグされた運用環境では、INFO レベルでのログは必要ありません。また、CONSOLE のログを有効にしておく必要もありません。これらのログを無効にするとパフォー \'83\'7d ンスが大きく向上します。

log4j の詳細については、http://logging.apache.org/log4j/docs にある資料を参照してください。

Identity Manager で Novell Audit を使用する際の詳細については、『*Novell Identity Manager: 管理ガイド*』を参照してください。

2.6.2 識別ボールト

利用頻度の高いディレクトリサーバ環境では、LDAP クエリがボトルネックになる可能性があります。Novell eDirectory (Identity Manager のアイデンティティボールトのベース) は、オブジェクトが多数でも高いレベルのパフォーマンスを維持するために、頻繁に要求される情報を記録し、インデックスに保存します。インデックス化された属性を持つオブジェクトに対して複雑なクエリを実行すると、クエリは高速に返されます。

eDirectory は最初から次の属性がインデックス化されています。

```
Aliased Object Name
cn
dc
Equivalent to Me
extensionInfo
Given Name
GUID
ldapAttributeList
ldapClassList
Member
NLS: Common Certificate
Obituary
Reference
Revision
Surname
uniqueID
uniqueID_SS
```

Identity Manager をインストールすると、デフォルトのディレクトリスキーマが、ユーザアプリケーションに関する新しいオブジェクトクラスタイプと新しい属性で拡張されます。デフォルトでは、ユーザアプリケーション固有の属性はインデックス化されません。パフォーマンスを向上させるため、特に 5,000 以上のオブジェクトがユーザコンテナに含まれる場合には、こうした属性の一部 (また必要に応じて従来の LDAP 属性のいくつか) をインデックス化できます。

一般的には、定期的にクエリされる属性のみをインデックス化しますが、運用環境によりそのような属性は異なる可能性があります。どの属性が頻繁に使用されるかを見極める唯一の方法は、ランタイム時に述語統計を収集することですただし、収集プロセス自体がパフォーマンスを低下させてしまいます。

述語統計の収集プロセスの詳細については、『*eDirectory 管理ガイド*』を参照してください。このガイドはインデックス化についても詳しく解説しています。一般的には、次の作業が必要です。

- ConsoleOne® を使用して、該当する属性の述語統計の収集を開始する。
- システムに負荷をかける。
- 統計の収集を無効にして結果を分析する。
- インデックス化しておくと便利な各属性のインデックスを作成する。

インデックス化する属性がわかっている場合は、ConsoleOne を使用する必要はありません。インデックスを作成し管理するには、iManager で *[eDirectory の保守] > [インデックス]* の順にクリックします。たとえば、組織図のユーザが isManager 属性に基づいて検索することがわかっている場合は、その属性をインデックス化し、パフォーマンスが向上するかどうかを確かめることができます。

注 : ベストプラクティスとして、最低限 manager および isManager 属性をインデックス化することをお勧めします。

属性のインデックス化とパフォーマンスの詳細については、Peter Kuo と Jim Henderson 共著の『*Novell's Guide to Troubleshooting eDirectory*』(QUE Books, ISBN 0-7897-3146-0) の「Tuning eDirectory」を参照してください。

また、『*eDirectory 管理ガイド*』の、パフォーマンスのチューニングに関する項目も参照してください。

2.6.3 JVM

Java 仮想マシンに割り当てられるヒープメモリの量はパフォーマンスに影響することがあります。最小メモリ値や最大メモリ値の設定値が低すぎたり高すぎたりすると (「高すぎる」とはコンピュータの物理メモリより多いことを意味します)、ページファイルのスワッピングが過剰に発生する可能性があります。

JBoss サーバの場合、テキストエディタを使って [IDM]/jboss/bin/ にある run.conf または run.bat ファイル (前者は Linux の場合、後者は Windows の場合) を編集して、最大 JVM* サイズを設定できます。「*-Xmx*」を *128m* から *512m* またはそれ以上に増やします。ご使用の環境に最適な設定が見つかるまで、調整を繰り返さなければならない場合があります。

注 : JBoss および Tomcat のパフォーマンスチューニングに関するヒントは、http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossASTuningSliming (http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossASTuningSliming) を参照してください。

2.6.4 セッションタイムアウト値

セッションタイムアウト (ユーザが Web ブラウザのページを表示したままにしてから、サーバによってセッションタイムアウトの警告ダイアログが表示されるまでの時間) は、IDM.war アーカイブの web.xml ファイルで変更できます。この値は、アプリケーションが実行されるサーバおよび使用環境に合わせて調整する必要があります。一般にセッションタイムアウト値は、実用上問題のない範囲でできるだけ小さくする必要があります。5 分間のセッションタイムアウトを指定すれば、タイムアウト値を 10 分間に設定した場合と比べて倍の早さで、未使用のリソースが解放されます。こうすることによって Web アプリケーションのパフォーマンスとスケーラビリティが向上します。

セッションタイムアウトの値を調整する場合は、次の事項に留意してください。

- セッションタイムアウトの時間が長いと、短時間に大勢のユーザがログインした場合、JBoss サーバのメモリが不足する可能性があります。これは、開かれたセッションが多すぎれば、どのアプリケーションサーバでも起こり得ます。
- ユーザがユーザアプリケーションにログインすると、そのユーザの LDAP 接続が作成されてセッションにバインドされます。このため、開かれたセッションが多いほど、保持される LDAP 接続の数が多くなります。セッションタイムアウトが長いほど、これらの接続が開かれた状態で保持される時間も長くなってしまいます。LDAP サーバへの開かれた接続が多すぎると、それがアイドル状態であったとしても、システムのパフォーマンスが低下してしまいます。

- サーバおよび使用環境のJVMヒープおよびガーベージコレクション調整パラメータが最適化されているにもかかわらず、サーバでメモリ不足エラーが発生するようになったら、セッションタイムアウト値を低くしてみてください。

アプリケーションサーバによっては、セッションタイムアウト値を指定する方法が用意されている場合があります。代わりに、IDM.war アーカイブを開いて web.xml ファイルを探し、ファイルの次の部分を編集してセッションタイムアウト値を調整することもできます (数値の 20 はデフォルト値で、20 分を表します)。

```
<session-config>
    <session-timeout>20</session-timeout>
</session-config>
```

次に、ファイルとアーカイブを保存して、サーバを再起動します。

注 : Web アーカイブファイルの手動編集は、Java Web アプリケーションの開発と展開に熟練したユーザが行ってください。

2.6.5 JBoss のチューニング

デフォルトでは、JBoss 配備スキャナは 5 秒間隔で実行されます。運用環境では、一般に JBoss 配備スキャナは不要であり、パフォーマンスに悪影響を及ぼすおそれがあります。配備スキャナの実行頻度を減らすか、または配備スキャナを完全に停止させることを検討してください。配備スキャナの環境設定については、『ConfiguringTheDeploymentScannerInConfjbossSystem (http://wiki.jboss.org/wiki/Wiki.jsp?page=ConfiguringTheDeploymentScannerInConfjbossSystem.xml)』を参照してください。

運用環境における JBoss チューニングの詳細は、『JBossASTuningSliming (http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossASTuningSliming)』を参照してください。

2.6.6 ユーザアプリケーションのアイデンティティボールトへの接続へのセキュアソケットの使用

デフォルトでは、ユーザアプリケーションサーバとアイデンティティボールト間の通信には、セキュアソケットが使用されます。ただし、環境によっては、すべての通信を保護する必要がない場合もあります。たとえば、ユーザアプリケーションサーバとアイデンティティボールトサーバが独立したネットワーク内にあり、外部から利用できるポートが HTTP ポートだけの場合は、これらのサーバ間の通信の一部を保護しない通信にできることがあります。アプリケーションの一部の機能では、セキュアな接続を使用しないように設定されていても、*常に*セキュアな接続を使用するものがあります (例 : パスワードの変更など)。セキュアな接続を無効にすると (特にユーザ接続)、パフォーマンスとスケーラビリティを大幅に向上することができます。同時に多数のユーザがログインするような環境で、ネットワーク設定でユーザアプリケーションサーバとアイデンティティボールトサーバ間の通信が保護されている場合、ユーザ接続の保護を無効にすると、処理できる同時ログイン数が大幅に増加します。この選択肢は、環境内のスケーラビリティ／パフォーマンスに関する問題の原因が明白で、新たな eDirectory サーバを追加することができない場合にのみ実行することをお勧めします。

また、管理者接続の保護を無効にすることもできます。このようなの接続は、ユーザ資格情報を必要としない、アイデンティティボールトサーバへの一般的なクエリに対して使用します。これらの接続は、ラウンドロビン方式でプールされ、使用されます。セキュアな

接続経由のバインドは、アプリケーションのスタートアップ時に 1 回のみ行われます (または、後ほど接続が応答不能になった場合にも)。そのため、ユーザ接続に関するスケーラビリティの問題にはなりません。ただし、終端間で行われる暗号化と複号化にかかる時間がオーバヘッドになってしまいます。特にパフォーマンスを向上する必要がない限り、デフォルトの設定を使用することをお勧めします。

管理者接続とユーザ接続の保護を解除する場合は、ユーザアプリケーションと iManager の両方で設定を変更する必要があります。管理者接続とユーザ接続の保護を解除する方法については、次の手順を参照してください。

- 53 ページの 「ユーザアプリケーション環境設定ツールを使ったセキュアな接続の無効化」
- 53 ページの 「iManager を使ったセキュアな接続の無効化」

ユーザアプリケーション環境設定ツールを使ったセキュアな接続の無効化

ユーザアプリケーションで管理者接続とユーザ接続の保護を無効にする

1. ユーザアプリケーションディレクトリにある configupdate スクリプトを、次のように実行します。
   - Linux: 次のように指定して、configupdate.sh を実行します。
     `./configupdate.sh`
   - Windows:configuupdate.bat を実行します。

   ユーザアプリケーション環境設定ユーティリティが起動します。
2. *［セキュアな管理者接続］* および *［セキュアなユーザ接続］* の選択を解除します。

3. *［OK］* をクリックします。

iManager を使ったセキュアな接続の無効化

iManager または ConsoleOne を使って eDirectory への管理者接続とユーザ接続に対してセキュア LDAP(LDAPS) 接続を無効にする

1. eDirectory ツリーにログインします。
2. *LDAP* グループオブジェクトに移動して、プロパティを表示します。
3. *［一般］* をクリックします。
4. *［パスワードとの単純バインドに TLS を必要とする］* の選択を解除します。

注 : マルチサーバ eDirectory ツリーで、LDAP グループの TLS を婿にすると、すべてのサーバから TLS 要求が削除されます。ツリー内の各サーバ個別に TLS 要求を混在させる場合は、各サーバの TLS 要求を有効にする必要があります。

2.7 クラスタリング

この項では、次のトピックについて説明します。

2.7.1 アプリケーションサーバのクラスタリング

クラスタとは、一連のサービスを提供するアプリケーションサーバノードの集まりです。クラスタの目的は、アプリケーションのパフォーマンスと信頼性を高めることにあります。一般に、クラスタはエンタープライズアプリケーションに次の 3 つの利点をもたらします。

- 高可用性
- スケーラビリティ (容量の増加)
- 負荷分散

高可用性とは、アプリケーションの信頼性が高く、展開されている間高い割合で使用できることを意味します。クラスタでは同じアプリケーションがすべてのノードで実行されるため、高可用性が実現します。1 つのノードでエラーが発生しても、他のノードではアプリケーションが引き続き実行されています。Identity Manager ユーザアプリケーションをクラスタで実行すると、高可用性の利点を享受できます。また、Identity Manager ユーザアプリケーションは、HTTP セッションのレプリケーションとセッションのフェイルオーバーをサポートしています。つまり、ノードでセッション処理中にノードにエラーが発生した場合、そのセッションは中断されることなく、同じクラスタ内の別のサーバ上で再開されます。

JBoss クラスタの詳細は、JBoss の wiki ページ (http://wiki.jboss.org/wiki/Wiki.jsp?page=JBossHA) にある高可用性とクラスタリングサービスに関する項目を参照してください。

JGroups クラスタグループ

JGroups 通信モジュールはグループ間の通信を提供し、これによって共通の名前、マルチキャストアドレス、およびマルチキャストポートを共有します。JGroups は JBoss と同時にインストールされます。JBoss がなくても使用できます。クラスタ環境におけるキャッシュをサポートするために、ユーザアプリケーションのユーザアプリケーション WAR には、JGroups モジュールが用意されています。

JBoss グループのクラスタグループ

JBoss クラスタは、JGroups 通信モジュールに基づいています。クラスタ化された JBoss サーバをインストールすると、クラスタの管理に使用する複数の JGroups グループクラス

タ環境設定が JBoss によって定義されます。この中の 1 つは *DefaultPartition* と呼ばれ、/deploy/cluster-service.xml に定義されています。このクラスタグループは、中核となるクラスタリングサービスを提供します。また、*Tomcat-Cluster* というクラスタグループも定義されます。この環境設定は、/deploy/jboss-web-cluster.sar/META-INF/jboss-service.xml に定義されています。このファイルは、JBoss 内で動作する Web サーバに対して、セッションのレプリケーションサポートを提供します。JBoss には、ejb3 サービスを管理するためのクラスタグループ環境設定も用意されています。

ユーザアプリケーションのクラスタグループ

Identity Manager ユーザアプリケーションは、JBoss/Websphere クラスタ環境内のユーザアプリケーションキャッシュを調整するために、別のクラスタグループを使用します。

ユーザアプリケーションのクラスタグループは、2 つの JBoss クラスタグループとは独立しており、データのやり取りなどは行われません。ユーザアプリケーションのクラスタグループと 2 つの JBoss グループは、異なるグループ名、マルチキャストアドレス、およびマルチキャストポートをデフォルトで使用するため、設定し直す必要はありません。

デフォルトでは、このクラスタグループは UUID 名を使用します。そのため、ユーザがサーバに追加する他のクラスタグループ名との競合リスクを最小限に抑えられます。デフォルト名は「c373e901aba5e8ee9966444553544200」です。デフォルトでは、このグループはマルチキャストアドレス 228.8.8.8 を使用し、ポート 45654 上で動作します。このクラスタの設定は JBoss サービスファイルを使用していません。その代わり、設定はディレクトリに存在し、ユーザアプリケーションの管理機能で設定できます。JGroups および JBoss のクラスタリングに精通しているユーザは、このインタフェースを使用してユーザアプリケーションのクラスタ設定を調整できます。クラスタ設定の変更をサーバノードに適用するには、そのサーバノードを再起動する必要があります。

ユーザアプリケーションのクラスタグループの設定は、ディレクトリ設定を共有するすべての Identity Manager アプリケーションと共有されます。ユーザアプリケーションの管理インタフェースにあるローカル設定オプションは、管理者がクラスタからノードを削除したり、クラスタ内のサーバのメンバーシップを変更したりできるようにする目的で用意されています。たとえば、グローバルでクラスタリングを無効にしてから、ディレクトリ設定を共有するサーバのサブセットに対してローカルでクラスタリングを有効にすることができます。

2.7.2 ユーザアプリケーションをインストールする前に行う作業

この節では、ユーザアプリケーションをインストールする前の注意事項と、インストール前に行う必要がある作業について説明します。

この項では、次のトピックについて説明します。

- 55 ページの 「同じネットワーク内の複数のクラスタについて」
- 56 ページの 「アプリケーションサーバ時刻の同期化」
- 56 ページの 「単一クラスタ内で同じブラウザの複数のタブからのログインの防止」
- 56 ページの 「ユーザアプリケーションデータベースについて」

同じネットワーク内の複数のクラスタについて

ネットワーク上で複数のクラスタが動作している場合、パフォーマンスの低下や誤動作を防止するために、クラスタを分離する必要があります。分離するには、クラスタごとに別

のパーティション名、マルチキャストアドレス、マルチキャストポートを使用します。同じネットワーク上に複数のクラスタがない場合でも、クラスタにはデフォルトのパーティション名を使用する代わりに、一意のパーティション名を指定することをお勧めします。

重要事項を次に示します。

- クラスタには、一意のクラスタパーティション名とマルチキャストアドレスが必要です。

  JBoss の場合、クラスタパーティション名とマルチキャストアドレスを指定するには、ユーザアプリケーションに同梱されている JBoss 起動スクリプト (start-jboss.bat(Windows) または start-jboss.sh(Linux)) を使用します。-D フラグを使って JBoss を起動するように設定し、システムプロパティの jboss.partition.name と jboss.partition.udpGroup を設定するように、サーバの JBoss 起動スクリプトを変更する必要があります (59 ページの 「ワークフローエンジンの環境設定」を参照)。.

- クラスタは、一意のマルチキャストポートを使用する必要があります。

  JBoss の場合、JBoss サーバの deploy\cluster-services.xml ファイルにある mcast_port 属性を編集し、使用するポートを指定します。

JBoss の場合、単一ネットワーク上での複数クラスタの運用に関する情報を、ブラウザを使って Two Clusters Same Network (http://wiki.jboss.org/wiki/Wiki.jsp?page=TwoClustersSameNetwork)(同一ネットワーク上の 2 つのクラスタ) から参照することができます。

アプリケーションサーバ時刻の同期化

ユーザアプリケーションクラスタ内のサーバの時刻は同期化する必要があります。サーバの時刻を同期化しないと、セッションタイムアウトが早期に発生し、HTTP セッションのフェイルオーバーが正しく機能しない可能性があります。時刻を同期化するには、さまざまな方法があります。使用する方法は、組織のニーズによって異なります。方法の 1 つとして、ネットワークタイムプロトコルの NTP を使用することが考えられます。時刻同期に xNTP プロトコルを使用する方法については、『Time Synchronization using Extended Network Time Protocol (xntp) (http://www.novell.com/coolsolutions/trench/15650.html)』を参照してください。

単一クラスタ内で同じブラウザの複数のタブからのログインの防止

あるブラウザで複数のタブを使ってログインしたり、同じホストで複数のブラウザセッションを使ってログインすることはお勧めいたしません。一部のブラウザはすべてのタブとプロセス間で cookies を共有します。そのため、複数ログインを行う他 HTTP セッションのフェイルオーバー時に問題が発生する可能性があります (さらに、1 台のコンピュータを複数ユーザが共有している場合、予期しない認証動作が発生する危険性もあります)。

ユーザアプリケーションデータベースについて

ユーザアプリケーションのインストールプログラムを使ってユーザアプリケーションをインストールする場合、使用するデータベースを指定します (例 :MySQL、Oracle、または Microsoft SQL Server など)。このデータベースは、ユーザアプリケーションデータと環境設定情報の保管に用いられます。

クラスタ環境にユーザアプリケーションをインストールする場合、JBoss クラスタ内のすべてのノードが同じインスタンスのデータベースにアクセスしなければなりません。ユー

ザアプリケーションは標準の JDBC コールを使用してデータベースのアクセスや更新を行います。ユーザアプリケーションは、JNDI ツリーにバインドされた JDBC データソースを使用してデータベースへの接続を開きます。

ユーザアプリケーションインストールプログラムを使って JBoss クラスタにユーザアプリケーションをインストールする場合は、データソースがインストールされます。インストールプログラムは IDM-ds.xml と言う名前のデータソースファイルを作成し、このファイルを展開用のディレクトリ (例 :server/IDM/deploy) に格納します。また、インストール時に指定されたデータベースに対応する適切な JDBC ドライバを、lib ディレクトリに格納します (例 :/server/IDM/lib)。クラスタ環境におけるユーザアプリケーションデータベースの設定方法の詳細は、58 ページの 「ユーザアプリケーションデータベースの指定」を参照してください。

注 : MySQL におけるデフォルトの最大接続数は 100 です。この値は、クラスタ内のワークフロー要求を処理するには小さすぎる場合があります。この値が小さすぎる場合、次の例外が発生することがあります。? ●

```
(java.sql.SQLException: Data source rejected establishment of
connection, message from server: "Too many connections.")
```

最大接続数を増やすには、my.cnf にある max_connections 変数の値を 100 以上に設定してください。

2.7.3 JBoss クラスタへのユーザアプリケーションのインストール

クラスタにユーザアプリケーションをインストールする場合は、ユーザアプリケーションインストールプログラムを使ってクラスタ内の各ノードにユーザアプリケーションをインストールします (『*Identity Manager 3.5 インストールガイド*』のユーザアプリケーションのインストール方法に関する項目を参照)。この節では、クラスタにユーザアプリケーションをインストールする場合の注意事項について説明しています。

この項では、次のトピックについて説明します。

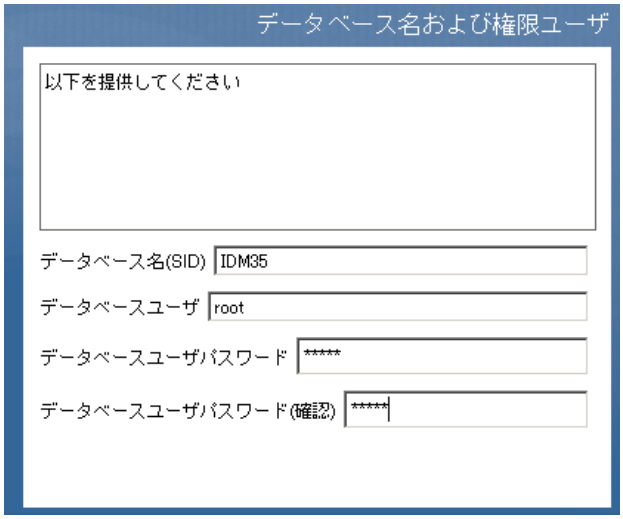
サーバ環境設定について

JBoss には、すぐに利用できる 3 種類のサーバ環境設定 *minimal*、*default*、および *all* が用意されています。クラスタリングは *all* の設定でのみ有効になります。/deploy フォルダにある cluster-service.xml ファイルには、デフォルトのクラスタパーティションの設定が記述されています。ユーザアプリケーションのインストール時に、インストールプログラムでクラスタにインストールするよう指定すると、インストールプログラムはすべての設定のコピーを作成し、そのコピーに IDM という名前を付けます (これはデフォルト設定です。インストールプログラムを使用して名前を変更できます)。それからこの設定にユーザアプリケーションをインストールします。

ユーザアプリケーションデータベースの指定

JBoss クラスタのノードはすべて、同じデータベースインスタンスにアクセスする必要があります。ユーザアプリケーションインストールプログラムを使用する場合、データベース名、ホスト、およびポートの指定を要求するメッセージが表示されます。

図 2-3 *データベースのホストとポートの指定*

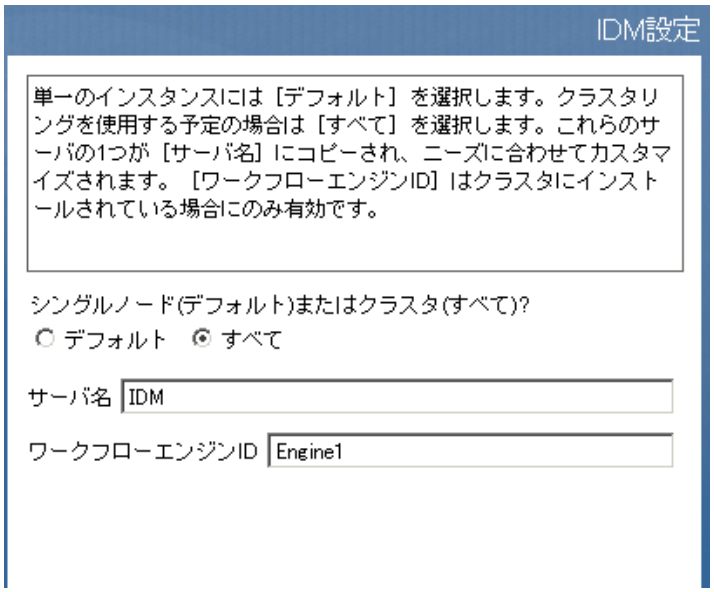


クラスタノードにユーザアプリケーションをインストールする際には、毎回同じデータベースパラメータを指定してください。

クラスタ (すべて) オプションの選択

ユーザアプリケーションインストールプログラムを使用する場合、IDM 環境設定の指定を要求するメッセージが表示されます。

図 2-4 *クラスタ (すべて) オプションとエンジン ID の指定*



[クラスタ (すべて)] オプションを選択します。

ワークフローエンジンの環境設定

ワークフローエンジンのクラスタリングは、ユーザアプリケーションのキャッシュフレームワークとは無関係に動作します。クラスタ環境でワークフローエンジンを正常に動作させるには、いくつかの手順を実行する必要があります。

- クラスタ内のサーバはすべて同じデータベースをポイントしている必要があります。

  ユーザアプリケーションインストールプログラムを使ってクラスタにユーザアプリケーションをインストールする場合 (57 ページの 「JBoss クラスタへのユーザアプリケーションのインストール」を参照)、ユーザアプリケーション用のデータベースをインストールしたサーバの IP アドレスとホスト名を指定します。

- クラスタ内の各サーバは、一意のエンジン ID で起動する必要があります。

  そのためには、サーバの起動時に com.novell.afw.wf.engine-id システムプロパティを設定します。たとえば、JBoss を起動して、サーバのワークフローエンジンにエンジン ID として ENGINE1 を割り当てる場合は、次のコ \'83\'7d ンドを使用します。

  ```
  run.sh -Dcom.novell.afw.wf.engine-id=ENGINE1 (Linux)
  run.bat -Dcom.novell.afw.wf.engine-id=ENGINE1 (Windows)
  ```

このシステムプロパティの設定と、他のシステムプロパティの設定を組み合わせることもできます (59 ページの 「JBoss 起動スクリプト中の JBoss システムプロパティの設定」を参照)。

実行中のワークフローの管理については、65 ページのセクション「クラスタ内のワークフローの管理」を参照してください。

JBoss 起動スクリプト中の JBoss システムプロパティの設定

クラスタ内の各サーバは、同じパーティション名とパーティション UDP グループを使って起動する必要があります (55 ページの 「同じネットワーク内の複数のクラスタについて」を参照)。クラスタ内の各サーバは、一意のエンジン ID を使用する必要があります (59 ページの 「ワークフローエンジンの環境設定」を参照)。

JBoss 起動スクリプト (start-jboss.bat(Windows)、start-jboss.sh(Linux)) を変更して、これらのシステムプロパティを指定することができます。このスクリプトは、ユーザアプリケーションを保存しているディレクトリにあります。たとえば、パーティション名「Example_Partition」、UDP グループ「228.3.2.1」、エンジン ID「Engine1」でサーバを起動するには、次の start-jboss スクリプトを追加します。

```
start run.bat -c IDM -Djboss.partition.name=Example_Partition -
Djboss.partition.udpGroup=228.3.2.1 -Dcom.novell.afw.wf.engine-
id=Engine1
```

クラスタ内の各ユーザアプリケーションでの同じマスタキーの使用

Identity Manager ユーザアプリケーションは、重要なデータを運号化します (37 ページのセクション 2.2.7「重要なユーザアプリケーションデータの暗号化」を参照)。マスタキーは、暗号化データにアクセスするために用いられます。クラスタ内のすべてのユーザアプリケーションは、同じマスタキーを使用する必要があります。クラスタ内のすべてのユー

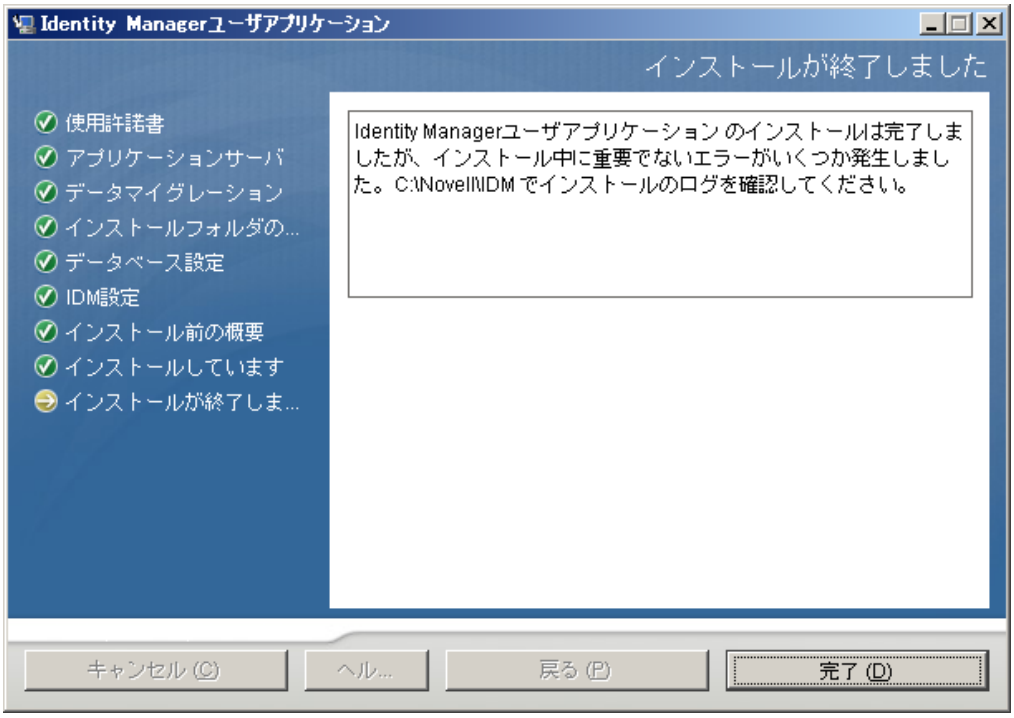
ザアプリケーションが同じマスタキーを使用するようにするには、これらの手順に従ってください。

1 ユーザアプリケーションインストールプログラムを使って、クラスタ内の最初のノードにユーザアプリケーションをインストールします。

ユーザアプリケーションインストールプログラムの使用については、『*Identity Manager 3.5 インストールガイド*』のユーザアプリケーションのインストールに関する項目を参照してください。

ユーザアプリケーションインストールプログラムを使って、クラスタ内の最初のユーザアプリケーションをインストールする場合、インストールの最後にユーザアプリケーションの新しいマスタキーが表示されます。

図 2-5 *マスタキー*

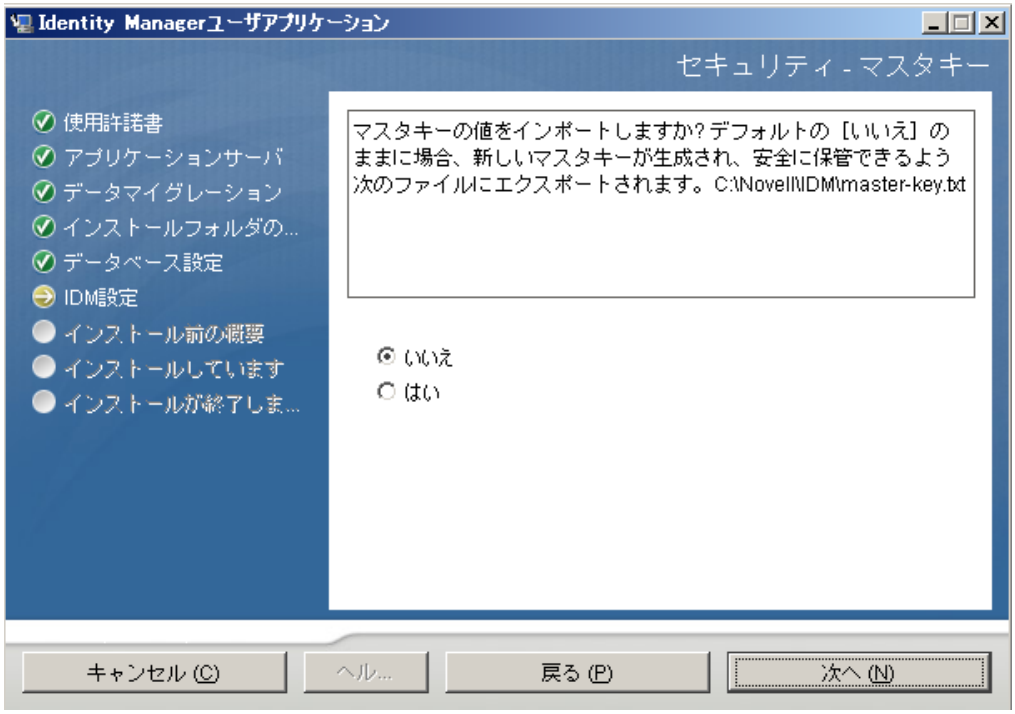


画面の指示に従って、マスタキーをテキストファイルに保存します。

2 ユーザアプリケーションインストールプログラムを使って、クラスタ内の他のノードにユーザアプリケーションをインストールします。

クラスタ内の他のノードにユーザアプリケーションをインストール際には、マスタキーをインポートするためのページが表示されます。

図 2-6 ユーザアプリケーションインストールプログラムへのマスタキーの貼り付け



3 60 ページのステップ 1 でテキストファイルに保存したマスタキーを、インポートします。

ユーザアプリケーションクラスタグループの開始

クラスタ内にユーザアプリケーションをインストールし終わったら、ユーザアプリケーションのクラスタ設定でクラスタを有効にする必要があります。

1 クラスタ内の最初のユーザアプリケーションを起動します。
2 ユーザアプリケーションの管理者としてログインします。
   まだ、他のサーバは起動しないでください。
3 *［管理］* をクリックします。
   ユーザアプリケーションに、アプリケーション環境設定ポータルが表示されます。
4 *［キャッシング］* をクリックします。

   ［キャッシュマネージャー］ ページが表示されます。
5 *［有効なクラスタ］* プロパティで、*［True］* を選択します。
6 ［*保存*］ をクリックします。
7 サーバを再起動します。

8 ローカル設定を使用する場合 (64 ページの 「ユーザアプリケーションクラスタグループのキャッシングの環境設定」を参照)、クラスタ内の各サーバでこの手順を繰り返します。

2.7.4 WebSphere クラスタへのユーザアプリケーションのインストール

この節では、WebSphere クラスタへのユーザアプリケーションのインストールと起動について説明します。この節の説明は、WebSphere アプリケーションサーバを熟知している方を前提にしています。

1 次のリンクなどを含めメーカーの指示に従って、WebSphere アプリケーションサーバとクラスタをインストール、環境設定します。

アプリケーションサーバの設定の詳細は、次の各項目を参照してください。

- IBM® WebSphere Application Server Library (https://www-306.ibm.com/software/webservers/appserv/was/library/library60.html). *［6.1］*タブを選択し、*［Show(表示)］*を選択して、*［WebSphere Application Server - distributed platforms］*のマニュアルを表示します。
- WebSphere Application Server, Version 6.1 Information Center (http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp)。ロードマップについては、*［Network Deployment: (Distributed platforms and Windows), Version 6.1］* > *［Installing your application serving environment］* > *［Distributed operating systems］* > *［Installing the product and additional software］* > *［Roadmap: Installing Network Deployment］*を参照してください。
- IBM WebSphere Extended Deployment, Version 6.1 Information Center (http://publib.boulder.ibm.com/infocenter/wxdinfo/v6r1/index.jsp)

2 メーカーの指示に従って、データベースをインストール、作成してください。データベースで UTF-8 を有効にします。

3 WebSphere サーバで、データベースドライバを追加、環境設定します。

4 JDBC プロバイダを作成します。

5 リレーショナルデータベースのデータソースを作成します。

6 ユーザアプリケーションインストールプログラムを実行し、WAS コンソールシステムにユーザアプリケーションをインストール、環境設定します。詳細は、『*Identity Manager 3.5.1 インストールガイド*』の第 5 章にある、ユーザアプリケーションのインストールに関する項目を参照してください。

インストールプログラムは、インストール時に指定されたディレクトリに sys-configuration-xmldata.xml ファイルを書き込みます。

7 インストール後は、『*Identity Manager 3.5.1 インストールガイド*』で説明されているように、WAS コンソールでクラスタ内の各ユーザアプリケーションサーバに対して新しい JVM カスタムプロパティを作成します。カスタムプロパティ com.novell.afw.wf.engine-id に名前を指定して、一意の値を指定します。各アプリケーションサーバがワークフローエンジンを実行します。これらの各エンジンには、一意のエンジン ID が必要です。

8 WebSphere キーストアに、ディレクトリサーバ認証局をインポートします。

9 WebSphere 管理コンソールから、IDM WAR ファイルを展開します。

10 アプリケーションを開始します。展開中に指定したコンテキストを使用してユーザアプリケーションにアクセスします。WebSphere 上の Web コンテナのデフォルトポートは 9080 です。または、セキュアポートの場合は 9443 です。URL は、次のようになります。

http:// <*server*>:9080/IDMProv

2.7.5 ユーザアプリケーションのインストール後に行う作業

この節では、ユーザアプリケーションのインストール後に行うユーザアプリケーションクラスタ環境設定に関する作業を説明します。

この項では、次のトピックについて説明します。

- 63 ページの 「クラスタリング用のユーザアプリケーションドライバの環境設定」
- 64 ページの 「ユーザアプリケーションクラスタグループのキャッシングの環境設定」
- 64 ページの 「クラスタ内のログの設定」
- 65 ページの 「クラスタ内のワークフローの管理」

クラスタリング用のユーザアプリケーションドライバの環境設定

クラスタリングでは、複数のユーザアプリケーションで同じユーザアプリケーションドライバが使用されます。これは、クラスタリングの場合のみです。ユーザアプリケーション用ドライバには、アプリケーション固有のさまざまな情報 (例 : ワークフロー環境設定情報、クラスタ情報) が保持されています。したがって、ユーザアプリケーション用ドライバの 1 つのインスタンスを複数のアプリケーション間で共有すべきではありません。

ユーザアプリケーションでは、アプリケーション環境を制御および設定するためのアプリケーション固有データがドライバに保持されます。たとえば、JBoss アプリケーションサーバのクラスタ情報や、ワークフローエンジン環境設定情報などが保持されます。ユーザアプリケーション用ドライバの 1 つのインスタンスを共有するユーザアプリケーションは、同じ JBoss クラスタの一部です。

クラスタで、ユーザアプリケーションドライバは、クラスタのディスパッチャまたはロードバランサのホスト名または IP アドレスを使用するように設定する必要があります。ユーザアプリケーションドライバは、ユーザアプリケーションのインストール時に作成します (『*Novell Identity Manager インストールガイド*』を参照)。ユーザアプリケーションドライバは、iManager を使って環境設定します。

1 識別 \'83\'7bールトを管理する iManager のインスタンスにログインします。
2 iManager のナビゲーションフレームにある［*Identity Manager*］ノードをクリックします。
3 *［Identity Manager の概要］*をクリックします。
4 ユーザアプリケーションドライバのドライバセットを含む Identity Manager の概要を表示するには、検索ページを使用します。
5 ドライバアイコンの右上隅にある円形のステータスインジケータをクリックします。

ドライバの起動と停止、およびドライバのプロパティの編集に関するコマンドが含まれたメニューが表示されます。

6 *[プロパティの編集]* をクリックします。

7 *[ドライバパラメータ]* セクションで、*[ホスト]* パラメータをディスパッチャのホスト名または IP アドレスに変更します。

8 *[OK]* をクリックします。

ユーザアプリケーションクラスタグループのキャッシングの環境設定

JGroups および JBoss クラスタリングに精通しているユーザは、ユーザアプリケーションの管理ユーザインタフェースを使用してクラスタグループのキャッシング設定を変更できます (96 ページの 「クラスタのキャッシュ設定」を参照してください)。クラスタ設定の変更をサーバノードに適用するには、そのサーバノードを再起動する必要があります。

たいていの場合、クラスタの設定時にはグローバル設定を行います。ただし、TCP を使用する必要がある場合、グローバル設定では問題が発生してしまいます。サーバの IP アドレスは、各サーバの JGroups 初期化文字列に指定する必要があります。*[クラスタのプロパティ:]* の *[ローカルの有効化]* を選択して、次に *[ローカル]* フィールドに JGroups 初期化文字列を入力して、ローカル設定を使って JGroups 初期化文字列を指定することができます。JGroups の TCP プロトコルスタックに関する作業例については、『JGroupsStackTCP (http://wiki.jboss.org/wiki/Wiki.jsp?page=JGroupsStackTCP)』を参照してください。

警告 : ローカル設定を指定して、JGroups 初期化文字列に誤った設定を入力すると、キャッシュクラスタ機能が開始されないことがあります。JGroups の正しい設定方法とプロトコルスタックを理解していない限り、ローカル設定は使用しないでください。

代わりに、*[クラスタプロパティ]* のグローバル設定にトークン (例 : “IDM_HOST_ADDR”) を追加することもできます。次に、クラスタ内の各サーバの hosts ファイルを編集して、サーバの IP アドレスを指定することができます。

クラスタ内のログの設定

この節では、クラスタ内でのログに関するヒントを説明しています。WebSphere に関するヒントはありません。

- 65 ページの 「JBoss のログ」
- 65 ページの 「ユーザアプリケーションのログ」

JBoss のログ

クラスタ内でログを記録するように、JBoss を設定することができます。クラスタのログを有効にするには、JBoss サーバ設定の \conf ディレクトリ (例 :\server\IDM\conf) にある jboss-log4j.xml 設定ファイルを編集し、ファイルの最後の方にある次のようなセクションをアンコメントする必要があります。

```
<!--  Clustering logging
  -->
- <!--
 Uncomment the following to redirect the org.jgroups and
      org.jboss.ha categories to a cluster.log file.
   <appender name="CLUSTER"
class="org.jboss.logging.appender.RollingFileAppender">
     <errorHandler
class="org.jboss.logging.util.OnlyOnceErrorHandler"/>
     <param name="File" value="${jboss.server.home.dir}/log
cluster.log"/>
     <param name="Append" value="false"/>
     <param name="MaxFileSize" value="500KB"/>
     <param name="MaxBackupIndex" value="1"/>
     <layout class="org.apache.log4j.PatternLayout">
       <param name="ConversionPattern" value="%d %-5p [%c] %m%n"/>
     </layout>
   </appender>
   <category name="org.jgroups">
     <priority value="DEBUG" />
     <appender-ref ref="CLUSTER"/>
   </category>
   <category name="org.jboss.ha">
     <priority value="DEBUG" />
     <appender-ref ref="CLUSTER"/>
   </category>
  -->
```

cluster.log ファイルは、JBoss サーバ設定の log ディレクトリ (例 : \server\IDM\log) にあります。

ユーザアプリケーションのログ

ユーザアプリケーションのログ記録設定 (101 ページのセクション 5.1.4「ログの設定」を参照) は、同じクラスタ内の他のサーバには反映されません。たとえば、あるクラスタ内のあるサーバ上で、［管理］タブの［ログ］サブタブで com.novell.afw.portal.aggregation に対するログ記録レベルとして［トレース］を選択した場合、この設定情報はそのクラスタ内の他のサーバに反映されません。クラスタ内の各サーバ個別に、メッセージをログするレベルを設定する必要があります。

クラスタ内のワークフローの管理

Identity Manager ユーザアプリケーションワークフロークラスタは、プロセスインスタンスをそれが開始されたエンジンとバインドします。このために、ワークフロープロセスインスタンスが、エンジン ID と関連付けられ、クラスタデータベースで管理されます。ワークフローエンジンが開始されると、そのエンジン ID に割り当てられたプロセスインスタンスが再開されます。こうすることによって、クラスタ内の複数のエンジンが同じプロセスインスタンスを再開することを防止できます。ワークフローエンジンが失敗する

と、そのエンジンで動作していたプロセスはクラスタ内の別のエンジンで自動的に再開されます。

プロセスをクラスタ内の他のエンジンに手動で再割り当てすることもできます。たとえば、管理者はワークフローエンジンが回復したときに失敗したワークフローエンジンにプロセスを再割り当てしたり、クラスタからエンジンが削除されたときに他のエンジンにプロセスを再配布することができます (333 ページのセクション 18.2.7「クラスタ内のワークフロープロセスの管理」を参照)。

ワークフローエンジンの開始時に、そのエンジン ID がクラスタ内のノードで使用されていないかどうかがチェックされます。使用されている場合は、クラスタデータベースをチェックして、エンジンのステータスがシャットダウンまたはタイムアウトになっているかどうかが確認されます。どちらかのステータスに該当する場合は、エンジンが開始されます。ステータスが開始中または実行中の場合、そのワークフローエンジンは警告メッセージをログに記録して、ハートビートタイムアウトが発生するまで待機します。ハートビートタイムアウトが発生した場合、それは同じ ID を持つ他のワークフローエンジンが正しくシャットダウンされなかったことを表します。そのため、安全にエンジンを開始できます。ハートビートタイマーが更新された場合、それは同じ ID を持つ他のワークフローエンジンがクラスタ内で動作中であることを表します。そのため、エンジンを開始することはできません。ハートビートタイムアウト (ワークフローエンジンがタイムアウトになったと判断するまでのハートビート間の最大経過時間) は、ユーザアプリケーションの *［ハートビートの間隔 :］* と *［ハートビートファクタ］* で設定できます (199 ページのセクション 8.4.2「ワークフロークラスタの環境設定」を参照)。

2.8 テキストのローカライズ

Identity Manager には、ユーザアプリケーションをローカライズするためのさまざまなツールが用意されています。この節では、ユーザアプリケーションでローカライズを行うための参照情報を説明します。

表 2-5 *ローカライズのトピック*

ローカライズのトピック	場所
ユーザアプリケーションの優先ロケールの設定	『*Identity Manager ユーザアプリケーション : ユーザガイド* (http://www.novell.com/documentation/idm35/index.html)』の優先言語の選択に関する項目を参照してください。
電子メールテンプレート	詳細については、346 ページのセクション 18.4.4「ローカライズされた電子メールテンプレートの追加」を参照してください。
本人確認の質問	『*Novell Identity Manager 管理ガイド*』の本人確認の質問に関する項目を参照してください。
パスワード同期ステータスアプリケーション名	詳細については、134 ページの ï 5-14§「パスワード同期ステータスアプリケーション設定」を参照してください。
コンテナページの名前	147 ページのセクション 6.2.1「コンテナページの作成」の *［ページ名］* を参照してください。
共有ページの名前	詳細については、155 ページのセクション 6.3.1「共有ページの作成」を参照してください。

ローカライズのトピック	場所
ポータル初期設定	詳細については、183 ページのセクション 7.3.5「ポートレット登録の初期設定の変更」を参照してください。
iManager で作成されたプロビジョニング要求定義	詳細については、304 ページのセクション 17.3.2「プロビジョニング要求の作成または編集」を参照してください。
プロビジョニングチーム定義	353 ページのセクション 19.2.2「プロビジョニングチームの作成または編集」
Designer での、ディレクトリ抽象化層オブジェクトおよびプロビジョニング要求定義の表示ラベルのローカライズに関する一般情報	『*Identity Manager 3.5.1 ユーザアプリケーション: 設計ガイド*』の表示ラベルのローカライズに関する項目を参照してください。
エンティティ表示ラベル	『*Identity Manager 3.5.1 ユーザアプリケーション: 設計ガイド*』のエンティティの追加に関する項目を参照してください。
グローバルリストのラベルの表示	『*Identity Manager 3.5.1 ユーザアプリケーション: 設計ガイド*』のリストの作業に関する項目を参照してください。
リレーションシッププロパティの表示ラベル	『*Identity Manager 3.5.1 ユーザアプリケーション: 設計ガイド*』のリレーションシッププロパティに関する項目を参照してください。
デジタル署名宣言文字列	『*Identity Manager 3.5.1 ユーザアプリケーション: 設計ガイド*』の署名宣言の作成に関する項目を参照してください。
ワークフローアクティビティ表示名	『*Identity Manager 3.5.1 ユーザアプリケーション: 設計ガイド*』のワークフローアクティビティ参照に関する項目を参照してください。

ログのセットアップ 3

この節では、次の内容について説明します。

3.1 イベントログについて

Identity Manager ユーザアプリケーションは、Apache Software Foundation より配布されるオープンソースログパッケージである log4j を使用してログを行います。詳細は、「Logging Services (http://logging.apache.org/log4j)」を参照してください。デフォルトでは、イベントメッセージは、システムコンソールおよびアプリケーションサーバのログファイルに、「情報」以上のログレベルで記録されます。Novell® Audit にログするようユーザアプリケーションを設定することもできます。イベントは、アクティブ化されたすべてのロガー (ログの記録先) にログされます。

重要 : Novell Audit にログを記録している場合は、Novell Audit ドキュメント (http://www.novell.com/documentation/novellaudit20/index.html) を参照してください。

警告 : デジタル署名したドキュメントを保持するには、Novell Audit(または Sentinel) を使用する必要があります。デジタル署名ドキュメントはワークフローデータと一緒にユーザアプリケーションデータベースには保管されません。ログデータベースに保管されます。これらのドキュメントを保管するには、ログを有効にする必要があります。

---

log4j 環境設定は、次の場所にあります。

- jboss-log4j.xml:JBoss アプリケーションサーバのインストールディレクトリにあります。
- log4j.xml:JBoss 以外のアプリケーションサーバの、ユーザアプリケーション WAR 内にあります。

3.1.1 ログレベル設定について

コンソールのログでは、同期書き込みが行われます。このため、ログの書き込み作業によってプロセッサ使用率の問題や同時並行のインピーダンスの問題が起こる可能性があります。JBoss サーバの優先値デフォルト設定を ERROR にすることができます。そうするには、<*installdir*> /jboss/server/IDM/conf/jboss-log4j.xml の設定を編集します。次のような root ノードを見つけます。

```
<root>
   <priority value="INFO"/>
   <appender-ref ref="CONSOLE"/>
   <appender-ref ref="FILE"/>
</root>
```

優先値を次のように変更します。

```
<root>
      <priority value="ERROR"/>
      <appender-ref ref="CONSOLE"/>
      <appender-ref ref="FILE"/>
</root>
```

root に値を割り当てると、レベルが明示的に割り当てられていないアペンダはすべて root のレベルを継承します。

3.1.2 ユーザアプリケーションのログレベル設定の変更

ユーザアプリケーションでは、各ロガー個別にログレベル設定を変更することができます。

1. ユーザアプリケーションにユーザアプリケーション管理者としてログインします。
2. ［*Administration*］タブを選択します。
3. ［*ログ*］リンクを選択します。
4. 任意のロガーの［*ログレベル*］を変更します。
5. 変更内容を保存するには、［*ログ変更を保持する*］を選択します。
6. ［*送信*］をクリックします。

ユーザアプリケーションのログ設定が、idmuserapp_logging.xml ファイルに保存されます。JBoss では、パスは <installdir>/jboss/server/IDM/conf/idmuserapp_logging.xml になります。

3.2 Novell Audit または Sentinel サーバへのログの記録

Novell Audit または Sentinel サーバにログを記録する

1. Identity Manager アプリケーションスキー \'83\'7d をログアプリケーションとして Novell Audit サーバに追加します。

   71 ページのセクション 3.2.1「ログアプリケーションとしての Identity Manager アプリケーションスキーマの Novell Audit サーバへの追加」
2. アプリケーションサーバで Novell Audit のプラットフォームエージェントを設定します。

   Novell Audit または Sentinel にイベントをレポートするクライアントはすべて、プラットフォームエージェントが必要です。プラットフォームエージェントは、logevent 環境設定ファイルで設定できます。このファイルには、プラットフォームエージェントが Novell Audit サーバと通信するために必要な構成情報が含まれています。このファイルはアプリケーションサーバで次のデフォルトの場所にあります。

   - Linux:/etc/logevent.conf
   - Windows:/ <*WindowsDir*> /logevent.cfg(通常は、c:\windows)

   次の 4 つのプロパティを指定します。

   Loghost: Novell Audit または Sentinel サーバの、IP アドレスまたは DNS 名です。例 :

LogHost=xxx.xxx.xxx.xxx

LogJavaClassPath: lcache jar ファイル NauditPA.jar の場所です。例 :
LogJavaClassPath=/opt/novell/idm/NAuditPA.jar

LogCacheDir: lcache がキャッシュファイルを保管する場所を指定します。例 :
LogCacheDir=/opt/novell/idm/naudit/cache

LogCachePort: lcache が接続を待機するポートを指定します。デフォルトは 288 ですが、Linux サーバの場合は 1000 より大きいポートを設定してください。例 :
LogCachePort=1233

BigData クライアントが許可する最大バイト数を指定します。これより大きいログデータは切り詰められます。デフォルトは 3072 バイトです。ただし、一般的な半ページに約 15 個のフィールドがあるフォームを処理するために、最低でも 8192 バイトに変更してください。
LogMaxBigData=8192

重要 : データがとても大きい場合は、この値をさらに増やしてください。デジタル署名を含むイベントを記録する場合、記録するデータを十分に処理できる値を LogMaxBigData に指定する必要があります。

ユーザの環境に応じて他の設定を指定します。

注 : 環境設定を変更した場合は、必ずプラットフォームエージェントを再起動してください。

---

logevent 環境設定ファイル構造の詳細については、『*Novell Audit Administration Guide*』のログシステムに関する章の「Configuring the Platform Agent (http://www.novell.com/documentation/novellaudit20/index.html)」を参照してください。

3 Novell Audit のログを有効にする詳細については、72 ページのセクション 3.2.2 「Audit のログを有効にする」を参照してください。

3.2.1 ログアプリケーションとしての Identity Manager アプリケーションスキーマの Novell Audit サーバへの追加

ログアプリケーションとして Identity Manager ユーザアプリケーションを使用するよう Novell Audit を設定する

1 次のファイルを検索します。
dirxml.lsc

このファイルは、Identity Manager ユーザアプリケーションインストールディレクトリにあります (例 :/opt/novell/idm)。

2 Web ブラウザを使って、Novell Audit プラグインがインストールされた iManager にアクセスし、管理者としてログインします。

3 *［Roles and Tasks (役割とタスク)］> ［Auditing and Logging (監査とログ)］* の順にクリックし、*［Logging Server Options (ログサーバオプション)］* を選択します。

4 ツリー内の［Logging Services container (ログサービスコンテナ)］を参照し、適切な［Audit Secure Logging Server (監査セキュアログサーバ)］を選択します。*［OK］をクリックします。*

5 *［Log Applications (ログアプリケーション)］*タブを表示し、適切なコンテナ名を選択してから *［New Log Application (新規ログアプリケーション)］*リンクをクリックします。

6 ［New Log Application (新規ログアプリケーション)］ダイアログボックスが表示されたら、次のように指定します。

設定項目	操作手順
Log Application Name	ユーザの環境で適切な名前を入力します。
Import LSC File	*［参照］*ボタンを使って、dirxml.lsc ファイルを選択します。

*［OK］*をクリックします。追加されたアプリケーションの名前が *［Log Applications (ログアプリケーション)］*タブに表示されます。

7 ［*OK］をクリックして Novell Audit サーバの設定を完了します。*

8 ログアプリケーションのステータスがオンになっていることを確認してください。(オンの場合、ステータスの下の円が緑色になっています。赤の場合は、クリックしてオンにしてください)。

9 Novell Audit サーバを再起動して、新しいログアプリケーション設定をアクティブ化します。

3.2.2 Audit のログを有効にする

Identity Manager ユーザアプリケーションで Novell Audit のログを有効にする :

1 ユーザアプリケーションにユーザアプリケーション管理者としてログインします。

2 ［*Administration］タブを選択します。*

3 *［ログ］*リンクを選択します。

4 *［ログメッセージを NovellAudit にも送信する］*チェックボックス (ページの下部) を選択します。

5 後でアプリケーションサーバが再起動されてもこの変更を保持するために、*［ログ変更を保持する］*が選択されていることを確認します。

6 ［*送信*］をクリックします。

3.2.3 ログに記録されるイベント

Identity Manager ユーザアプリケーションは、ワークフロー、検索、詳細、およびパスワードの要求から自動的にイベントのセットをログに記録します。デフォルトでは、Identity Manager ユーザアプリケーションはアクティブなログチャンネルすべてに次のイベントを自動的にログに記録します。

表 3-1 *ログに記録されるイベント*

イベント ID	Process	イベント	重大度
31400	詳細ポートレット	Delete_Entity	Info
31401		Update_Entity	Info

イベント ID	Process	イベント	重大度
31410	パスワード変更ポートレット	Change_Password_Failure	エラー
31411		Change_Password_Success	Info
31420	パスワードを忘れた場合のポートレット	Forgot_Password_Change_Failure	エラー
31421		Forgot_Password_Change_Success	Info
31430	検索ポートレット	Search_Request	Info
31431		Search_Saved	Info
31440	作成ポートレット	Create_Entity	Info
31470	デジタル署名	Digital_Signature_Verification_Request	Info
31471		Digital_Signature_Verification_Failure	エラー
31472		Digital_Signature_Verification_Success	Info
31520	ワークフロー	Workflow_Error	エラー
31521		Workflow_Started	Info
31522		Workflow_Forwarded	Info
31523		Workflow_Reassigned	Info
31524		Workflow_Approved	Info
31525		Workflow_Refused	Info
31526		Workflow_Ended	Info
31527		Workflow_Claimed	Info
31528		Workflow_Unclaimed	Info
31529		Workflow_Denied	Info
31534		Workflow_Escalated	Info
31535		Workflow_Reminder_Sent	Info
31536		Digital_Signature	Info
31537		Workflow_ResetPriority	Info
3152A		Workflow_Completed	Info
3152B		Workflow_Timedout	Info
3152C		User_Message	Info
31533		Workflow_Retracted	Info

イベント ID	Process	イベント	重大度
3152D	プロビジョニング	Provision_Error	エラー
3152E		Provision_Submitted	Info
3152F		Provision_Success	Info
31530		Provision_Failure	エラー
31531		Provision_Granted	Info
31532		Provision_Revoked	Info
31450	セキュリティコンテキスト	Create_Proxy_Definition_Success	Info
31451		Create_Proxy_Definition_Failure	エラー
31452		Update_Proxy_Definition_Success	Info
31453		Update_Proxy_Definition_Failure	エラー
31454		Delete_Proxy_Definition_Success	Info
31455		Delete_Proxy_Definition_Failure	エラー
31456		Create_Delegatee_Definition_Success	Info
31457		Create_Delegatee_Definition_Failure	エラー
31458		Update_Delegatee_Definition_Success	Info
31459		Update_Delegatee_Definition_Failure	エラー
3145A		Delete_Delegatee_Definition_Success	Info
3145B		Delete_Delegatee_Definition_Failure	エラー
3145C		Create_Availability_Success	Info
3145D		Create_Availability_Failure	エラー
3145E		Delete_Availability_Success	Info
3145F		Delete_Availability_Failure	エラー

3.2.4 ログレポート

Novell Audit のデータベースチャネルにイベントのログを記録する場合、そのデータに関するレポートを生成することができます。Novell Audit のデータベースにログされるデータに対し、次のような方法でレポートを生成できます。

- Novell Audit のレポートアプリケーションを使用して、独自のレポートを実行する。または、75 ページの 「事前定義されたログレポート」で説明されている事前定義レポートを実行する。
- iManager の *［Auditing and Logging (監査とログ)］> ［Queries (クエリ)］* を使用して、ログデータに対するクエリを記述する。
- ログデータに対する SQL クエリを独自に記述する。
- Sentinel で Identity Manager を生成する (77 ページの 「Sentinel のレポート」を参照)。

デフォルトの Novell Audit のテーブルは NAUDITLOG という名前が付けられます。

事前定義されたログレポート

次の事前定義されたログレポートは Crystal Reports(.rpt) 形式で作成され、Novell Audit データベースに記録されたデータをフィルタリングできます。

レポート名	説明
Administrative Action(管理アクションレポート)	Identity Manager のユーザアプリケーションポータルで開始された管理アクションがすべて表示されます。このレポートには、アクションを開始した管理者が含まれます。 iManager または Designer for Identity Manager を使用して実行された管理上の変更は除外されます。
Historical Approval Flow(認証フロー履歴レポート)	指定した期間内での認証フローのアクティビティが \'95\'5c 示されます。
Resource Provisioning(リソースプロビジョニングレポート)	すべてのプロビジョニングアクティビティがリ \'83\'5cース別に \'95\'5c 示されます。
User Audit Trail(特定ユーザの監査記録レポート)	あるユーザに関係するアクティビティがすべて表示されます。アクティビティには、プロビジョニングとセルフサービスの両方のアクティビティが含まれます。
Specific User Provisioning (特定のユーザのプロビジョニング)	特定ユーザのプロビジョニングアクティビティがすべて \'95\'5c 示されます。
User Provisioning (ユーザのプロビジョニング)	すべてのプロビジョニングアクティビティがユーザ別に \'95\'5c 示されます。

User Audit Trail(特定ユーザの監査記録レポート) の例を次に示します。

図 3-1 サンプルの監査記録レポート

Novell® Audit Report for Identity Manager

特定ユーザの監査記録

レポート期間： - 10/13/2005 8:51:32AM

ユーザ ID： ablake

レポートの最終変更日: 10/13/2005
レポートの生成日: 10/13/2005
合計ページ： 8

承認フロー

ワークフローイベント: fecedbe80a3d4abd83c9476a1b576ea2

日付/時間	アクション	イニシエータID
9/12/2005 3:20:42PM	ワークフローが開始されました	cn=ablake,ou=users,ou=idmsample-Jeff,o=novell
9/12/2005 3:20:43PM	ワークフローが転送されました	ワークフロー管理者
9/12/2005 3:25:43PM	ワークフローが再割り当てされました	未クレーム
9/12/2005 3:30:44PM	ワークフローが転送されました	ワークフロー管理者
9/12/2005 3:30:44PM	ワークフローが終了しました	ワークフロー管理者
9/12/2005 3:30:44PM	ワークフローが却下されました	システム

ワークフローイベント: fc6d74b1268243b3beac52261439dea0

日付/時間	アクション	イニシエータID
9/28/2005 1:12:19PM	ワークフローが開始されました	cn=ablake,ou=users,ou=idmsample-Jeff,o=novell
9/28/2005 1:12:22PM	ワークフローが転送されました	ワークフロー管理者
9/28/2005 2:12:23PM	ワークフローが転送されました	ワークフロー管理者
9/28/2005 2:12:23PM	ワークフローが転送されました	ワークフロー管理者
9/28/2005 2:12:23PM	ワークフローが転送されました	ワークフロー管理者
9/28/2005 2:12:23PM	ワークフローが転送されました	ワークフロー管理者
9/28/2005 2:12:23PM	ワークフローが承認されました	システム
9/28/2005 2:12:23PM	ワークフローが承認されました	システム
9/28/2005 2:12:23PM	ワークフローが完了しました	ワークフロー管理者
9/28/2005 2:12:27PM	ワークフローが転送されました	ワークフロー管理者
9/28/2005 2:12:27PM	ワークフローが終了しました	ワークフロー管理者
9/28/2005 2:12:27PM	プロビジョンが送信されました	ワークフロー管理者
9/28/2005 2:12:27PM	プロビジョンが許可されました	ワークフロー管理者

ワークフローイベント: efaa8304e07641edb9e375a1a36e396

日付/時間	アクション	イニシエータID
10/12/2005 11:58:13AM	ワークフローが開始されました	cn=ablake,ou=users,ou=idmsample-qatest,o=novell
10/12/2005 11:58:13AM	ワークフローが転送されました	ワークフロー管理者

ワークフローイベント: ea341eb11a824e669e356837745fe264

日付/時間	アクション	イニシエータID
9/27/2005 4:24:44PM	ワークフローが開始されました	cn=mmackenzie,ou=users,ou=idmsample-Jeff,o=novell
9/27/2005 4:24:44PM	ワークフローが転送されました	ワークフロー管理者

Page 1 of 8

Specific User Audit Trait

レポートファイルは、次の場所にあります。

プラットフォーム	ディレクトリ
Windows	/nt/dirxml/reports

これらのレポートをテンプレートとして使用して、Crystal Reports Designer でカスタムレポートを作成できます。また、Novell Audit 付属の Windows プログラムである Audit Report(lreport.exe) を使用してレポートを実行することもできます。事前定義されたレポー

トは、Novell Audit のデフォルトログデータベース naudit とデータベーステーブル nauditlog に対してデータの問い合わせを行います。ご使用の Novell Audit ログデータベースの名前が異なる場合は、Crystal Reports Designer の［*Set Datasource Location*］メニュー項目を使用して、データベース名 naudit をご使用の環境のデータベース名に変えてください。

詳細については、Novell Audit の \'83\'7d ニュアル (http://www.novell.com/documentation/novellaudit20) にあるレポートでの作業のセクションを参照してください。

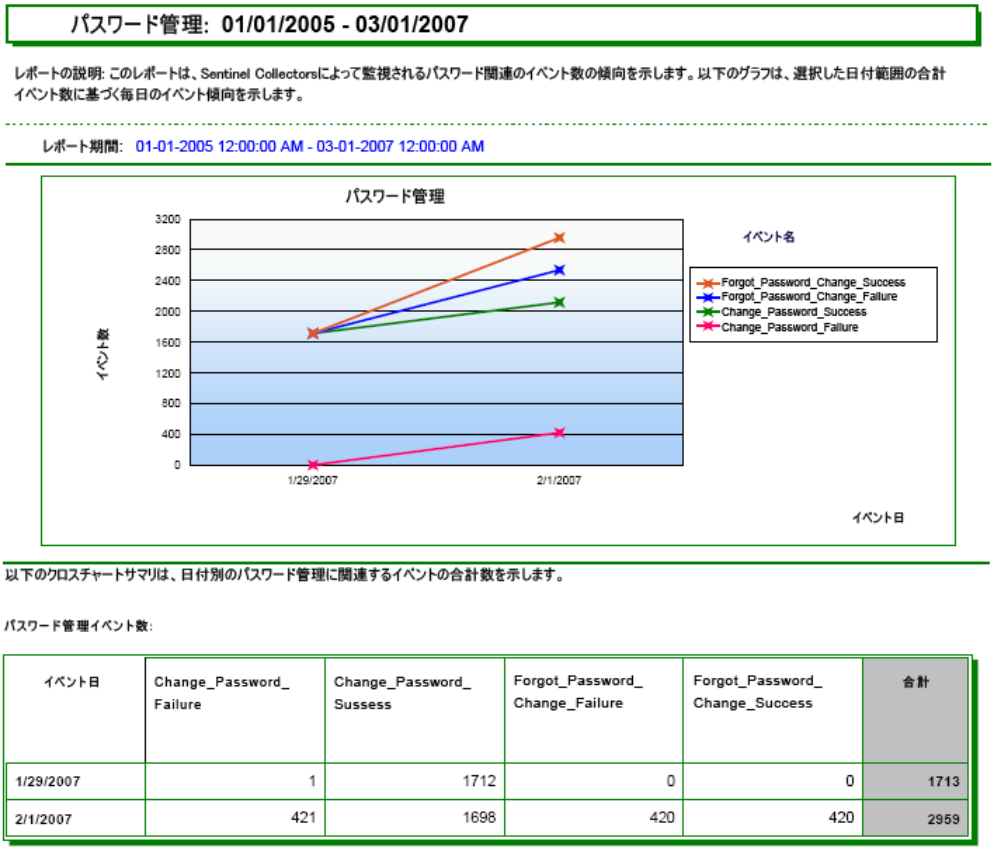
Sentinel のレポート

イベントを Sentinel に送信するようにプラットフォームエージェントを設定している場合、Sentinel で Identity Manager イベントに関する次のレポートを生成できます。

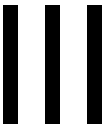
- IDM_Administrative_Action_Report.rpt
- IDM_Historical_Approval_Flow_Report.rpt
- IDM_Password-Management.rpt
- IDM_Provisioning_Report_by_Top_10_DHNs.rpt
- IDM_Provisioning_Report_by_Top_10_DIPs.rpt
- IDM_Resource_Provisioning_Report.rpt
- IDM_Specific_User_Audit_Trail_Report.rpt
- IDM_Specific_User_Provisioning_Report.rpt
- IDM_Sync-vs-Reset.rpt
- IDM_User_Provisioning_Report.rpt
- IDM_Workflow_Stats_by_Top_10_DHNs.rpt
- IDM_Workflow_Stats_by_Top_10_DIPs.rpt

Sentinel レポートの詳細は、『*Sentinel User's Guide*(Sentinel ユーザガイド)』を参照してください。パスワード管理に関する Sentinel レポートの例を次に示します。

図 3-2 *サンプルの Sentinel レポート*



ユーザアプリケーションの管理



これらの節では、ユーザインタフェースの*［管理］*タブを使用して、Identity Manager ユーザアプリケーションを設定、管理する方法について説明します。

［Administration］タブの使用

4

この節では Identity Manager ユーザインタフェースの *［管理］* タブを使用する方法を解説します。*［管理］* タブを使用して Identity Manager ユーザアプリケーションを設定および管理する方法について説明していきます。主なトピックは次のとおりです。

4.1 ［管理］タブについて

Identity Manager のユーザインタフェースは主にエンドユーザがアクセスし、タブやページを使用して識別情報のセルフサービスやワークフローベースのプロビジョニング (Provisioning Module for Identity Manager を使用) を行います。ただし、このブラウザベースのユーザインタフェースには、*［管理］* タブ／ページも用意されています。このタブで、管理者は Identity Manager ユーザアプリケーションのさまざまな項目を設定することができます。

たとえば、次の作業を行う場合に *［管理］* タブを選択します。

- ユーザインタフェースの外観や操作に使用するテーマの変更
- エンドユーザが使用する Identity セルフサービス機能のカスタマイズ
- 管理者アクションを実行できるユーザの指定
- ユーザアプリケーションおよびその実行方法に関する他の詳細の管理

4.2 ［管理］タブを使用できるユーザ

［管理］ タブは、Identity Manager ユーザインタフェースの一般エンドユーザには表示されません。このタブを表示、アクセスできるユーザは次の 3 つのタイプに限定されます。

ユーザアプリケーション管理者 : ユーザアプリケーション管理者は、Identity Manager ユーザに関連するすべての管理機能を実行できます。この中には、Identity Manager ユーザインタフェースの *［管理］* タブにアクセスし、そこでサポートされているすべての管理アクションを実行する操作も含まれます。インストール時に、1 人のユーザがユーザアプリケーション管理者として指定されます。インストール後、そのユーザは *［管理］* タブにある［セキュリティ］ページを使用して、必要に応じてその他のユーザアプリケーション管理者を指定できます。詳細については、201 ページの第 9 章「セキュリティ設定」を参照してください。

プロビジョニングアプリケーション管理者 : プロビジョニングアプリケーション管理者は、Identity Manager ユーザアプリケーションのプロビジョニングに関連する作業を行います。インストール時に、1 人のユーザがプロビジョニングアプリケーション管理者として指定されます。詳細については、189 ページの第 8 章「プロビジョニング環境設定」を参照してください。ユーザアプリケーション管理者は、*［管理］* タブの［セキュリティ］ページから、プロビジョニングアプリケーション管理者を指定できます。これには、*［管*

理] タブの［プロビジョニング］ページにおける作業の実施も含まれます。詳細については、201 ページの第 9 章「セキュリティ設定」を参照してください。

ユーザアプリケーション管理者に許可されたユーザ : 必要に応じて、ユーザアプリケーション管理者は、1 人または複数のエンドユーザに対し、*［管理］* タブの特定のページへのアクセス許可を割り当てることができます。これらの許可の割り当てには、*［管理］* タブの［ページ管理］ページを使用します。詳細については、139 ページの第 6 章「ページの管理」を参照してください。

4.3 ［管理］タブへのアクセス

ユーザアプリケーション管理者 (または許可された他のユーザ) になると、Identity Manager ユーザアプリケーションを管理するために、Identity Manager ユーザインタフェースの *［管理］* タブにアクセスできます。サポートされている Web ブラウザさえあればアクセスできます。

サポートされている Web ブラウザの詳細については、『*Novell Identity Manager: インストールガイド*』を参照してください。.

注 : Identity Manager ユーザインタフェースを使用する場合、Web ブラウザで JavaScript* と cookies が有効になっていることを確認してください。

［管理］ タブにアクセスする

1 Web ブラウザで、Identity Manager ユーザインタフェースの URL へ移動します (サイトの設定により異なります)。例 :

```
http://myappserver:8080/IDM
```

［ようこそゲストさん］ページが表示されます。

2 ページヘッダにある［*Login*］のリンクをクリックします。

ユーザ名とパスワードを入力するよう要求されます。

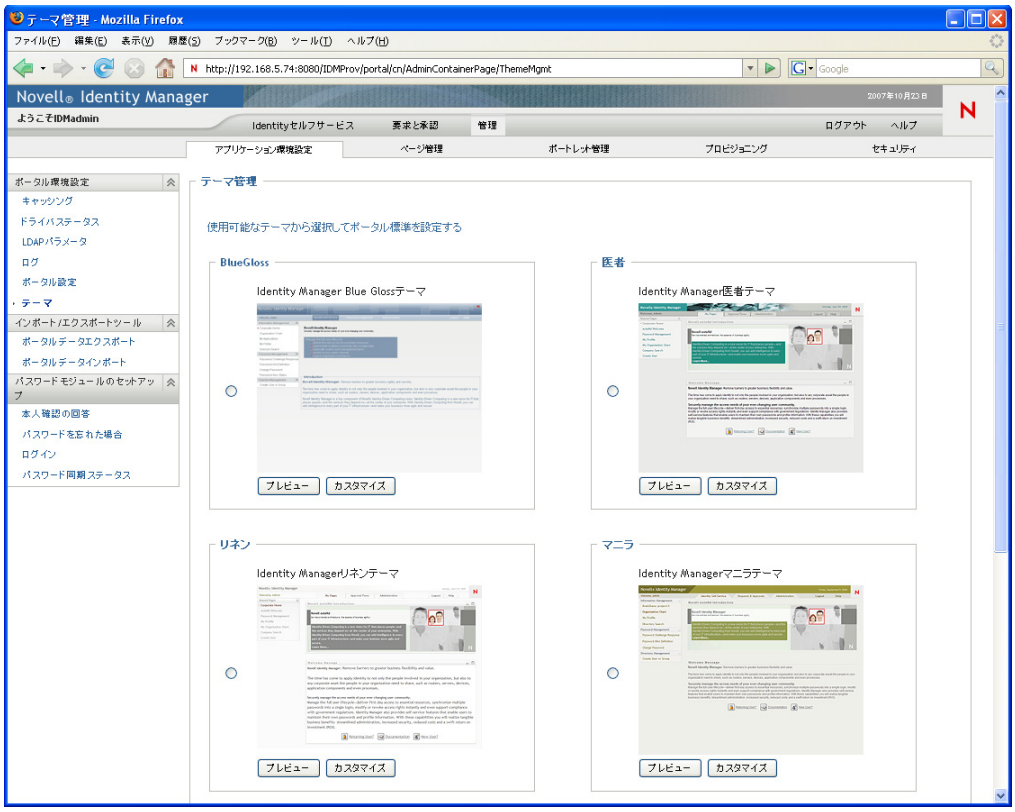
3 ユーザアプリケーション管理者 (または［*管理*］タブにアクセスできるユーザ) のユーザ名とパスワードを指定して、［*ログイン*］をクリックします。

ログインすると、そのユーザ用のコンテンツが表示されます。

デフォルトでは、［*Identity Self-Service*］タブが *\'95\'5c* 示されます。

4 ［*Administration*］タブをクリックします。

［*管理*］タブには、実行できる管理者アクションのメニューが表示されます。メニューの各項目から、対応する設定や制御のページが表示されます。デフォルトでは、［アプリケーション環境設定］ページが表示されます。



Identity Manager ユーザインタフェースのアクセスや作業に関する一般的な情報については、『*Identity Manager ユーザアプリケーション : ユーザガイド*』を参照してください。

4.4 実行できる管理アクション

[管理] ページが表示されたら、使用可能なアクションを使用して Identity Manager ユーザアプリケーションを設定および管理できます。表 4-1 に概要を示します。

表 4-1 *管理アクションの概要*

アクション	説明
アプリケーション環境設定	キャッシュ、ログ、パスワード管理、および LDAP 接続パラメータなどの、ユーザアプリケーション環境設定を制御します。ドライバステータスとポータルに関する読み込み専用情報が表示されます。ポータル (Identity Manager ユーザアプリケーションで使用されるページやポートレット) のコンテンツをエクスポート／インポートできるツールを利用できます。 詳細については、87 ページの第 5 章「アプリケーション環境設定」を参照してください。
PageAdmin	Identity Manager のユーザインタフェースに \'95\'5c 示されるページの制御、およびそれに対する許可を持つユーザの制御。 詳細については、139 ページの第 6 章「ページの管理」を参照してください。

アクション	説明
Portlet Admin	Identity Manager のユーザインタフェースで使用できるポートレットの制御、およびそれに対する許可を持つユーザの制御。 詳細については、173 ページの第 7 章「ポートレットの管理」を参照してください。
プロビジョニング	委任／代理人タスク、デジタル署名サービス、エンジン、クラスタなどの環境設定を制御します。 詳細については、189 ページの第 8 章「プロビジョニング環境設定」を参照してください。
セキュリティ	Identity Manager ユーザアプリケーションのユーザアプリケーション管理者やプロビジョニング管理者を指定します。 詳細については、201 ページの第 9 章「セキュリティ設定」を参照してください。

アプリケーション環境設定

5

このセクションでは、［アプリケーション環境設定］ページで行える作業について説明していきます。この章は次の節から構成されています。

5.1 ポータル環境設定作業

この節には、次の情報を記載しています。

5.1.1 キャッシュ管理

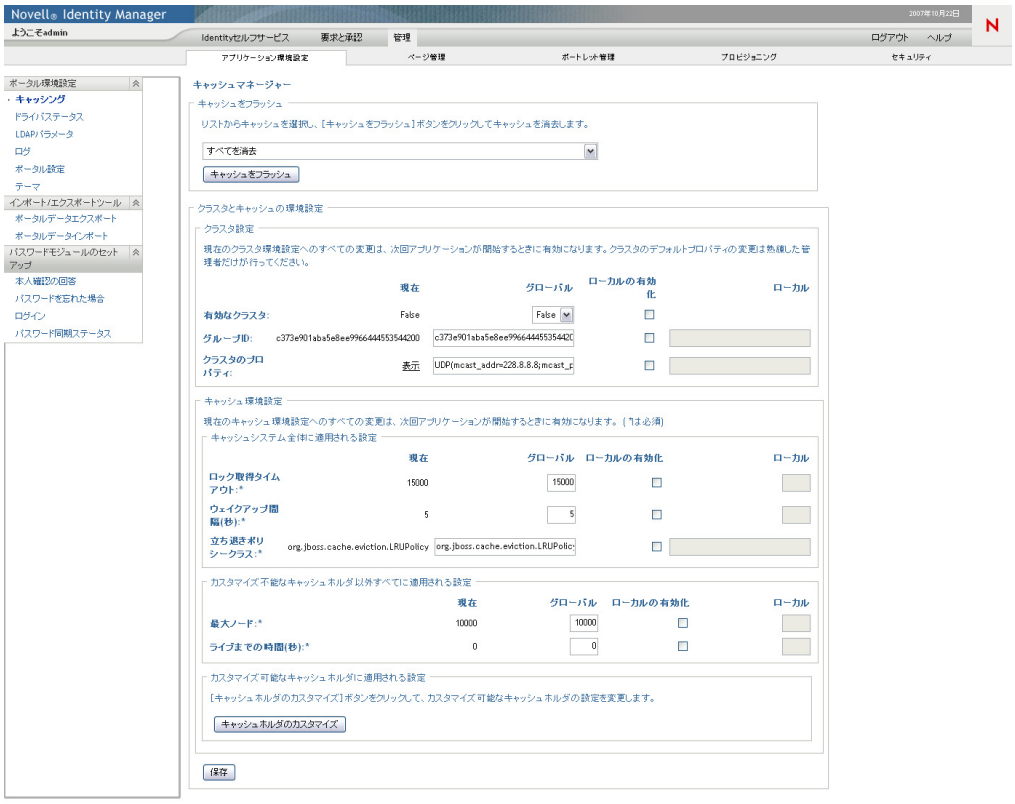
［キャッシング］ページを使用して、Identity Manager ユーザアプリケーションが使用するさまざまなキャッシュを管理できます。再利用可能な一時データをアプリケーションサーバに格納してパフォーマンスを最適化するために、ユーザアプリケーションではキャッシュが使用されます。

コンテンツの消去およびその設定の変更を行うと、必要に応じてこれらのキャッシュを制御できます。

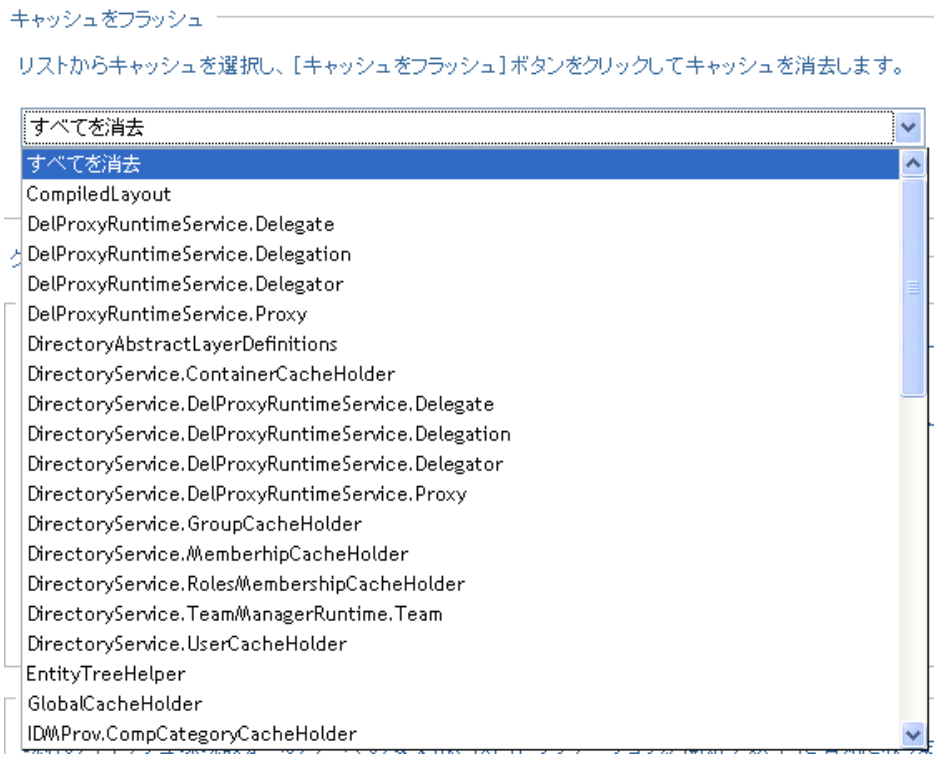
キャッシュの消去

キャッシュは、Identity Manager ユーザアプリケーションでそのキャッシュを使用するサブシステムに基づいて名前が付けられます。通常は、データの使用頻度またはソースデータの変更時期に基づいてユーザアプリケーションが自動的にキャッシュをフラッシュするため、ユーザが自らキャッシュを消去する必要はありません。ただし、必要に応じて、選択したキャッシュまたはすべてのキャッシュを手動でフラッシュできます。

1 ［Caching］ページに移動します。



2 ページの *[キャッシュをフラッシュ]* セクションで、ドロップダウンからフラッシュする対象のキャッシュを選択します (または *[すべてを消去]* を選択します)。



使用可能なキャッシュのリストは、動的なリストです。その時点でキャッシングされているデータに従って変わります。

3 *[キャッシュをフラッシュ]* ボタンをクリックします。

ディレクトリ抽象化層キャッシュのフラッシュ

ユーザアプリケーションのディレクトリ抽象化層にもキャッシュが存在します。DirectoryAbstractLayerDefinitions キャッシュは、すべてのデータモデル操作についてのパフォー \'83\'7d ンスを最適化するために、アプリケーションサーバに抽象化層定義を格納します。

通常、ユーザアプリケーションは、DirectoryAbstractLayerDefinitions キャッシュと、アイデンティティボールトに格納されている抽象化層定義との同期を自動的に行います。ただし、必要に応じて、最新定義を強制的にアイデンティティボールトからロードさせるために、87 ページの 「キャッシュの消去」 で説明している方法で DirectoryAbstractLayerDefinitions キャッシュを手動でフラッシュすることもできます。

ユーザアプリケーションのディレクトリ抽象化層の詳細は、「*Identity Manager ユーザアプリケーション: 設計ガイド*」を参照してください。

クラスタ内のキャッシュのフラッシュ

キャッシュのフラッシュは、クラスタアプリケーションサーバ環境および非クラスタアプリケーションサーバ環境の両方について、サポートされています。アプリケーションサーバがクラスタの一部である場合に手動でキャッシュをフラッシュすると、クラスタ内にあるすべてのサーバのキャッシュも自動的にフラッシュされます。

キャッシュの設定

［キャッシング］ページを使用して、クラスタアプリケーションサーバ環境または非クラスタアプリケーションサーバ環境のキャッシュ環境設定を表示または変更できます。変更はただちに保存されますが、次回ユーザアプリケーションが再起動されるまで有効になりません。

ヒント : ユーザアプリケーションを再起動するには、アプリケーションサーバの再起動、アプリケーションの再展開 (WAR が変更されている場合)、アプリケーションの強制的な再起動 (アプリケーションサーバのマニュアルに記載されている方法による) のいずれかを行います。

キャッシングの実装方法

Identity Manager ユーザアプリケーションでは、キャッシングは JBoss Cache により実装されています。JBoss Cache は、JBoss Application Server に含まれているオープンソースのキャッシングアーキテクチャであり、他のアプリケーションサーバでも実行できます。

JBoss Cache の詳細については、www.jboss.org/products/jbosscache (http://www.jboss.org/products/jbosscache) を参照してください。

キャッシュ設定の格納方法

キャッシュの環境設定は、グローバルレベルとローカルレベルで制御できます。これらの設定を使って、Identity Manager ユーザアプリケーションのキャッシュの動作をカスタマイズできます。キャッシュの環境設定については、90 ページの 表 5-1 を参照してください。

表 5-1 *キャッシュの環境設定*

レベル	説明

［Global］設定	グローバル設定は、複数のアプリケーションサーバが同じ設定値を使用できるように、まとめてアイデンティティボールトに格納されます。たとえば、アプリケーションサーバのあるクラスタのユーザは通常、グローバル設定のクラスタ設定値のを使用します。
	
	アイデンティティボールトからグローバル設定を見つけるには、Identity Manager ユーザアプリケーションドライバの下にある次のオブジェクトを探します。
	`configuration.AppDefs.AppConfig`
	
	例 :
	`configuration.AppDefs.AppConfig.MyUserApplicationDriver.MyDriverSet.MyOrg`
	
	設定オブジェクトの XmlData 属性には、グローバル設定データが含まれています。

レベル	説明
ローカル設定	ローカル設定は、各サーバが 1 つまたは複数のグローバル設定の値を上書きできるように、各アプリケーションサーバに個別に保存されます。たとえば、アプリケーションサーバをグローバル設定で指定したクラスタから削除したり、サーバを別のクラスタに再割り当てしたりする場合に、ローカル設定を指定できます。 JBoss アプリケーションサーバのローカル設定は、JBoss サーバ環境設定の conf ディレクトリ下にある sys-configuration-xmldata.xml ファイルを参照してください (例 : jboss/server/IDM/conf/sys-configuration-xmldata.xml)。 WebSphere アプリケーションサーバのローカル設定は、インストール時に設定した extend.local.config.dir プロパティで指定した場所にある sys-configuration-xmldata.xml ファイルを参照してください。 サーバがローカル設定になっている場合、そのデータはこのファイルに含まれます (ローカル設定が指定されていない場合、ファイルは存在しません)。

グローバル設定は、ユーザアプリケーションドライバの特定のインスタンスを使用する各アプリケーションサーバのデフォルト値と考えます。グローバル設定の変更は、サーバが個別にローカル上書きを指定している場合を除き、次回ユーザアプリケーションの再起動時に、各サーバに反映されます。

キャッシュ設定の表示方法

[キャッシング] ページでは、現在の (最後にユーザアプリケーションを再起動してからの) キャッシュ設定が表示されます。また、これらの設定に対応するグローバル値およびローカル値も表示され、設定を変更することもできます (変更された設定は、次回ユーザアプリケーションの再起動時から有効になります)。

グローバル設定には常に値があります。一方、ローカル設定はオプションです。

基本的なキャッシュ設定

次のキャッシュ設定は、クラスタアプリケーションサーバ環境および非クラスタアプリケーションサーバ環境の両方に適用されます。

基 \'96\'7b 的なキャッシュ設定を行う

1 ［Caching］ページに移動します。

2 *ページの Cache Configuration］セクションで、必要に応じて、次の設定のグローバル値またはローカル値*を指定します。

設定	操作
ロック取得タイムアウト	オブジェクトでロックが取得されるまでキャッシュが待機する間隔 (ミリ秒) を指定します。ユーザアプリケーションのアプリケーションログに大量のロックタイムアウト例外が書き込まれる場合に、この設定値を増やすことができます。デフォルトは 15000 ミリ秒です。
Wake Up Interval Seconds	次を行うまでにキャッシュ強制更新ポリシーが待機する間隔 (秒) を指定します。 ◆ 強制更新ノードイベントの処理 ◆ サイズ制限および期限切れノードのクリーンアップ

設定	操作
Eviction Policy Class	使用するキャッシュ立ち退きポリシーのクラス名を指定します。デフォルトは、JBoss Cache が提供する LRU 強制更新ポリシーです。 `org.jboss.cache.eviction.LRUPolicy` この設定は、必要に応じて、JBoss Cache がサポートする別の強制更新ポリシーに変更できます。 サポート対象の立ち退きポリシーについては、www.jboss.org/products/jbosscache (http://www.jboss.org/products/jbosscache) を参照してください。
Max Nodes	キャッシュで許容される最大ノード数を指定します。制限がない場合には、次を指定します。 0 一部のキャッシュホルダに対して、この設定をカスタマイズできます。詳細については、93 ページの 「カスタマイズできるキャッシュホルダ」を参照してください。
Time To Live Seconds	ノードが一掃されるまでのアイドル時間 (秒) を指定します。制限がない場合には、次を指定します。 0 一部のキャッシュホルダに対して、この設定をカスタマイズできます。詳細については、93 ページの 「カスタマイズできるキャッシュホルダ」を参照してください。
MaxAge	エントリが作成されてからキャッシュホルダ内に存在できる秒数を指定します。制限がない場合は、次の値を指定します : 0 この設定は、93 ページの 「カスタマイズできるキャッシュホルダ」でのみ利用できます。

これらの設定は必 \'90\'7b です。各設定にはグローバル値が必要であり、オプションでローカル値も使用します。

設定のグローバル値をローカル値で上書きする場合は、その設定の *[ローカルの有効化]* チェックボックスを選択します。次に、ローカル値を指定してください。(ローカル値がすべて有効であることを確認してください。そうしないと変更を保存できません)。

注 : *[ローカルの有効化]* チェックボックスを選択しない場合、保存時に既存のローカル値が削除されます。

3 [*保存*] をクリックします。

4 保存した設定を反映できる状態になったら、該当アプリケーションサーバ上でユーザアプリケーションを再起動します。

カスタマイズできるキャッシュホルダ

一部のキャッシュホルダに対して、*[最大ノード]*、*[有効期間]*、*[MaxAge]* の設定をカスタマイズできます。これらのキャッシュホルダについては、表 5-2 を参照してください。

表 5-2 カスタマイズできるキャッシュホルダ

キャッシュホルダ名	説明
DirectoryAbstractionLayerDefinitions	すべてのデータモデル操作のパフォーマンスを最適化するために、ディレクトリ抽象化層定義をキャッシュします。詳細については、89 ページの 「ディレクトリ抽象化層キャッシュのフラッシュ」を参照してください。
DirectoryService.ContainerCacheHolder	ディレクトリ層のコンテナをキャッシュします。コンテナは多くのユーザ／グループと共有され、ディレクトリ層から読み込まれる際にはネットワーク通信 (LDAP サーバを使用) とオブジェクト作成を伴います。デフォルトでは、キャッシュのコンテナ数は 50 に制限されています。また、LRU の TTL(有効期間) は、デフォルト値の 10 分になっています。社内のディレクトリトポグラフィに応じて、最大ノード数や TTL を調整してください (コンテナオブジェクトに対するクエリが LDAP サーバに集中し、パフォーマンスが低下しているような場合)。大量の使用可能コンテナがある環境で設定値を大きくしすぎると、メモリが無駄に消費されネットワークパフォーマンスが低下する可能性があります。
DirectoryService.DelProxyRuntimeServiceDelegate	委任割り当てをキャッシュします。
DirectoryService.DelProxyRuntimeService.Delegation	ユーザ可用性設定をキャッシュします。
DirectoryService.DelProxyRuntimeService.Delegator	委任者エンティティをキャッシュします。
DirectoryService.DelProxyRuntimeService.Proxy	代理人割り当てをキャッシュします。
DirectoryService.GroupCacheHolder	ディレクトリ層のグループをキャッシュします。グループはしばしば多くのユーザに共有され、ディレクトリ層から読み込まれる際にはネットワーク通信 (LDAP サーバを使用) とオブジェクト作成を伴います。デフォルトでは、キャッシュのグループ数は 500 に制限されています。また、LRU の TTL(有効期間) は、デフォルト値の 10 分になっています。社内のユーザ／グループトポグラフィに応じて、最大ノード数や TTL を調整してください (グループオブジェクトに対するクエリが LDAP サーバに集中し、パフォーマンスが低下しているような場合)。大量の使用可能グループがある環境で設定値を大きくしすぎると、メモリが無駄に消費されネットワークパフォーマンスが低下する可能性があります。

キャッシュホルダ名	説明
DirectoryService.MemberhipCacheHolder	ユーザとグループのセット間の関係をキャッシュします。ユーザが所属するグループセットをクエリすると、ネットワークと LDAP サーバの CPU に負荷がかかることがあります (特にダイナミックグループを使用している場合)。そのため、グループ内の包含／除外基準 (時間ベースのダイナミックグループなど) の変更が反映されるように、関係が有効期限までキャッシュされます。MaxAge のデフォルトは、5 分です。ただし、きめ細かく時間を制御する必要があるダイナミックグループを使用する場合は、このキャッシュホルダの MaxAge をダイナミックグループの必要に応じて調節した最小時間をわずかに下回るように変更することができます。この値が小さいほど、セッション時にユーザのグループがより多くクエリされます。大きすぎる値を指定すると、ユーザ／グループの関係がユーザのセッションよりも長くメモリに保管され、不要にメモリを消費してしまいます。
DirectoryService.RolesMembershipCacheHolder	役割によるアプリケーション役割メンバーシップリストをキャッシュします。
DirectoryService.TeamManagerRuntime.Team	アプリケーションチームインスタンスとチームプロビジョニング要求をキャッシュします。
DirectoryService.UserCacheHolder	ディレクトリ層のユーザをキャッシュします。ディレクトリ層からのユーザの読み込みには、ネットワーク通信（LDAP サーバを使用）とオブジェクト作成の両方が含まれます。デフォルトでは、キャッシュのユーザ数は 1000 に制限されています。また、LRU の TTL(有効期間) は、デフォルト値の 10 分になっています。社内のユーザトポグラフィに応じて、最大ノード数や TTL を調整してください (ユーザオブジェクトに対するクエリが LDAP サーバに集中し、パフォーマンスが低下しているような場合)。多数のユーザがログインする環境で設定値を大きくしすぎると、メモリが無駄に消費されネットワークパフォーマンスが低下する可能性があります。
GlobalCacheHolder	汎用キャッシュホルダです。この設定は、カスタマイズ不可能すべてのキャッシュホルダ (この表に記載されていないすべてのキャッシュホルダ) に適用されます。

キャッシュホルダ名	説明
JUICE	ユーザインタフェース制御と DN 表示式ルックアップ結果が使用するリソースバンドルをキャッシュします。キャッシュホルダの設定を変更すると、ユーザアプリケーション内で DN 表示式ルックアップが頻繁に使用されるため、パフォーマンスが低下する可能性があります。下限値には 300 秒以上の値を指定してください。上限値は 900 秒を超えても構いません。DN 表示式で使われる属性が頻繁に変更される場合は、下限値を使用します。
RoleManager.RolesCacheHolder	ユーザ役割メンバーシップをユーザ別にキャッシュします。
Workflow.Model.Process	プロビジョニングプロセス XML オブジェクト構造をキャッシュします。
Workflow.Model.Request	プロビジョニング要求 XML オブジェクト構造をキャッシュします。
Workflow.Provisioning	完了していないプロビジョニング要求インスタンスをキャッシュします。LRU キャッシュの最大容量のデフォルトは 500 です。この容量を変更するには、*[管理] ／ [プロビジョニング]* タブをクリックして、[エンジンおよびクラスタの設定] を選択します。このページには、[プロセスキャッシュ最大容量] が表示されます。このキャッシュは、パフォーマンスに影響を与えずにワークフロー処理のメモリ占有量を減らします。

クラスタのキャッシュ設定

この節では、Identity Manager ユーザアプリケーションをアプリケーションサーバのクラスタで実行する場合のキャッシングの設定方法について説明します。

Identity Manager ユーザアプリケーションでは、キャッシングのクラスタサポートは *JGroups* により実装されます。JGroup は、オープン \'83\'5c ースのクラスタリングアーキテクチャであり、JBoss Application Server に含まれていますが他のアプリケーションサーバでも実行できます。

ユーザアプリケーションのクラスタは、JGroups を実行し、共通のグループ ID を使用するネットワーク上のノードから構成されます。デフォルトでは、ユーザアプリケーションのクラスタに用意されているグループ ID は、次のような UUID となります。

```
c373e901aba5e8ee9966444553544200
```

UUID により一意性が保たれるため、ユーザアプリケーションのクラスタのグループ ID が環境内にある他のクラスタのグループ ID と競合することはありません。たとえば、JBoss Application Server では複数の JGroups クラスタが使用され、それぞれ対応するグループ ID である DefaultPartition や Tomcat-Cluster などの関係名は予約されています。

JGroups の詳細については、www.jboss.org/products/jgroups (http://www.jboss.org/products/jgroups) を参照してください。

クラスタでのキャッシングの動作

ユーザアプリケーションを起動すると、アプリケーションの［キャッシング］ページのクラスタ設定により、クラスタに参加してそのクラスタ内の他のノードのキャッシュ変更を無効化するかどうかかが判断されます。クラスタリングが有効になっている場合、ユーザアプリケーションは、変更発生時にキャッシュエントリ無効メッセージを各ノードに送信することによりこれを実行します。

クラスタを使用するための準備作業

クラスタ間でキャッシュを使用する

1. JGroups クラスタを設定します。この作業には、インストールプログラムを使った Identity Manager ユーザアプリケーションのクラスタ内の各アプリケーションサーバへのインストールが含まれます (54 ページのセクション 2.7「クラスタリング」を参照)。
2. ユーザアプリケーションのキャッシュ環境設定におけるクラスタ使用の有効化

   詳細については、97 ページの 「クラスタのキャッシュ設定の実行」を参照してください。

クラスタのキャッシュ設定の実行

クラスタの使用準備ができたら、クラスタのキャッシングサポートについての設定を実行します。

1. ［Caching］ページに移動します。
2. ページの［Cluster Configuration］セクションで、*必要に応じて、次の設定のグローバル値またはローカル値*を指定します。

設定	操作
Cluster Enabled	グループ ID が指定するクラスタ内の他のノードへのキャッシュ変更を無効化するには、*［True］*を選択します。クラスタに参加しない場合は、*［False］*を選択します。
グループ ID	参加対象の JGroups クラスタのグループ ID を指定します。別のクラスタを使用する場合を除き、ユーザアプリケーションのクラスタのために用意されているグループ ID のデフォルトを変更する必要はありません。 グループ ID は一意で、DefaultPartition や Tomcat-Cluster などの既知の JBoss クラスタ名は使用できません。 **ヒント :** グループ ID をログメッセージに表示する場合は、キャッシングログ (com.sssw.fw.cachemgr) のレベルが「情報」以上になっていることを確認します。

設定	操作
Cluster Properties	グループ ID が示すクラスタの JGroups プロトコルスタックを指定します。これは、クラスタのプロパティを調整する必要がある熟練した管理者の方向けの設定です。熟練管理者でない場合は、デフォルトのプロトコルスタックを変更しないでください。 現在のクラスタのプロパティを \'95\'5c 示するには、［*view*］をクリックします。 JGroups プロトコルスタックの詳細については、www.joss.org/wiki/Wiki.jsp?page/JGroups (http://www.jboss.org/wiki/Wiki.jsp?page=JGroups) を参照してください。

設定のグローバル値をローカル値で上書きする場合は、その設定の［*ローカルの有効化*］チェックボックスを選択します。ローカル値を指定します。

［*ローカルの有効化*］チェックボックスを選択しない場合、保存時に既存のローカル値が削除されます。

クラスタ内のすべてのノードの［グループ ID］および［クラスタのプロパティ］が同じ設定になっていることを確認します。特定のノードについてこれらの設定を確認する場合には、そのサーバ上のユーザインタフェースの URL を参照することにより、そのノードで実行している Identity Manager ユーザインタフェースにアクセスし、それから［キャッシング］ページを表示する必要があります。

デフォルトの UDP プロトコルの代わりに TCP プロトコルを使用する場合は、64 ページの 「ユーザアプリケーションクラスタグループのキャッシングの環境設定」を参照してください。

3 ［*保存*］をクリックします。

4 保存した設定を反映できる状態になったら、該当アプリケーションサーバ上でユーザアプリケーションを再起動します。

5.1.2 Driver Status (ドライバのステータス)

［ドライバステータス］ペインでドライバの有効期限ステータスを判断できます。

図 5-1 *トライアルドライバのサンプルのドライバステータス*

有効期限が *［無期限］* の場合、ドライバは開始され完全にライセンスされているか、まだ開始されていません。開始されていない場合は、トライアルドライバである可能性もあります。有効期限が設定されている場合、ドライバはトライアルドライバであり、開始されています。このページは、UNKNOWN の値の意味を説明しています。

5.1.3 LDAP パラメータ

［LDAP パラメータ］ペインでは、次の作業を行えます。

- Identity Managerユーザアプリケーションがアイデンティティボールト(LDAPプロバイダ) に接続するときに使用する資格情報を変更する
- LDAP の匿名アカウントではなく特定のゲストアカウントを使用するようにシステムが設定されている場合に、ゲストアカウントの資格情報を変更する
- Identity Managerユーザアプリケーションの他のLDAPプロパティを表示する. これらの設定の値は、ユーザアプリケーションのインストール時に指定されます。

インストール時のゲストアカウントの設定によって、ユーザインタフェースに表示されるフィールドは異なります。ゲストアカウントを指定した場合、ユーザインタフェースにはそのアカウントの資格情報を更新できるフィールドが表示されます。LDAP パブリック匿名アカウントを使用するようにシステムが設定されている場合、ユーザインタフェースには次のメッセージが表示されます :「アプリケーションはパブリックな匿名アカウントを使用するように設定されています。特定のゲストアカウントを使用するには、ldap 環境設定ツールを使用して、ゲストアカウントを有効にします。」

LDAP 接続パラメータを管理する

1 ［アプリケーション環境設定］ページで、左側のナビゲーションメニューから *［LDAP 接続パラメータ］* を選択します。

［LDAP Connection Parameters］パネルが \'95\'5c 示されます。

2 必要に応じて設定の確認、あるいは変更を行います。詳細は、100 ページの 「変更可能な設定」を参照してください。

3 変更を適用する場合には、［Submit］をクリックします。

変更可能な設定

［LDAP 接続パラメータ］パネルでは、次の資格情報に関する設定を変更できます。

- アイデンティティボールト (LDAP プロバイダ) へ接続時の Identity Manager ユーザアプリケーション。
- ゲストアカウント (設定されている場合)。

資格情報の初期値は、インストール時に指定されます。これらのインストール値は、sys-configuration-xmldata ファイルに書き込まれます。［管理］ページでこれらの資格情報を変更した場合、変更内容はユーザアプリケーションのデータベースに保存されます。sys-configuration-xmldata には保存されません。データベースに値が書き込まれると、ユーザ

アプリケーションは sys-configuration-xmldata ファイルに書き込まれた値をチェックしなくなります。つまり、configupdate ユーティリティを使って資格情報を変更することはできません。その資格情報は無視されてしまうからです。ただし、configupdate ユーティリティを使って、ゲストユーザのタイプ (LDAP ゲストまたはパブリック匿名アカウント) を変更できます。

***表 5-3** LDAP パラメータ*

設定	操作
管理ユーザ名	アイデンティティボールトでフルの管理者権限を持つユーザの名前を入力します。Identity Manager ユーザアプリケーションは、管理者としてアイデンティティボールトにアクセスできる必要があります。 通常、アイデンティティボールトのルート管理者を LDAP 接続ユーザ名として指定します。ルート管理者はツリーをフルに制御できるため、トラスティ権利を特に割り当てる必要はありません。 例 : `cn=admin,o=myorg` その他のユーザを指定した場合、ユーザアプリケーションドライバのプロパティ「All Attributes Rights」および「Entry Rights」に継承可能なトラスティ権利を割り当てる必要があります。 **注 :** 混乱を回避するため、ユーザアプリケーションのユーザアプリケーション管理者を LDAP 接続ユーザ名として指定しないことをお勧めします。これら 2 つの目的には、別々のアカウントを使用することが適しています。
管理パスワード および [パスワードの確認]	アイデンティティボールトのユーザ名に現在設定されているパスワードを入力します。
ゲストユーザ名	ゲストユーザの識別名を入力します。
ゲストパスワードの確認	ゲストユーザのパスワードを入力します。

LDAP サーバで TLS を有効にしている場合、管理ユーザ名とパスワードを更新すると次のエラーが発生します :「LDAP プロバイダを認証できません。このエラーを修正するには、iManager で TLS を無効にしてください。」

5.1.4 ログの設定

[ログ] ページを使用すると、Identity Manager ユーザアプリケーションが生成するログメッセージのレベルを制御したり、これらのメッセージを Novell Audit® に送信するかどうかを指定したりすることができます。

Identity Manager ユーザアプリケーションは、Apache Software Foundation より配布されるオープンソースログパッケージである log4j を使用してログを行います。デフォルトでは、イベントメッセージは次の両方にログされます。

- Identity Managerユーザアプリケーションが展開されるアプリケーションサーバのシステムコンソール。
- Identity Managerr ユーザアプリケーションが展開されるアプリケーションサーバのログファイル。たとえば、:
  `jboss/server/IDM/log/server.log`

これはローリングログファイルです。特定のサイズに達すると、別のファイルにロールオーバーします。Novell Audit を含むように環境を設定した場合は、イベントメッセージをそこにも記録することができます。ログ環境および Novell Audit の設定の詳細については、69 ページの第 3 章「ログのセットアップ」を参照してください。

ログについて

［ログ］ページでは、ログが一覧表示されます。各ログは、Identity Manager ユーザアプリケーションの別々の部分からイベントメッセージを出力します。出力レベルはログごとに異なります。

ログ名は log4j 規則に基づきます。生成されるイベントメッセージに、メッセージ出力のコンテキストとともにログ名が \'95\'5c 示されます。

ログの詳細は、102 ページの 表 5-4 を参照してください。

表 5-4 *Identity Manager ユーザアプリケーションログ*

ログ名	説明
com.novell	他の Identity Manager ユーザアプリケーションログの親
com.novell.afw.portal.aggregation	ポータルページ処理に関連するメッセージ
com.novell.afw.portal.persist	ポータルデータ (ポータルページおよびポートレット登録を含む) の持続に関連するメッセージ
com.novell.afw.portal.portlet	ポータルコアポートレットおよびアクセサリポートレットからのメッセージ
com.novell.afw.portal.util	ポータルインポート／エクスポートおよびナビゲーションポートレットからのメッセージ
com.novell.afw.portlet.consumer	ポートレットレンダリングに関連するメッセージ
com.novell.afw.portlet.core	コアポートレット API に関連するメッセージ
com.novell.afw.portlet.persist	ポートレットデータ (ポートレット環境設定および設定値を含む) の持続に関連するメッセージ
com.novell.afw.portlet.producer	ポータル内のポートレットの登録および設定に関連するメッセージ
com.novell.afw.portlet.util	ポートレットにより使用されるユーティリティコードに関連するメッセージ
com.novell.afw.theme	テー \'83\'7d サブシステムからのメッセージ

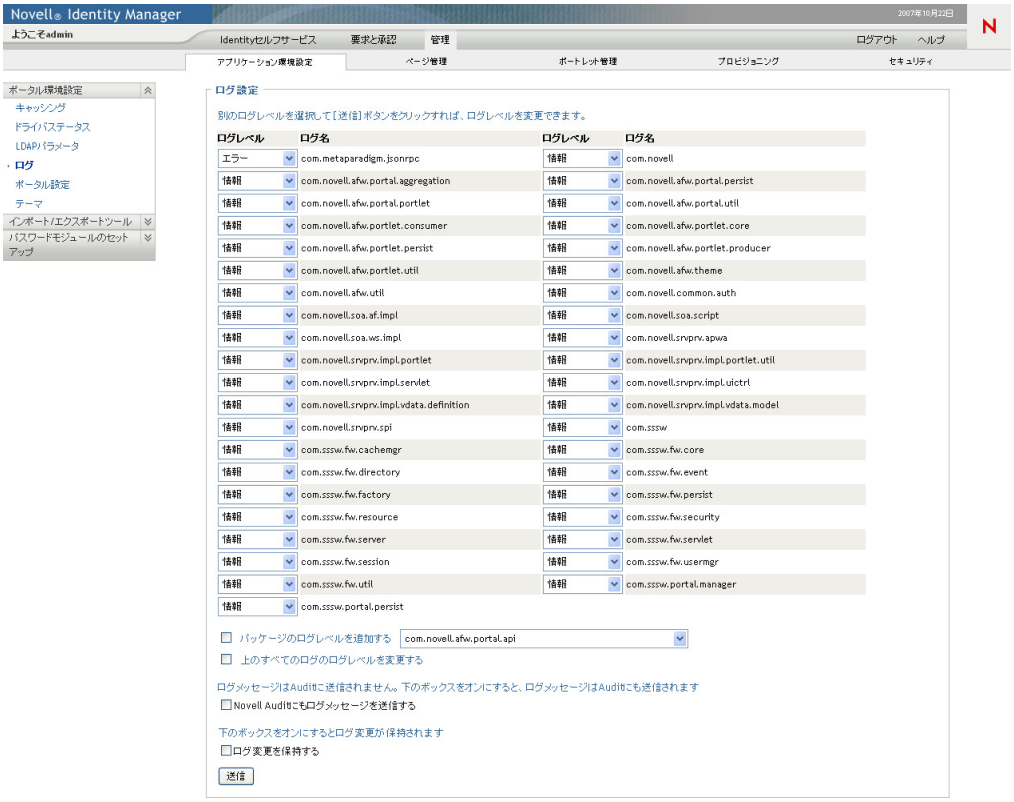
ログ名	説明
com.novell.afw.util	ポータルユーティリティクラスに関連するメッセージ
com.novell.soa.af.impl	承認フロー (プロビジョニングワークフロー) サブシステムからのメッセージ
com.novell.srvprv.apwa	「要求と承認」Web アプリケーション (アクションおよびタグ) からのメッセージ
com.novell.srvprv.impl.portlet.core	コア識別ポートレットおよびパスワードポートレットからのメッセージ
com.novell.srvprv.impl.portlet.util	識別関連ユーティリティポートレットからのメッセージ
com.novell.srvprv.impl.servlet	UI 制御フレームワークの Ajax サーブレットおよび Ajax サービスからのメッセージ
com.novell.srvprv.impl.uictrl	UI 制御レジストリおよび承認形式レンダリングからのメッセージ
com.novell.srvprv.impl.vdata	ディレクトリ抽象化層からのメッセージ
com.novell.srvprv.spi	UI 制御レジストリ API からのメッセージ
com.sssw.fw.cachemgr	フレームワークキャッシュサブシステムに関連するメッセージ
com.sssw.fw.core	フレームワークコアサブシステムに関連するメッセージ
com.sssw.fw.directory	フレームワークディレクトリサブシステムに関連するメッセージ
com.sssw.fw.event	フレームワークイベントサブシステムに関連するメッセージ
com.sssw.fw.factory	フレームワークファクトリサブシステムに関連するメッセージ
com.sssw.fw.persist	フレームワーク持続サブシステムに関連するメッセージ
com.sssw.fw.resource	フレームワークリ \'83\'5cースサブシステムに関連するメッセージ
com.sssw.fw.security	フレームワークセキュリティサブシステムに関連するメッセージ
com.sssw.fw.server	フレームワークサーバサブシステムに関連するメッセージ
com.sssw.fw.servlet	フレームワークサーブレットサブシステムに関連するメッセージ
com.sssw.fw.session	フレームワークセッションサブシステムに関連するメッセージ
com.sssw.fw.usermgr	フレームワークユーザサブシステムに関連するメッセージ
com.sssw.fw.util	フレームワークユーティリティサブシステムに関連するメッセージ
com.sssw.portal.manager	Portal Manager に関連するメッセージ
com.sssw.portal.persist	ポータル持続に関連するメッセージ

ユーザアプリケーションログは階層構造になっています。たとえば、com.novell はその下にある他のログの親となります。他のログは、そのプロパティを継承します。

ログレベルの変更

特定のログに設定されているレベルを変更することにより、そのログに書き込まれる情報量を制御できます。デフォルトでは、すべてログは［*Info*］*に設定されています。これは中間レベルです。*

1 ［Logging］ページに移動します。



2 ページ上部で、レベルを変更するログを検索します。

3 ドロップダウンリストを使用して次のいずれかのレベルを選択します。

レベル	説明
Fatal	詳細性は最低 . 致命的エラーをログに書き込みます。
エラー	エラー (上記すべてに加えて) をログに書き込みます。
Warn	警告 (上記すべてに加えて) をログに書き込みます。
Info	情報メッセージ (上記すべてに加えて) をログに書き込みます。
デバッグ	デバッグ情報 (上記すべてに加えて) をログに書き込みます。
トレース	詳細性は最高 . トレース情報 (および上記すべて) をログに書き込みます。

4 必要に応じて、他のログに対して、ステップ 2 とステップ 3 を繰り返します。

5 ［*送信*］をクリックします。

すべてのログのログレベルを同じ設定にする場合は、*［上のすべてのログのログレベルを変更する］*を選択して、ドロップダウンリストからレベルを選択します。

他のパッケージのログの追加

ユーザアプリケーションが使用する他のパッケージのログを追加できます。

1 ［Logging］ページに移動します。

2 ページ下部の *［パッケージのログレベルを追加する］*を選択し、次にドロップダウンリストからパッケージを選択します。

3 ドロップダウン リストからログレベルを選択し、次に *［送信］*をクリックします。

Novell Audit へのログメッセージの送信

［ログ］ページから、Identity Manager ユーザアプリケーションがイベントメッセージ出力を Novell Audit に送信するかどうかを制御できます。デフォルトでは、ユーザアプリケーションのインストール時に有効にしない限り、Novell Audit へのログは無効になっています。

Novell Audit ログのオン／オフを切り替える

1 ［Logging］ページに移動します。

2 必要に応じて、*［Novell Audit にもログメッセージを送信する］*の設定を選択／選択解除します。

3 ［*送信*］をクリックします。

ログ設定の保持

デフォルトでは、［ログ］ページで加えられた変更は、次にアプリケーションサーバが再起動されるか、ユーザアプリケーションが再展開されるまで有効です。それ以降は、ログ設定はデフォルト値に戻ります。

ただし、［ログ］ページには、設定に対する変更を保持できるオプションがあります。この機能を有効にすると、ログ設定値は、Identity Manager ユーザアプリケーションが展開されたアプリケーションサーバのログ環境設定ファイルに保存されます。例 :

- JBoss の場合、これは次のファイルになります。
  `jboss/server/IDM/conf/idmuserapp_logging.xml`
- WebSphere の場合、このファイルはカスタムプロパティ idmuserapp.logging.config.dir に従って指定されます。

設定の保持を有効または無効にする :

1 ［Logging］ページに移動します。

2 必要に応じて *［ログ変更を保持する］*を選択／選択解除します。

3 ［*送信*］をクリックします。

5.1.5 ポータル設定

［ポータル］ページでは、Identity Manager ユーザアプリケーションの特徴を参照できます。これらの情報は参考情報で、変更することはできません。これらの設定値は、ユーザアプリケーション WAR に設定されています。(*［デフォルトのテーマ］*には［テーマ］ページで選択している現在のテーマが反映されます)。

5.1.6 テーマ管理

［Themes］ページを使用して、Identity Managery ユーザインタフェースの外観や操作方法を制御できます。

「テーマ」とは外観上の特徴のセットで、ユーザインタフェース全体 (Guest ページ、ログインページ、*［Identity セルフサービス］*タブ、*［要求と承認］*タブ、および *［管理］*タブ) に適用されます。ユーザインタフェースでは常に 1 つのテーマだけが有効になっています。［Themes］ページでは、切り替えることができるよういくつかのテー \'83\'7d が用意されています。

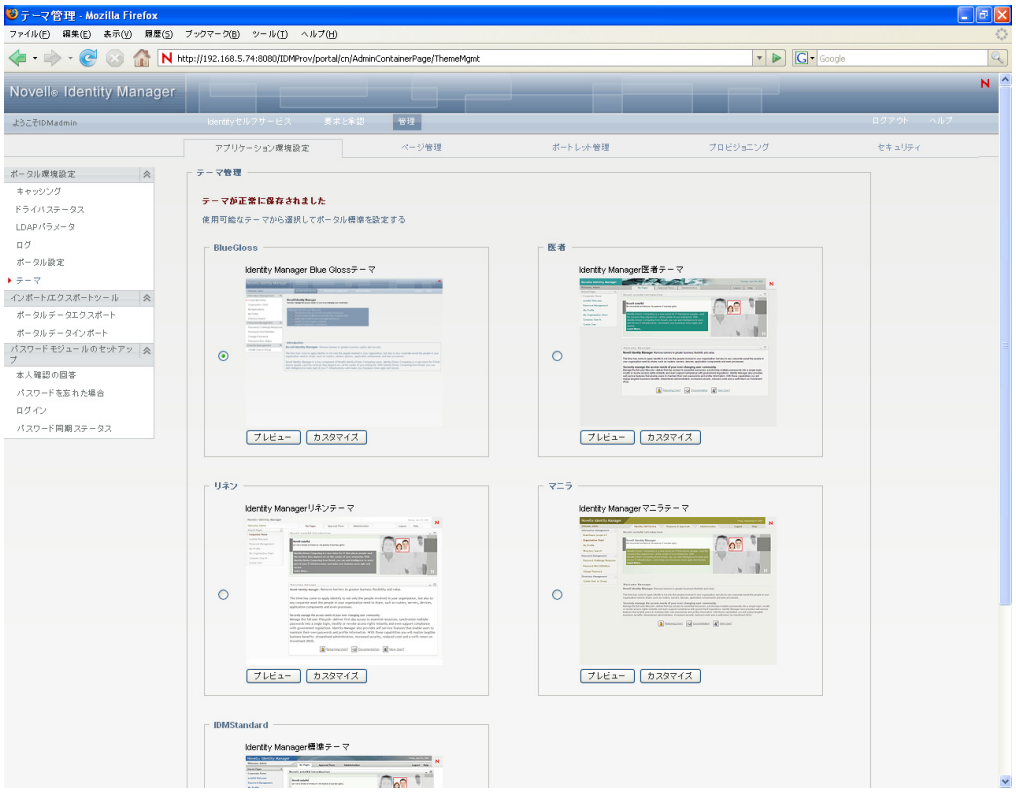
［Themes］ページでは、次のこともできます。

- 各テー \'83\'7d をプレビューして、どのように \'95\'5c 示されるか確認できます。
- 任意のテーマをカスタマイズして、ロゴなどのブランディングを付けることができます。

テーマのプレビュー

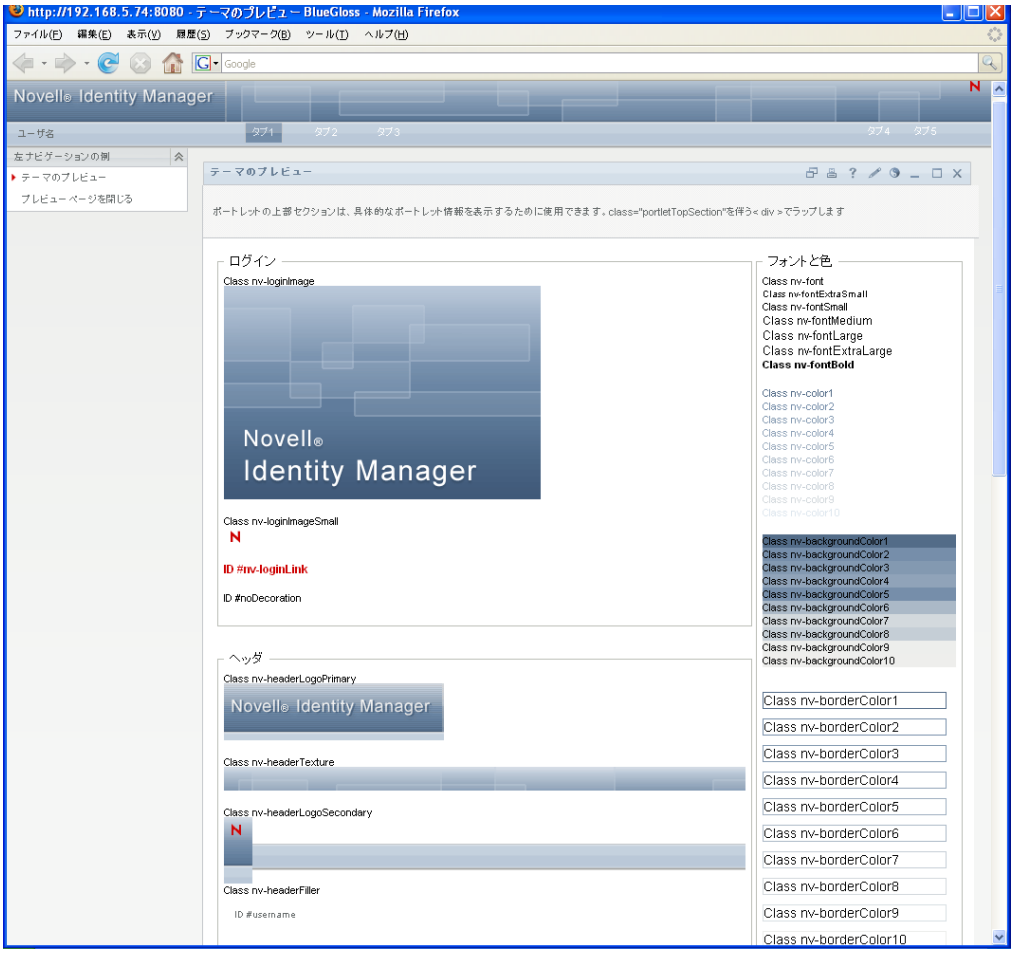
テー \'83\'7d を選択する前に、テー \'83\'7d によって Identity Manager ユーザインタフェースの外観がどのように変わるかプレビューできます。

1 ［Themes］ページに移動します。



2 プレビューするテーマを選び、そのテーマの *[プレビュー]* ボタンをクリックします。

新しいブラウザウィンドウにそのテー \'83\'7d のプレビューが \'95\'5c 示されます。



3 プレビューをスクロールして、テー \'83\'7d の特長を確認します。

4 確認できたら、*[プレビューページを閉じる]* (左上隅にあります) をクリックするか、手動でプレビューウィンドウを閉じます。

テーマの選択

気に入ったテー \'83\'7d が見つかったら、そのテー \'83\'7d を Identity Manager ユーザインタフェースの現在のテー \'83\'7d として選択できます。

1 ［Themes］ページに移動します。

2 使用するテー \'83\'7d のラジオ \'83\'7b タンをクリックします。

3 ［*Save*］ *\'83\'7b タンをクリックします。*

ユーザインタフェースの外観が、選択したテー \'83\'7d を反映して変更されます。

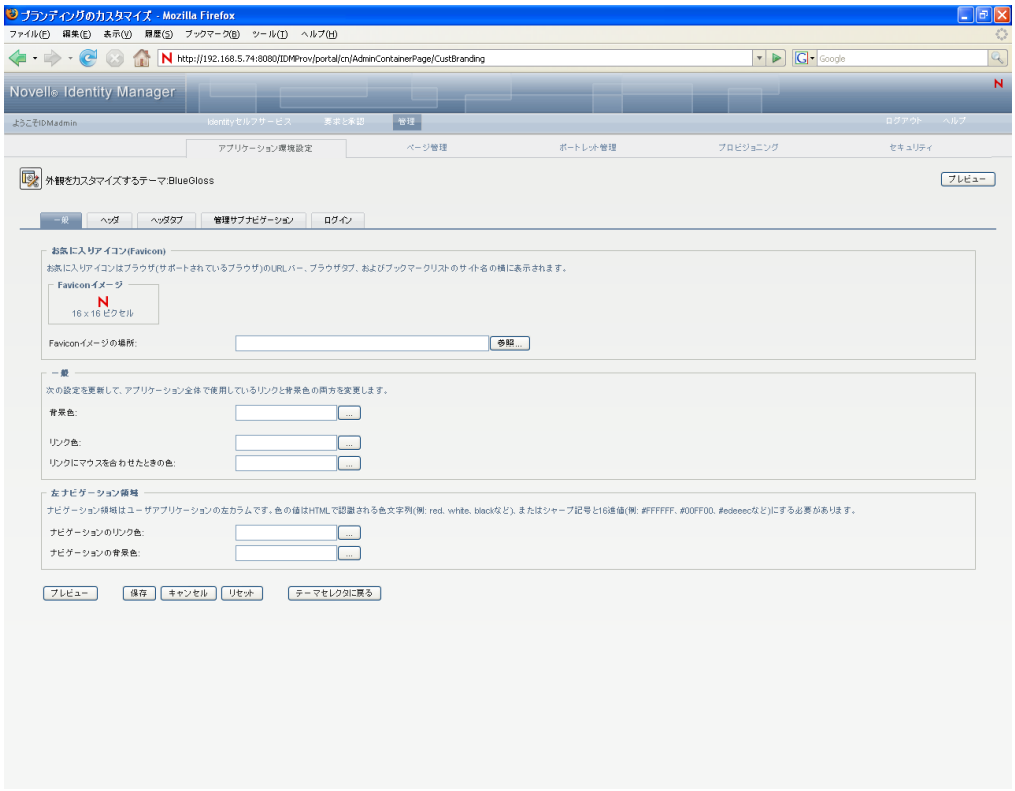
テーマのブランディングのカスタマイズ

どのテーマも、ユーザ独自の画像に入れ替えたり、色設定を変更したりしてカスタマイズできます。これにより、会社や組織のブランド設定に合わせて Identity Manager ユーザインタフェースをカスタム \'95\'5c 示できます。

1 ［Themes］ページに移動します。

2 カスタマイズするテーマを見つけ、そのテーマの［*カスタマイズ*］ボタンをクリックします。

［Themes］ページによって、そのテー \'83\'7d の［Customize Branding］設定が \'95\'5c 示されます。



3 必要に応じて、各タブの項目を設定、変更してください。各タブには、それぞれユーザアプリケーションインタフェースの異なる部分の設定項目がまとめられています。次のプロパティがあります。

- *一般*: お気に入りアイコン、リンクの色、リンクにマウスを合わせた時の色、および左ナビゲーション領域のプロパティなどの一般的なテーマプロパティを指定できます。
- *ヘッダ*: ヘッダの色、テクスチャ、ロゴ、およびユーザ名プロパティを指定できます。
- *ヘッダタブ*: ヘッダタブのプロパティを指定できます。
- *管理サブナビゲーション*: *［管理］*タブのプロパティを指定できます。
- *ログイン*: ログイン画面のプロパティを指定できます。

画面の指示に従って、各項目を設定してください。変更内容は保存しないと、ユーザアプリケーションには反映されません。変更内容を保存していない場合、変更内容が保存されてないことを知らせるアスタリスク (*) が *［保存］*ボタンに表示されます。

4 ［*保存*］をクリックします。

現在のテーマを編集した場合は、カスタマイズした内容が反映され、ユーザインタフェースの外観が変わります。テーマに対するカスタマイズをすべて取り消す場合は、*［リセット］*ボタンをクリックします。

5 このテーマの作業が完了したら、*[テーマセレクタに戻る]* ボタンをクリックします。

カスタムテーマの定義

独自のカスタムテーマを作成し、それを独自の WAR ファイルで配布できます。配布したカスタムテーマは、*[管理]* タブのテーマ管理ページから利用できます。独自のカスタムテーマを作成する前に、次のテクノロジに関する十分な知識と作業経験があることを確認してください。

- J2EE WAR ファイルの構造、WAR ファイルの内容の変更方法、ファイルのアプリケーションサーバへの配布方法。
- CSS ファイルと XML ファイルの変更方法
- テーマのグラフィック部分の作成方法

カスタムテーマの作成

カスタムテーマを作成するには、まずユーザアプリケーション WAR にある既存のテーマ (*BlueGloss* など) をコピーします。

1 配布したユーザアプリケーション WAR ファイル (IDM.WAR または IDMProv.WAR) を、インストールディレクトリにバックアップします (例 :/opt/novell/idm サブディレクトリ)。

2 テスト環境で、ユーザアプリケーション WAR ファイルの内容を抽出します。

ユーザアプリケーションのテーマを構成しているファイルは、resources\themes サブディレクトリにあります。各テーマはそれぞれ適切な名前の独立したディレクトリに保存されています。

3 テスト環境で、カスタムテーマ用のディレクトリを作成します。

任意の有効なディレクトリ名を使用できますが、テーマ名を反映した名前でなければなりません。また、スペースを使用することはできません。

4 テーマ BlueGloss の内容を、抽出した WAR ファイルから、作成したサブディレクトリにコピーします。次のファイルに対して作業を行います。

ファイル名	説明
theme.xml	テーマ記述子ファイルです。このファイルには、表示名や説明に関するエントリがあります。これは、*[管理]* タブの *[テーマ]* ページで使用されます。残りのエントリは、ブランド可能なセレクタに対応しています。これらのエントリの width 属性と height 属性は、ユーザがこれらのイメージのカスタマイズ版をアップロードした際に実際に必要な寸法をブランディングページで参照するために使用されます。これらのエントリは、該当するイメージと一致する必要があります (themes.css の幅と高さ)。
theme.css	ユーザインタフェースの外観を整えるために使用する CSS セレクタがあります。
print.css	プリントフレンドリ版のユーザインタフェースを整えるために使用する CSS セレクタがあります。
images サブディレクトリ	テーマが使用するイメージを保管します。

これらのファイルの作業規則 :

- theme.xml、theme.css、および print.css ファイルのファイル名は変更できません。
- CSS セレクタ名は同じでなければなりません。ただし、外観を設定するために、セレクタのプロパティは変更できます。
- images サブディレクトリには任意の名前を使用できます。ただし、CSS ファイルと XML ファイル内で正しい名前を指定して参照させる必要があります。

5 必要に応じてイメージ、CSS スタイルシート、および他の要素を変更します。次の項目を変更することをお勧めします。

- theme.xml:
  - **display-name:** 自分のテーマを表す名前に変更します。ここで指定した名前が、ユーザアプリケーションの［*管理*］タブにある［テーマ］ページに表示されます。
  - **description:** この値には、テーマの説明を指定します。ここで指定した説明が、ユーザアプリケーションの *［管理］* タブにある［テーマ］ページの説明として表示されます。
  - *display-name* および *Description* フィールドをローカライズするかどうかを検討してください。
- graphics ディレクトリ :
  - **thumbnails.gif:** これを自分のイメージと置換します。イメージは、*［管理］* タブの［テーマ］ページに前述のテーマ名や説明と一緒に表示されます。一般的にこのイメージは、対応するテーマの適用時のユーザアプリケーションのランディングページの外観を表します。
  - **グラフィックファイルの名前変更 :** グラフィックファイルの名前を変更した場合 (同名の別の画像を使用するのではなく)、theme.xml ファイルと theme.css ファイルに指定されているイメージ参照が、正しいファイル名を指していることを確認してください。ブランディングインタフェースでイメージを使用しない場合は（例 :theme.xml ファイル内のブランド可能イメージのサブセットとして指定しない場合）、theme.css ファイル内のイメージへの参照を変更するだけで構いません。images/header_left.gif の名前を images/my_company_name.gif に変更する場合を考えてみましょう。この場合、theme.css ファイルを編集して、新しいイメージ名を指定します。

6 テーマファイルの変更がすべて完了したら、カスタマイズしたテーマのディレクトリを、カスタムテーマを保管する WAR ファイルに追加します。新しい WAR ファイルをテスト用アプリケーションサーバに展開します。テストのヒント :［テーマ］ページを開きます (*［管理］* タブ)。ここには、付属のテーマと一緒に作成したテーマが表示されます。［テーマのプレビュー］アクションを使って、カスタマイズしたテーマがどのように表示されるかを確認してください。これは、テーマを意図した通りに変更できたかどうかを確認するために役立ちます。

7 変更内容をテストし、正しく変更できたことを確認したら、カスタムテーマを含む WAR ファイルを実働環境のアプリケーションサーバに展開できます。

1 つの WAR ファイルには、複数のカスタムテーマを入れることができます。カスタムテーマを保管した WAR ファイルは複数を展開することもできます。

テーマの公開を中止するには、アプリケーションサーバの deploy ディレクトリから、該当するテーマのある WAR ファイルを削除してください。公開を中止する前に、その WAR ファイルに含まれているテーマがユーザアプリケーションのデフォルトテーマとして定義されていないことを確認してください。デフォルトテーマがある WAR ファイルを

削除した場合、［テーマ管理］画面にエラーメッセージが表示され、ユーザアプリケーションのテーマがインストール時に指定されたデフォルトのテーマに戻ります。

外部パスワード WAR 用テーマのカスタマイズ

パスワード管理で *［外部パスワード WAR］* を使用するように設定した場合、その外部パスワード WAR に［パスワードを忘れた場合］ページのテーマが定義されます。外部パスワード WAR のデフォルト名は、IDMPwdMgt.WAR になります。デフォルトでは、IDMPwdMgt.WAR には 1 つのテーマ *BlueGloss* があります。これには、このテーマを変更したり、ブランディングするユーザインタフェースは含まれていません。

外部の［パスワードを忘れた場合］ページにはカスタムテーマを設定できます。カスタムテーマの定義方法については、110 ページの 「カスタムテーマの定義」を参照してください。ただし、外部の［パスワードを忘れた場合］ページの展開方法は異なっており、カスタムテーマ WAR の規則もより厳格になっています。カスタムテーマの定義後 :

- WAR 内の IDMPwdMgtTheme.WAR と言う名前のテーマをパッケージ化します。
- IDMPwdMgtTheme.WAR には 1 つのテーマを入れることができます。このテーマは、WAR 内の resource/themes/Theme ディレクトリになければなりません。
- 外部 WAR があるアプリケーションサーバに IDMPwdMgtTheme.WAR を展開します。1 回に 1 つのカスタムテーマしか展開できません。

5.2 インポート／エクスポートツールでの作業

［ツール］ページを使用すると、Identity Manager ユーザアプリケーションが使用するポータルコンテンツ (ページおよびポートレット) をエクスポート／インポートできます。このコンテンツは *ポータル設定状態* としても知られ、次のものが含まれます。

- コンテナおよび共有ページ (各ページの割り当て済みポートレット、各ポートレットの環境設定および設定など)
- ポートレット登録

表 5-5 *ポータルデータエクスポート／インポートツール*

ツール	動作方法
Portal Data Export	選択したコンテナページ、共有ページ、およびポートレットの XML 記述を生成します。XML ファイルはポータルデータエクスポート ZIP ファイルに格納され、ポータルデータインポートツールへの入力に使用できます。
Portal Data Import	ポータルデータエクスポートの ZIP ファイルを入力データとして受け取ります。ポータルデータエクスポートの ZIP ファイルを使用して、ポータル (ユーザアプリケーション) 内にコンテナページ、共有ページ、およびポートレットを生成します。

エクスポートおよびインポートのためのツールを使用すると、必要に応じて、1 つのポータル (ユーザアプリケーション) から別のポータルにポータル設定状態を移動できます。これらのツールの仕組みについては、112 ページの 表 5-5 を参照してください。

ポータルデータエクスポートツールとポータルデータインポートツールは、次の用途に使用できます。

- テスト (\'83\'5cース) 環境から運用 (ターゲット) 環境に、ポータル設定状態を移動する
- ポータルの設定状態の増分を更新する
- ポータルをコピーする
- (オプション) ターゲットポータルの設定状態を上書きする

5.2.1 要件

ポータルデータエクスポートツールおよびポータルデータインポートツールを使用するには、ソースおよびターゲットのアプリケーションサーバ上で、Identity Manager ユーザアプリケーション (ポータル) が展開および実行されている必要があります。

ソースサーバとターゲットサーバが同じアイデンティティボールトにアクセスしている必要はありません。適切であれば、異なるアイデンティティボールトにアクセスしていても問題ありません。アイデンティティボールト内のユーザ、グループおよびコンテナが同じである必要もありません。

5.2.2 制限

ポータルデータエクスポートツールとポータルデータインポートツールは、次の用途には使用できません。

- サーバによるユーザ要求の処理中のポータル設定状態のエクスポートまたはインポート
- ポータルのクラスおよびリ \'83\'5cースのエクスポートまたはインポート
- ポートレットのクラスおよびリ \'83\'5cースのエクスポートまたはインポート
- ポータルで使用されている識別およびプロビジョニングデータのエクスポートおよびインポート
- ページおよびポートレット以外の管理設定のエクスポートおよびインポート
- 古いバージョンのポータルから新しいバージョンのポータルへの設定状態の移行 (ポータルは同じバージョンである必要あり)

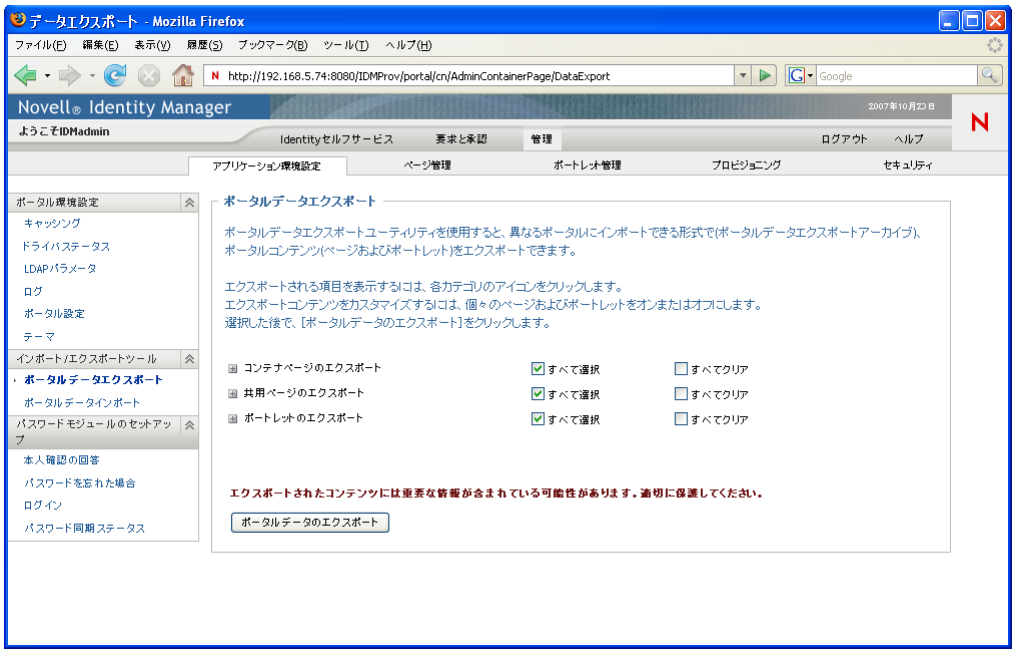
5.2.3 ポータルデータのエクスポート

この節では、ポータル環境設定状態をポータルデータエクスポートの ZIP ファイルにエクスポートする方法について説明します。

1 増分更新を実行する場合、ターゲットポータルのバックアップを作成します。

2 ［アプリケーション環境設定］ページで、左側のナビゲーションメニューから *［ポータルデータエクスポート］* を選択します。

［Portal Data Export］パネルが \'95\'5c 示されます。



3 画面の指示に従い、エクスポートするポータルページおよびポートレットを選択します。

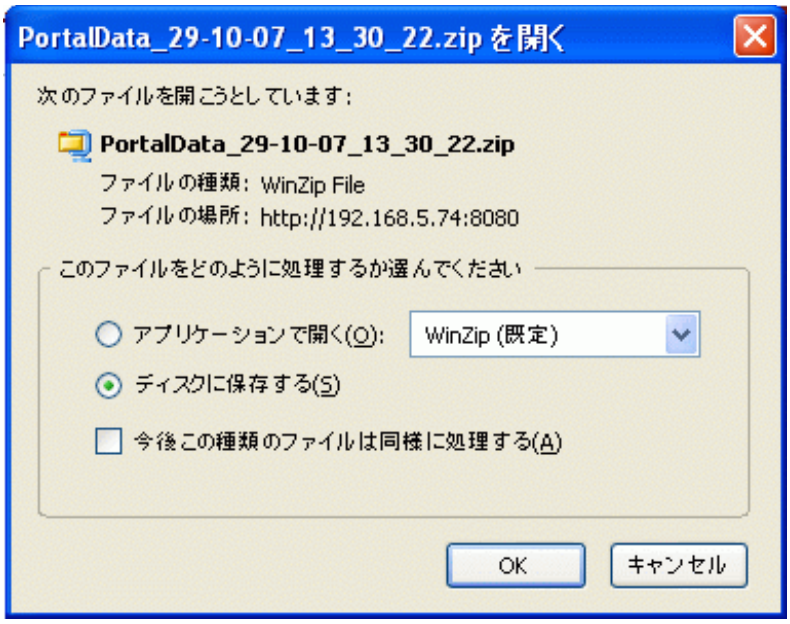
エクスポート対象として選択しなかったポートレットの中には、それでもエクスポートされるものがあります。ポートレットを含むページをエクスポートする場合、そのポートレットをエクスポート対象として選択しないときでも、(エクスポートされたページについてランタイムエラーが発生しないことを確認するため) ポートレットはエクスポートされます。

4 選択が完了したら、*[ポータルデータのエクスポート]* ボタンをクリックします。

新しいポータルデータエクスポートの ZIP ファイルが作成され、現在の日付および時刻を含むデフォルトの名前が付けられます。例 :

```
PortalData.21-Oct-05.09.12.16.zip
```

この ZIP ファイルをローカルに保存する (または適切なアーカイブユーティリティで開く) ように要求するメッセージが表示されます。例 :



5 ポータルデータエクスポートの ZIP ファイルを適切な場所に保存します。

5.2.4 ポータルデータのインポート

この節では、ポータルデータエクスポートの ZIP ファイルをポータルにインポートする方法について説明します。

注 : インポート中、ターゲットアプリケーションサーバは実行中で、またユーザ要求の処理中ではないことが必要です。

---

1 増分更新を実行する場合、ターゲットポータルのバックアップを作成します。

2 ［Tools］ページで、左側のナビゲーションメニューから［*Portal Data Import*］*を選択します。*

［Portal Data Import］パネルが \'95\'5c 示されます。

3 次のインポートについての一般設定を指定します。

設定	操作
アーカイブ	*[参照]* をクリックし、インポートするポータルデータエクスポートの ZIP ファイルを選択します。例 : `PortalData.21-Oct-05.09.12.16.zip`
Import security settings?	次のいずれか 1 つを選択します。 ◆ *はい*: ユーザ、グループ、およびコンテナによるページおよびポートレットへのアクセスとして、ポータルデータエクスポート ZIP ファイルに指定している許可をインポートする場合。関連するユーザ、グループ、およびコンテナがターゲットポータルのアイデンティティボールトに存在することを確認してください。存在しないエンティティの許可はインポートできません。 ◆ *いいえ*: ポータルデータエクスポート ZIP ファイルに指定している許可を無視する場合。

4 *[インポートアーカイブの表示]* をクリックします。

このパネルには、選択したポータルデータエクスポートの ZIP ファイルについての詳細情報とインポート方法が表示されます。

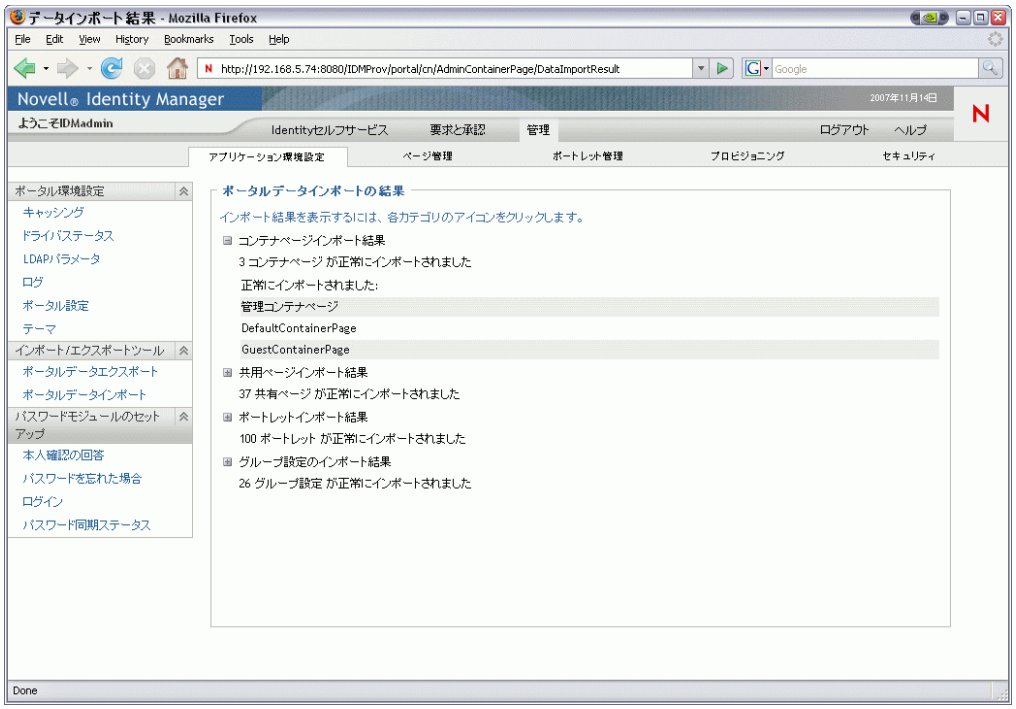
5 次のインポートについての詳細設定を指定します。

設定	操作
Replace existing data?	次のいずれか 1 つを選択します。 • *はい*: ターゲットポータルにすでに存在するページおよびポートレットの内容を、ポータルデータエクスポート ZIP ファイルの対応する内容で上書きする場合。たとえば、ポータルデータエクスポートの ZIP ファイルに MyPage という名前の共有ページがあり、ターゲットポータルにも MyPage という名前の共有ページがある場合、ターゲットポータルの既存のページは上書きされます。 • *いいえ*:既存のページおよびポートレットすべてについて、インポートをスキップする場合。
Access level for imported objects	次のいずれか 1 つを選択します。 • *すべてのユーザ*: インポートされたページおよびポートレットへのアクセスを制限しない場合。 • *管理者のみ*: インポートされたページおよびポートレットへのアクセスを制限する場合。 セキュリティ設定のインポートを選択した場合、このアクセスレベルは、セキュリティ設定をインポートできなかった、インポートされたページおよびポートレットについてのみ適用されます (通常、指定したユーザ、グループ、コンテナはターゲットポータルのアイデンティティボールトに存在しないため)。 セキュリティ設定のインポートを選択しなかった場合、このアクセスレベルは、インポートされたページおよびポートレットすべてに適用されます。

設定	操作
Import group settings?	(セキュリティ設定のインポートを指定する場合) 次のいずれか 1 つを選択します。 ◆ *はい*: ポータルデータエクスポート ZIP ファイルがグループについて指定する、デフォルトのコンテナページおよびデフォルトの共有ページ割り当てをインポートする場合。関連するグループがターゲットポータルのアイデンティティボールトに存在することを確認してください。存在しないグループの割り当てはインポートできません。 ◆ *いいえ*: ポータルデータエクスポート ZIP ファイルがグループについて指定する、デフォルトのページ割り当てを無視する場合。
Import Container Pages *Import Shared Pages* *Import Portlets*	画面の指示に従い、ポータルデータエクスポートの ZIP ファイルからターゲットポータルにインポートするページおよびポートレットを選択します。 **注 :** インポート対象として選択しなかったポートレットの中には、それでもインポートされるものがあります。ポートレットを含むページをインポートする場合、そのポートレットをインポート対象に含めない場合も、インポートされたページについてランタイムエラーが発生しないことを確認するため、ポートレットがインポートされます。
アーカイブ内のポートレットアプリケーション名をローカルサーバ上の既存のポートレットアプリケーションにマップしてください	*[アーカイブ]* **ドロップダウンメニューおよび** *[ローカル]* ドロップダウンメニューを使用して、アーカイブ (ポータルデータエクスポートの ZIP ファイル) 内のポートレットアプリケーション名を、ローカル (ターゲット) アプリケーションサーバ上の既存のポートレットアプリケーションにマップします。

6 インポートを開始できる準備が整ったら、*[ポータルデータのインポート]* をクリックします。

インポートが完了すると、［Portal Data Import Results］パネルが \'95\'5c 示されます。



失敗したインポートは赤く表示されます。インポートまたはエクスポートに関する問題をトラブルシューティングするには、アプリケーションサーバのシステムコンソールまたはログファイル (jboss/server/IDM/log/server.log など) を確認し、次のユーザアプリケーションログからのメッセージを探します。

```
com.novell.afw.portal.util
```

7 ターゲットポータルに対象のデータがインポートされたことををテストします。

5.3 パスワード管理の環境設定

この節では、パスワードセルフサービスおよびユーザ認証機能を Identity Manager ユーザアプリケーションに設定する方法について説明します。主なトピックは次のとおりです。

- 120 ページのセクション 5.3.1「パスワード管理機能について」
- 123 ページのセクション 5.3.2「本人確認の回答の設定」
- 125 ページのセクション 5.3.3「「パスワードを忘れた場合」の環境設定」
- 128 ページのセクション 5.3.4「ログインの環境設定」
- 136 ページのセクション 5.3.7「パスワードの変更の環境設定」
- 131 ページのセクション 5.3.5「パスワード同期ステータスの環境設定」
- 135 ページのセクション 5.3.6「パスワードのヒントの変更の環境設定」
- 136 ページのセクション 5.3.7「パスワードの変更の環境設定」

5.3.1 パスワード管理機能について

Identity Manager ユーザアプリケーションがサポートするパスワード管理機能には、ユーザ認証とパスワードセルフサービスがあります。これの機 \'94\'5c を使用できるようにすると、アプリケーションで次のことが行われます。

- Novell eDirectory™ の認証のための*ログイン*情報 (ユーザ名およびパスワード）の入力を要求するプロンプトが表示される。
- パスワード変更のセルフサービスをユーザに提供する
- パスワードを忘れた場合のセルフサービス (本人確認の回答の入力を促すメッセージの表示、パスワードヒントの表示、パスワード変更の許可など) をユーザが利用できるようにする。パスワードを忘れた場合のセルフサービスをファイアウォール内で実行するように設定することも、ファイアウォール外で実行するように設定することもできます。
- チャレンジ質問のセルフサービスをユーザに提供する
- パスワードヒントのセルフサービスをユーザに提供する

eDirectory に必要な設定

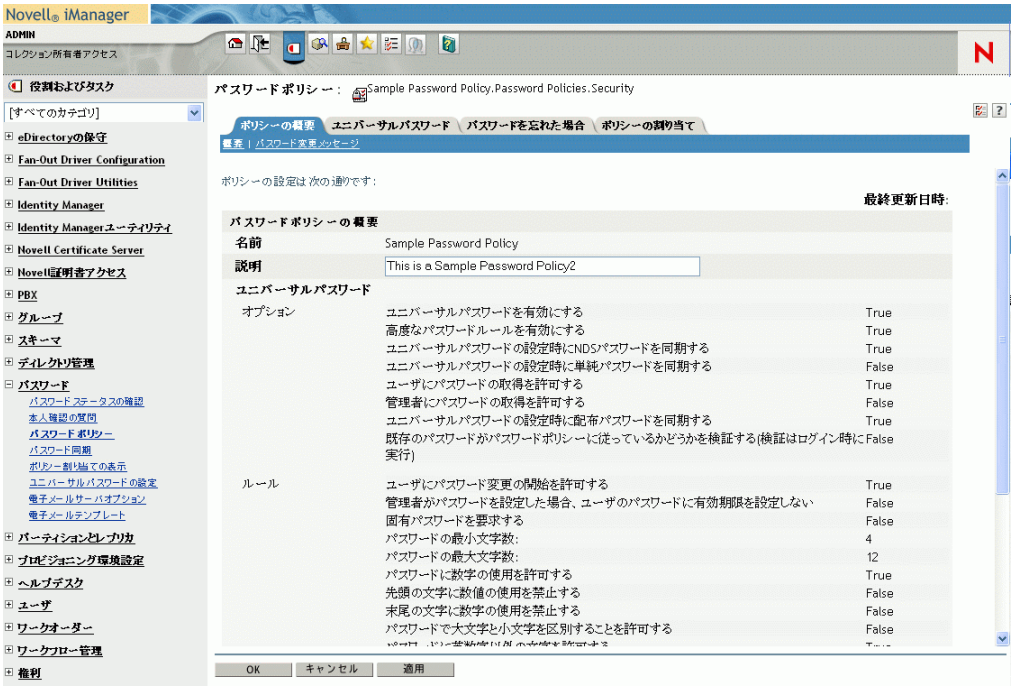
パスワードセルフサービスおよびユーザ認証の機 \'94\'5c のほとんどでは、使用する前に eDirectory で次の手順を実行する必要があります。

- ユニバーサルパスワード*を有効にする*
- 1 つまたは複数の Password Policy(パスワードポリシー) を作成する
- ユーザに適切な Password Policy(パスワードポリシー) を割り当てる

パスワードポリシーは管理者が定義するルールのコレクションで、ユーザパスワードの作成および変更時に基準を指定する目的で使用されます。Novell Identity Manager では、NMAS™ (Novell Modular Authentication Service) を利用して、管理者が eDirectory のユーザに割り当てるパスワードポリシーを強制します。

必要な設定手順を実行するには、Novell iManager を使用します。たとえば、［DocumentationPassword Policy］では次のように定義します。

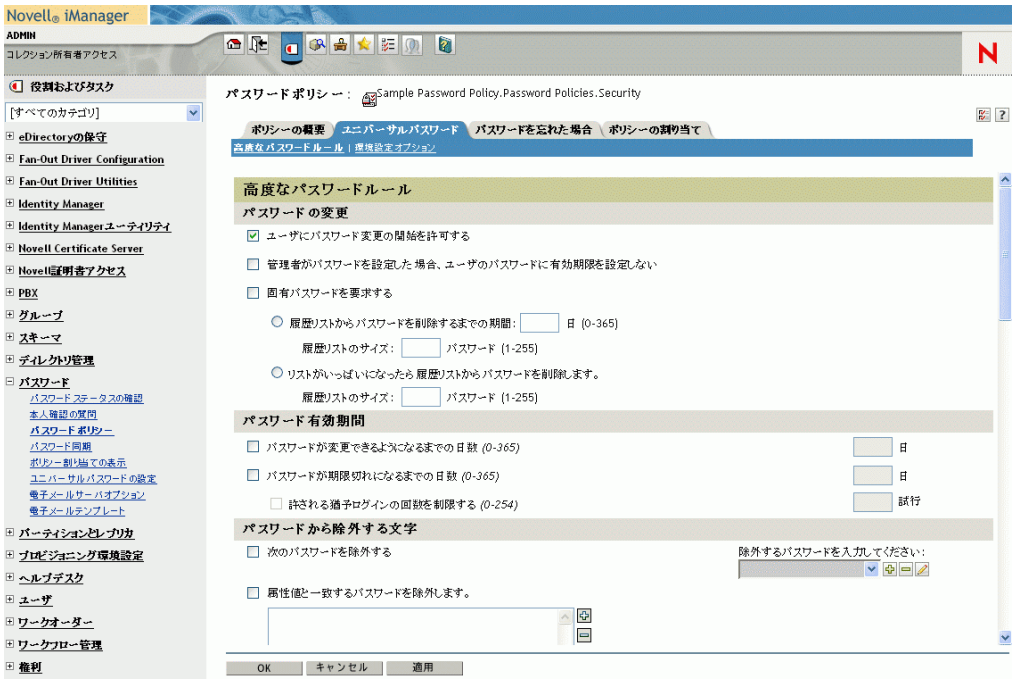
図 5-2　サンプルのパスワードポリシー



この Password Policy(パスワードポリシー) は次のものを指定しています。

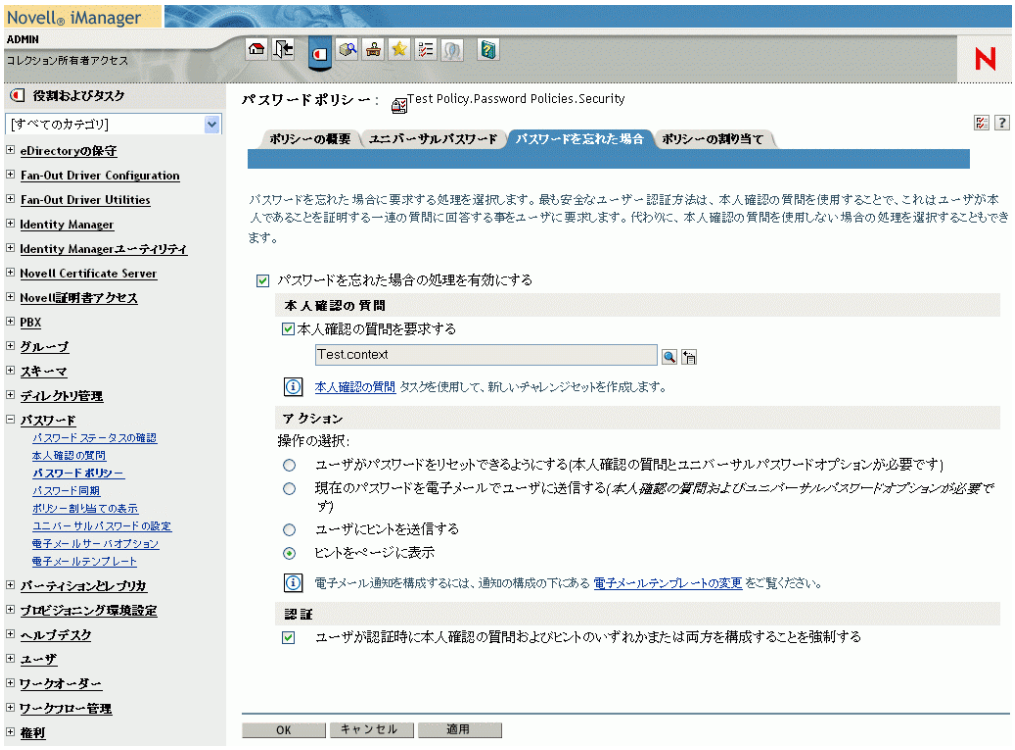
- ［Universal Password］設定

図 5-3　サンプルのユニバーサルパスワード設定



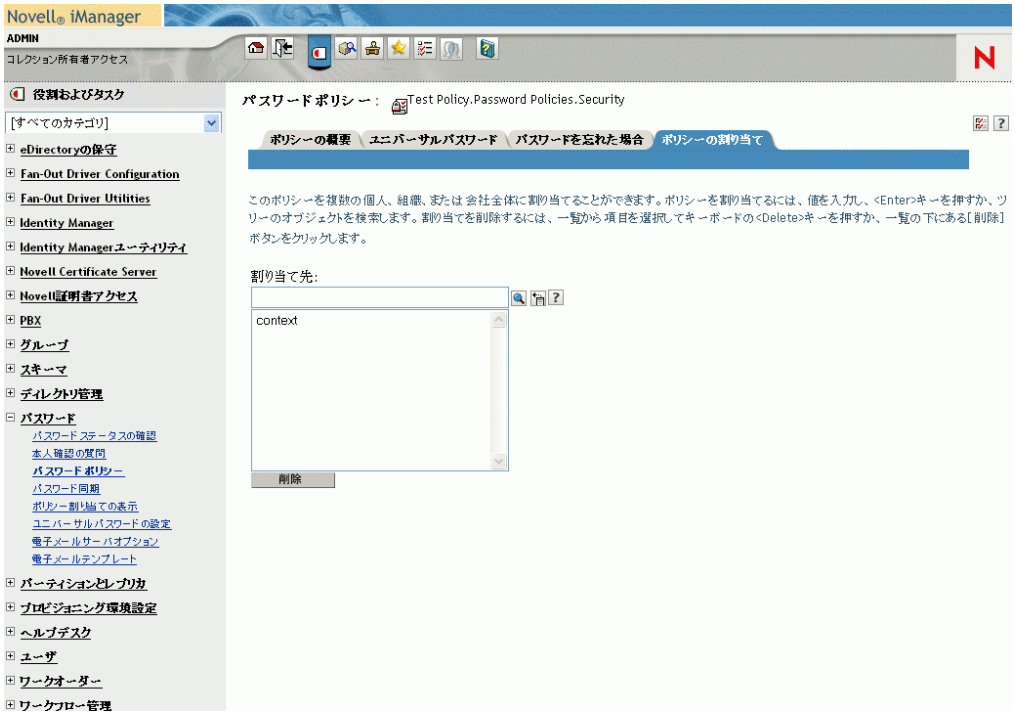
- パスワードを忘れた場合の処理方法の設定

図 5-4 *サンプルのパスワードポリシー*



- ポリシーを特定のユーザに適用する割り当て

図 5-5 *サンプルのポリシー割り当て*



eDirectory におけるユニバーサルパスワードおよび Password Policy (パスワードポリシー) の設定の詳細については、『Novell Identity Manager Administration Guide (http://www.novell.com/documentation/dirxml20/index.html)』を参照してください。

大文字と小文字を区別するパスワード

デフォルトでは、パスワードで大文字と小文字は区別されません。大文字と小文字を区別するパスワードを導入するためのパスワードポリシーを作成できます。*［パスワードポリシー］>［ユニバーサルパスワード］>［高度なパスワードルール］*を選択して、*［パスワードで大文字と小文字を区別することを許可する］*を指定できます。大文字と小文字を区別するパスワードを許可した場合、*［ユーザにパスワードの取得を許可する］*の設定も有効にする必要があります。デフォルトでは、このオプションは有効になっています。有効になっているかどうかを確認するには、iManager で *［パスワードポリシー］>［ユニバーサルパスワード］>［環境設定オプション］*を選択してください。

パスワードポリシーコンプライアンス

ユニバーサルパスワードを有効にした場合、既存のパスワードがパスワードポリシーに準拠しているかどうかを確認するようにシステムを設定することをお勧めいたします。設定するには、iManager を使用します。iManager で、*［パスワード］>［パスワードポリシー］>［ユニバーサルパスワード］>［環境設定オプション］*を選択します。*［既存のパスワードがパスワードポリシーに従っているかどうかを検証する (検証はログイン時に実行)］*オプションが選択されていることを確認します。これにより、ユーザアプリケーションで作成されたユーザには［パスワードの変更］ページが表示されます。ここでは、Identity Manager パスワードポリシーに準拠したパスワードを入力します。

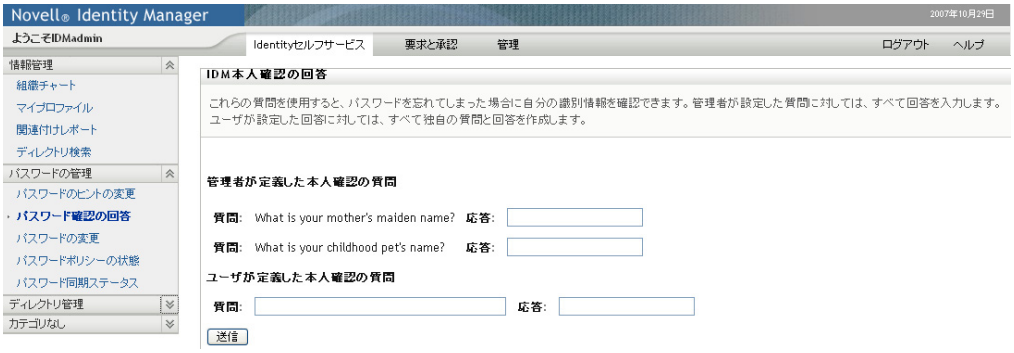
5.3.2 本人確認の回答の設定

本人確認の回答の設定セルフサービスページでは、次の作業を行えます。

- 管理者定義のチャレンジ質問に対する有効な回答の設定、およびユーザ定義のチャレンジ質問および回答の設定
- 管理者定義のチャレンジ質問に対する有効な回答の変更、およびユーザ定義のチャレンジ質問および回答の変更

ヒント : iManager で本人確認の回答をローカライズした場合、*［ログイン環境設定］*ロケールチェックを有効にするに True を設定します。

図 5-6 *本人確認の回答の例*



要件

本人確認の回答の要件については、124 ページの 表 5-6 を参照してください。

表 5-6 *本人確認の回答の要件*

トピック	要件
パスワードポリシー	パスワードを忘れた場合のセルフサービスが有効で、本人確認の質問と回答を設定する必要があります。
ユニバーサルパスワード	ユニバーサルパスワードを有効にする必要はありません。
eDirectory の設定	ログインユーザが属すコンテナの LDAP 管理者にスーパバイザ権が付与されている必要があります。これらの特権を付与されると、ユーザは、チャレンジ回答をシークレットストアに書き込むことができます。 たとえば、LDAP レルム管理者が cn=admin, ou=sample、n=novell と設定されており、ユーザが cn=user1、ou=testou、o=novell としてログインする場合を考えてみましょう。この場合、cn=admin、ou=sample、n=novell を *testou* のトラスティとして割り当て、*[All attribute rights]* でスーパバイザ権を付与する必要があります。

「本人確認の回答」機能の使用

本人確認の回答機能を使用するには、次のことを理解しておく必要があります。

- 124 ページの 「ログイン時の「本人確認の回答」機能の動作」
- 124 ページの 「ユーザアプリケーション上での「本人確認の回答」機能の動作」

ログイン時の「本人確認の回答」機能の動作

ログイン中、ユーザが本人確認の質問および回答を設定する必要がある場合には必ず、［ログイン］ページから自動的に「本人確認の回答」にリダイレクトされます。たとえば、管理者が iManager でユーザにパスワードポリシーを割り当てた後に初めてそのユーザがアプリケーションにログインしようとする場合がこれに当たります。Password Policy(パスワードポリシー) は忘れたパスワードを有効にし、チャレンジセットを含める必要があります)。

ユーザアプリケーション上での「本人確認の回答」機能の動作

デフォルトでは、本人確認の質問および回答を変更するためのセルフサービスをユーザアプリケーションで使用できます。

本人確認の回答の設定

本人確認の回答の環境設定に (*［管理］タブ*) ついては、次の表を参照してください。

表 5-7　「本人確認の回答」の環境設定

設定	説明
応答テキストのマスク	［はい］を選択すると、ユーザが入力した応答テキストはアスタリスク (*) で表示されます。

5.3.3 「パスワードを忘れた場合」の環境設定

この機能は、ユーザに各自のパスワードに関する情報を提供するために、「本人確認の回答」認証を使用します。結果は割り当てられたパスワードポリシーにより異なりますが、次にその例を示します。

- ユーザのパスワードヒントの画面への \'95\'5c 示
- ユーザへのヒントを電子メールで送信する
- ユーザのパスワードを電子メールで送信する
- パスワードのリセット (変更) を要求するプロンプト \'95\'5c 示

一般的に「パスワードを忘れた場合」セルフサービスは、配布されたユーザアプリケーション WAR を通じてファイアウォール内で利用できます。ただし、「パスワードを忘れた場合」管理機能を個別のパスワード管理 WAR に保管するようにシステムを設定することもできます。その後、ファイアウォール内外にある個別のシステムにパスワード管理 WAR を配布します。コアユーザアプリケーション WAR 外での「パスワードを忘れた場合」機能の設定方法については、46 ページのセクション 2.5「「パスワードを忘れた場合」の環境設定」を参照してください。

要件

「パスワードを忘れた場合」機能の要件については、126 ページの 表 5-8 を参照してください。

表 5-8 「パスワードを忘れた場合」機能の要件

トピック	要件
パスワードポリシー	「パスワードを忘れた場合」セルフサービスが有効で、本人確認の質問と回答を設定する必要があります。 パスワードポリシーを使用する場合、ユーザが初めてログインする時にパスワードの変更を要求するメッセージを表示するために、iManager の［パスワードポリシー］ページで次の設定を行う必要があります。 ◆ *［ユーザが認証時に本人確認の質問およびヒントのいずれかまたは両方を構成することを強制する］*を有効にする必要があります。この設定は、［認証］の［パスワードを忘れた場合］パネルにあります。 ◆ *［既存のパスワードがパスワードポリシーに従っているかどうかを検証する (検証はログイン時に実行］*を有効にする必要があります。この設定は、［環境設定オプション］>［認証］の［ユニバーサルパスワード］パネルにあります。 ◆ *［許される猶予ログインの回数を制限する］*(0 ～ 254) を有効にする必要があります。デフォルト値の 6 をそのまま使用することもできます。この設定は、［高度なパスワードルール］>［パスワード有効期間］の［ユニバーサルパスワード］パネルにあります。この設定は、ユーザの作成アクションをサポートするために必要です。ユーザの作成アクションは、ユーザのパスワードを失効させ、猶予ログインの値を 1 に設定します。そのため、初回ログイン時に、ユーザにパスワードの変更を強制することができます。
ユニバーサルパスワード	パスワードのリセットやパスワードのユーザへの電子メール送信をサポートするのではない限り、ユニバーサルパスワードを有効にする必要はありません。

「パスワードを忘れた場合」機能の使用

「パスワードを忘れた場合」機能を使用するには、次のことを理解しておく必要があります。

- 126 ページの 「ログイン時の「パスワードを忘れた場合」機能の使用方法」
- 127 ページの 「電子メールアクション用環境の設定」
- 127 ページの 「「パスワードを忘れた場合」の環境設定」

ログイン時の「パスワードを忘れた場合」機能の使用方法

ログイン処理中に、ユーザが *［パスワードを忘れた場合］*リンクをクリックすると、［ログイン］ページから［パスワードを忘れた場合］ページにリダイレクトされます。Forgot Password］が \'95\'5c 示されるとき、次のことが行われます。

1. ［username］の入力を要求するプロンプトが \'95\'5c 示されます。
2. ユーザの本人確認の回答認証を行うために、［本人確認の回答］ページにリダイレクトされます。

3. 認証されたユーザに割り当てられているパスワードポリシーで指定された、*パスワードを忘れた場合*アクションを実行します。次のいずれか 1 つを実行します。
   - ユーザがパスワードをリセットするために、［パスワードの変更］ページにリダイレクトする
   - パスワードまたはヒントを、ユーザに電子メールで送信する
   - ヒントを \'95\'5c 示する

電子メールアクション用環境の設定

パスワードを忘れた場合の電子メールによるアクションをサポートする場合、電子メール通知サーバを適切に設定する必要があります。

1. Web ブラウザを使用して eDirectory サーバの iManager にアクセスし、管理者としてログインします。
2. *［Roles and Tasks (役割とタスク)］>［パスワード］* の順にクリックし、*［電子メールサーバオプション］* を選択します。
3. 適切な設定を指定し、［OK］*をクリックします。*

「パスワードを忘れた場合」機能は、2 種類の電子メールテンプレートを使用します。iManager の、*［Roles and Tasks（役割とタスク）］>［パスワード］>［Edit Email Templates(電子メールテンプレートの編集)］* にあります。次のような名前が付けられています。

- *Password hint request*
- *Your password request*

これらのテンプレートは、必要に応じて編集できます。ただし、構造は変更しないでください。［パスワードを忘れた場合］ページでは、ユーザの言語に基づいてローカライズ版の電子メールテンプレートを表示するかどうかが判断されます。

「パスワードを忘れた場合」の環境設定

［パスワードを忘れた場合］ページの環境設定は、*［管理］* タブで行います。を参照してください。127 ページの 表 5-9

表 5-9 *「パスワードを忘れた場合」の環境設定*

環境設定	説明
ログインシーケンス	使用する NMAS ログインシーケンスです。このバージョンでは、本人確認の回答のみをサポートしています。
LDAP セキュアポート	使用するセキュア LDAP ポートです。デフォルトは 636 です。
ログインでワイルドカードを許可する	ユーザ名の指定時にワイルドカードの使用を許可する場合、True を選択します。(デフォルト値は False です)True を設定した場合、DN 情報の表示も True にする必要があります。 True を設定した場合、ユーザ名の先頭数文字を入力した後にワイルドカード文字を入力すれば、入力した文字列に一致する DN の一覧が［パスワードを忘れた場合］ページに表示されます。

環境設定	説明
DN 情報の表示	［パスワードを忘れた場合］ページに DN 値を表示する場合、True を選択します。このオプションは、**ログインでワイルドカードを許可する**と一緒に使用できます。False を設定した場合、DN コンテキスト情報は表示されません。
汎用パスワードポリシーユーザ DN	不正ユーザが有効なユーザ名を推測してシステムに不正アクセスすることを防止するために作成された、既存のアイデンティティボールトの DN を指定します。 デフォルトでは、ユーザが不正な名前を入力すると、「*ユーザが見つかりません*」というメッセージが表示されます。状況によっては不正ユーザが有効なユーザ名を推測できて、本人確認にも正しく回答できる可能性があります。これを防止する方法の 1 つとして、この値を指定できます。必要な環境設定作業については、128 ページの「汎用パスワードポリシーユーザ DN の設定」を参照してください。
エンコーディング	使用する文字のエンコードです。デフォルトは［utf-8］です。
パスワードリセットのヒントを表示	［パスワードのリセット］画面にユーザのパスワードのヒントを表示する場合は、*True* を選択します。 ［パスワードのリセット］画面にユーザのパスワードのヒントを表示しない場合は、*False* を選択します。

汎用パスワードポリシーユーザ DN の設定

汎用パスワードポリシーユーザ DN をサポートするには、ユーザコンテナにユーザを設定する必要があります。ユーザは次の条件を満たしていなければなりません。

- 推測しにくいパスワードを持っている。
- 電子メールアドレスをユーザアプリケーション管理者に割り当てている。

次の項目を設定する必要があります。

- 管理者が定義する、このユーザ用の本人確認の質問。
- 本人確認の質問に使用するパスワードポリシー。パスワードポリシーでは、ForgotPassword を有効にする必要があります。

このユーザとしてユーザアプリケーションにログインし、管理者が定義した質問に対する回答を最低 1 回は入力する必要があります。

最後に、ユーザアプリケーション管理者としてユーザアプリケーションにログインし、*［管理］*タブの *［パスワードを忘れた場合］*ページに移動します。ログインでワイルドカードを許可すると DN 情報の表示に False を指定します。新しく作成したユーザを、汎用パスワードポリシーユーザ DN として指定します。

5.3.4 ログインの環境設定

［ログイン］ページは、ユニバーサルパスワード、パスワードポリシー、および NMAS を通して、Identity Manager がサポートする堅牢なユーザ認証を実行します。［ログイン］ページでは、ログインプロセス中に必要な他のパスワードページにリダイレクトされます。

要件

［ログイン］ページの要件については、表 5-10 を参照してください。

表 5-10 *ログインの要件*

トピック	要件
パスワードポリシー	高度なパスワードルールを使用したり、ユーザに *［パスワードを忘れた場合］* リンクを使用させる場合を除き、このページにパスワードポリシーは不要です。
ユニバーサルパスワード	高度なパスワードルールと共にパスワードポリシーを使用するのでない限り、このページでユニバーサルパスワードを有効にする必要ありません。
SSL	このページでは SSL を使用するため、LDAP レルムへの SSL 接続をサポートするようにアプリケーションサーバが適切に設定されていることを確認します。

［パスワードモジュールのセットアップ］ ログインアクションを使って、次の設定を行います。

表 5-11 *ログインの環境設定*

環境設定	説明
ID のワイルドカードを許可する	True にした場合、ユーザ名の先頭数文字を入力すれば、その文字を含むユーザ名のリストが表示されます。ユーザは、そこからユーザ名を選択できます。
パスワードを忘れた場合のリンクを有効にする	True にした場合、［ログイン］ページに *［パスワードを忘れた場合］* リンクが表示されます。

環境設定	説明
パスワードを忘れた場合のリンク	この値は、［パスワードを忘れた場合］ページの名前とパスを定義します。この初期値は、インストール時に作成されます。外部パスワード管理 WAR を使用しない場合は、デフォルト値のままで構いません。 詳細については、46 ページのセクション 2.5「「パスワードを忘れた場合」の環境設定」を参照してください。
パスワードを忘れた場合の返信リンク	［パスワードを忘れた場合］リンクのように、この値はインストール時に設定されます。外部パスワード管理 WAR を使用しない場合は、変更する必要はありません。 外部パスワード管理 WAR を使用する場合、ユーザが［送信］をクリックした時に、*［パスワードを忘れた場合］*ページがユーザアプリケーションに戻るために使用する URL を指定します。戻るためのリンクは次の形式で指定します。 `protocol://servername:port/userappcontext` 例 :https://idmhost:8080/IDMProv 詳細については、46 ページのセクション 2.5「「パスワードを忘れた場合」の環境設定」を参照してください。
SSO を有効にする	True にした場合、ユーザ名とパスワードはセッションに保管され、他の適切に設定されたポートレットからアクセスすることができます。ユーザ名は SSO ユーザ ID キーに保管され、パスワードは SSO パスワードキーに保管されます。
SSO ユーザ ID キー	SSO を有効にするを True にした場合、このキーを使ってユーザ名がセッションに保存されます。
SSO パスワードキー	SSO を有効にするを True にした場合、このキーを使ってパスワードがセッションに保存されます。
ヒントの移行を有効にする	True にした場合、既存のヒントが nsimHint から nsimPasswordReminder に移動されます。
ロケールチェックを有効にする	ユーザが各自のロケールを設定していない場合、True にすると優先ロケールを設定できるページが表示されます。
パスワードオートコンプリートの有効化	ブラウザがサポートしている場合、True にすると、ユーザのブラウザにログイン資格情報を記憶しておくかどうかを尋ねるメッセージが表示されます。 False にした場合、ログイン資格情報を記憶しておくかどうかを尋ねるメッセージは表示されません。

［ログイン］ページの使用

［ログイン］ページを使用するには、次のことを理解しておく必要があります。

- 131 ページの 「［ログイン］ページから他のページへのリダイレクト方法」
- 131 ページの 「猶予ログインの使用」

［ログイン］ページから他のページへのリダイレクト方法

ランタイム時、［ログイン］ページでは、ログインプロセス中に必要となるものに応じて他のパスワードページにリダイレクトされます。詳細は、131 ページの 表 5-12 を参照してください。

表 5-12 *他のページへのリダイレクト*

ユーザの状況	ログインのリダイレクト先
［パスワードを忘れた場合］リンクをクリックする	［パスワードを忘れた場合］ページ
チャレンジ質問および回答を設定する必要がある	［本人確認の回答］ページ
パスワードヒントを設定する必要がある	［ヒントの設定］ページ
無効なパスワードをリセットする必要がある	［パスワードの変更］ページ

猶予ログインの使用

猶予ログインを使用すると、［ログイン］ページにパスワードの変更を要求する警告メッセージと猶予ログインの残り回数が表示されます。猶予ログインの残り回数がなくなると、［ログイン］ページから「パスワードの変更」ポートレットにリダイレクトされます。

5.3.5 パスワード同期ステータスの環境設定

ユーザはパスワード同期ステータスで、接続中のシステムのパスワード変更プロセスの進捗状況を確認できます。接続中の各システムを表す個別のイメージを指定できます。パスワード同期ステータスを設定する

- 同期化プロセス中にユーザにステータスを参照させる、接続したアプリケーションを定義します。134 ページの 表 5-14 を参考に、［パスワード同期ステータスのアプリケーション設定］で接続したアプリケーションを定義します。
- ユーザに表示する［パスワード同期ステータス］ページの設定を行います。詳細は、133 ページの ï 5-13§ 「パスワード同期ステータスクライアント設定」を参照してください。

デフォルトでは、ユーザアプリケーション管理者は、132 ページの 図 5-7 にある［パスワード同期ステータス］ページにアクセスして、他のユーザのパスワード同期ステータスを参照することができます。他のユーザの同期ステータスにアクセスするには、ユーザの DN を指定した後に、*［同期ステータスのチェック］*をクリックします。

図 5-7 *パスワード同期ステータス*

また、ユーザアプリケーション管理者に加えて、あなたは他のユーザの同期ステータスのチェックを行えるユーザセットを定義できます (トラブルシューティングや他の目的で)。PasswordManagement グループのメンバーには、他のユーザのパスワード同期ステータスを表示する権利が自動的に与えられます。デフォルトでは、このグループは存在していません。このグループを作成する場合、次の条件を満たしていなければなりません。

- グループ名は「PasswordManagement」.
- アイデンティティボールトへの権限を与えるこのグループには、パスワード同期ステータスを表示するユーザの eDirectory オブジェクト属性に対する読み込み権が必要です。

表 5-13 パスワード同期ステータスクライアント設定

環境設定	説明
パスワード同期バッファ時間 (ミリ秒)	パスワード同期ステータスチェックは、複数のアイデンティティボールトおよび接続したシステムのタイムスタンプを比較します。このバッファ時間は、これらの個別のマシンのシステム時間の差異を調整することを目的にしています。ここに指定した時間がユーザオブジェクトのパスワード変更属性のタイムスタンプに追加され、変更が行われたかどうかが判断されます。パスワード同期ステータスプロセスは、バッファ時間を次のように使用します。 ◆ 接続しているシステムの DirXML-PasswordSyncStatus にあるタイムスタンプ値 (パスワード同期時刻) が「前回パスワードが変更されたタイムスタンプ (ユーザオブジェクトの pwdChangedTime 属性) + パスワード同期バッファ時間」より古い場合、ステータスは古いと判断され、接続したシステムに対して更新ステータスのポーリングが続行されます。 ◆ 接続しているシステムの DirXML-PasswordSyncStatus にあるタイムスタンプ値が「前回パスワードが変更されたタイムスタンプ + パスワード同期バッファ時間」より新しい場合、パスワード同期機能からステータスコードまたはメッセージが返されて、接続したシステムの更新されたステータスが表示されます。 ◆ 前回パスワードが変更されたタイムスタンプは、ユーザのパスワードが変更された後にユーザオブジェクトに設定されます。この機能は、NMAS 3.1.3 以降で利用できます。
1 行のイメージ	Identity セルフサービスの［パスワード同期ステータス］ページで、1 行に表示するアプリケーションイメージ数を指定します。
個々のアプリケーションタイムアウト (ミリ秒)	パスワード同期ステータスプロセスが、接続した各アプリケーションのステータスを待機する時間。この時間が経過すると、次のステータスのチェックを開始します。
すべてのアプリケーションタイムアウト (ミリ秒)	パスワード同期ステータスプロセスがすべて完了するまでの許容時間です (すべての接続しているシステムに対して)。この時間が経過するまでの間、パスワード同期ステータスプロセスはポーリングを継続します。すべてのステータス値が更新されるか、タイムアウトが発生すると、プロセスは終了します。タイムアウトになった場合は、その旨を知らせるエラーメッセージが表示されます。
プロセス数	接続している各システムのパスワード同期ステータスが確認された回数です。
パスフレーズ	DirXML-PasswordSyncStatus にパスワードハッシュがある場合、このフィールドに入力した値がその値と比較されます。両方の値が等しくない場合、ハッシュが無効であることを知らせるメッセージが表示されます。
アプリケーションイメージサイズ制限 (バイト)	アップロードできるアプリケーションイメージサイズの最大値を設定できます (バイト)。このイメージは、表 5-14 で説明している［アプリケーションイメージ］で指定します。

パスワード同期ステータスのアプリケーション設定については、表 5-14 を参照してください。

表 5-14 *パスワード同期ステータスアプリケーション設定*

環境設定	説明
パスワード同期アプリケーション名	接続したアプリケーションを説明するために使用する名前です。アプリケーション名を複数のロケールで指定することができます。 言語 (ロケール) を追加する 1. *［言語の追加］* をクリックします。 2. 目的の言語に対応したフィールドにアプリケーション名を入力します。 3. ［*保存*］ をクリックします。 ローカライズ版のアプリケーション名を指定しない場合、*［パスワード同期アプリケーション名］* に設定された値が使用されます。
アプリケーション DirXML-PasswordSyncStatus GUID	ドライバ GUID を取得するには、ドライバオブジェクトの属性を参照します。次の 2 種類の方法があります。 ◆ このフィールドの隣にある［参照］ボタンをクリックする。この［参照］ボタンは、ユーザアプリケーションドライバがあるドライバセット内のドライブの GUID のみを取得します。 ◆ iManager を使ってドライバを参照し (オブジェクトの変更に使用する *［一般 - その他］* タブから)、GUID をコピーしてこのフィールドに貼り付ける。
アプリケーションイメージ	アップロードする接続したアプリケーションイメージの名前です。［パスワード同期ステータスのクライアント設定］セクションの［アプリケーションイメージサイズ制限］フィールドで、アプリケーションイメージサイズを設定することができます。.bmp、.jpeg、.jpg、.gif、および .png のファイル形式がサポートされています。
アプリケーションフィルタ	オプション。［パスワードの同期ステータスのチェック］ページでユーザがアプリケーション名を参照することを許可／禁止する LDAP フィルタを指定します。 任意の標準 LDAP フィルタを使用できます。

環境設定	説明
従属ドライバ	オプション。このアプリケーションが依存する他のドライバを指定します。 従属ドライバチェーン内に、ユーザが参照できないドライバがある場合は、アプリケーション DirXML-PasswordSyncStatus GUID に指定されたドライバもユーザには表示されません。 従属ドライバチェーン内のドライバがパスワード同期ステータスのチェックに失敗した場合、アプリケーション DirXML-PasswordSyncStatus GUID に指定されたドライバもパスワード同期ステータスのチェックに失敗します。 ドライバ GUID を取得するには、ドライバオブジェクトの属性を参照します。次の 2 種類の方法があります。 ◆ ［従属ドライバ］フィールドの隣にあるオブジェクトセレクタボタンを使用する。 この方法では、アプリケーションドライバの完全識別名 (FDN) が保存されます。ユーザがパスワード同期ステータスをチェックする時には、この FDN がユーザオブジェクトの DirXML-Associations 属性中の FDN フィールドの値と比較されます。2 つの FDN が一致しない場合、ユーザにこのアプリケーションは表示されません。一致した場合で、さらに DirXML-Associations 属性のドライバステータスフィールドの値が 0 でなく、またドライバデータフィールドがヌルでない場合、このアプリケーションはユーザに表示されます。 ◆ 従属ドライバの GUID を手動入力する。 このアプリケーションドライバが、ユーザアプリケーションドライバのある現在のドライバセットにない場合、この方法を使用します。この方法では、FDN は保存されません。ユーザがパスワード同期ステータスをチェックする時に、FDN は比較されません。また、アプリケーションフィルタで除外されているユーザ以外のユーザには、この従属ドライバが表示されます。

5.3.6 パスワードのヒントの変更の環境設定

このセルフサービスページでは、パスワードのヒントを設定、変更することができます。このヒントは、パスワードを忘れた場合に、そのヒントとして表示されるか、または電子メールで送信されます。

図 5-8 パスワードのヒントのサンプルの定義

要件

パスワードのヒントの変更に関する要件は、表 5-15 を参照してください。

表 5-15 パスワードのヒントの変更に関する要件

トピック	要件
ユニバーサルパスワード	ユニバーサルパスワードを有効にする必要はありません。

[パスワードのヒントの変更] ページの使用

[パスワードのヒント] ページを使用するには、次のことを理解しておく必要があります。

- 136 ページの 「ログイン時の [パスワードのヒント] ページの使用方法」
- 136 ページの 「ユーザアプリケーションでの [パスワードのヒントの変更] の使用」

ログイン時の [パスワードのヒント] ページの使用方法

ログインプロセス時に、ユーザがパスワードのヒントを設定する必要がある場合、[ログイン] ページから [パスワードのヒントの変更] ページに自動的にリダイレクトされます。たとえば、管理者が iManager でパスワードポリシーを割り当てたユーザが初めてログインする際には、パスワードポリシーはにより「パスワードを忘れた場合」が有効になり、アクションが *[ユーザにヒントを送信する]* または *[ヒントをページに表示]* に設定されます。

ユーザアプリケーションでの [パスワードのヒントの変更] の使用

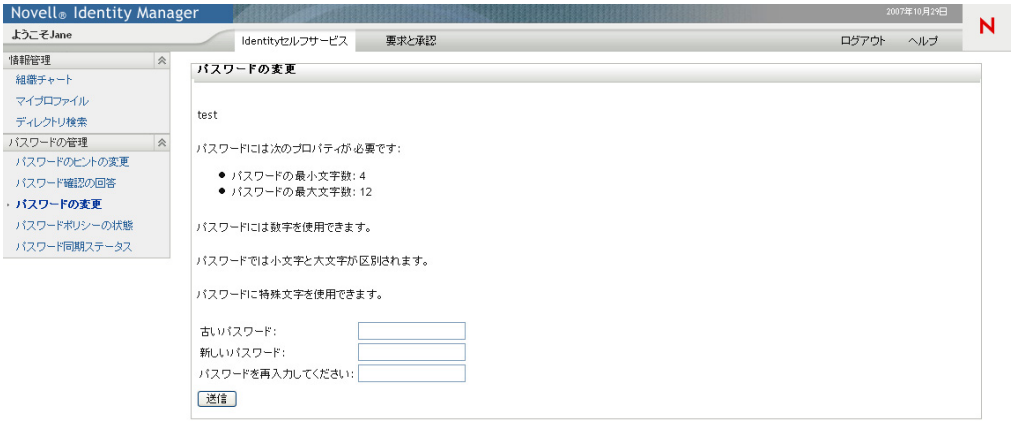
デフォルトでは、パスワードヒントを変更するためのセルフサービスをユーザアプリケーションで使用できます。

5.3.7 パスワードの変更の環境設定

このセルフサービスページにより、割り当てられたパスワードポリシーに従って、ユーザがユニバーサルパスワードを変更 (リセット) できます。新しいパスワードが準拠する必要のあるルールを \'95\'5c 示するには、そのポリシーを使用します。

ユニバーサルパスワードが有効になっていない場合、このページがユーザの eDirectory(シンプル) パスワードを変更します。このとき、ユーザのパスワード制限が適用されます。

図 5-9 パスワード変更



パスワードの変更に関する環境設定はありません。

要件

［パスワードの変更］ページの要件については、表 5-16 を参照してください。

表 5-16 パスワードの変更に関する要件

トピック	要件
ディレクトリ抽象化層の設定	. このページでは、ディレクトリ抽象化層の設定は必要ありません。
パスワードポリシー	ユニバーサルパスワードを有効にする高度なパスワードルールを使用するのでない限り、このページではパスワードポリシーは必要ありません。
ユニバーサルパスワード	このページをユニバーサルパスワードに使用する場合、ユーザに割り当てるパスワードポリシーの［高度なパスワードルール］で、*［ユーザにパスワード変更の開始を許可する］* 設定を有効にする必要があります。 このページを eDirectory (シンプル) パスワードに使用する場合、ユーザのパスワード制限で、*［Allow user to change password (ユーザにパスワードの変更を許可する)］* 設定を有効にする必要があります。

［パスワードの変更］ページの使用

［パスワードの変更］ページを使用するには、次のことを理解しておく必要があります。

- 137 ページの 「ログイン時の［パスワードの変更］ページの使用方法」
- 138 ページの 「ユーザアプリケーションでの［パスワードの変更］ページの使用」

ログイン時の［パスワードの変更］ページの使用方法

ログインプロセス時、ユーザが無効なパスワードをリセットする必要がある場合には、［ログイン］ページから［パスワードの変更］ページへと自動的にリダイレクトされます。

たとえば、管理者がユーザに対してパスワードのリセットを要求するパスワードポリシーを適用した場合、その後ユーザが初めてログインする際にはパスワードのリセットが要求されます。

ユーザに割り当てられパスワードポリシーに、パスワードを忘れた場合のアクションとしてパスワードのリセットが指定されている場合、［パスワードを忘れた場合］ページも自動的に［パスワードの変更］ページにリダイレクトされます。

ユーザアプリケーションでの［パスワードの変更］ページの使用

デフォルトでは、［パスワードの変更］ページによるパスワード変更のセルフサービスをユーザアプリケーションで使用できます。

ページの管理

6

この節では、Identity Manager ユーザインタフェースの *［管理］* タブの［ページ管理］ページを使用する方法について説明します。主なトピックは次のとおりです。

［管理］ タブにアクセスして操作する一般的な情報については、81 ページの第 4 章「［Administration］タブの使用」を参照してください。

6.1 ページの管理について

［ページ管理］ページを使用して、Identity Manager ユーザインタフェースに表示されるページを制御したり、それに対するアクセス権をユーザに割り当てたりすることができます。ユーザインタフェースは次の 2 種類のページで構成されます。

表 6-1 *ページタイプ*

ページの種類	説明
コンテナ	コンテナページは共有ページをラップし、外観や操作方法、企業のブランド、およびナビゲーションの方法の一貫性を保ちます。
共有	共有ページは特定の目的 (ユーザのプロファイルの更新など) に使用される、一貫したコンテンツのセットを提供します。これは複数のユーザが使用するサービスを提供することから共有ページと呼ばれます。

どちらのページタイプでも、ポートレット (プラグ可\'94\'5c ユーザインタフェースエレメントの Java 標準) の形式でコンテンツが含まれます。

ポートレットの詳細については、173 ページの第 7 章「ポートレットの管理」および 207 ページのパート IV「ポートレットリファレンス」を参照してください。

6.1.1 コンテナページについて

このセクションでは、Identity Manager ユーザインタフェースで重要な役割を果たすいくつかのコンテナページについて説明します。

これらのコンテナページは、必要に応じて変更できます。また、独自のコンテナページを追加することもできます。

コンテナページの操作の詳細については、147 ページのセクション 6.2「コンテナページの作成とメンテナンス」を参照してください。

GuestContainerPage

デフォルトでは、ユーザが Identity Manager ユーザインタフェースにアクセスすると、図 6-1 に示すような GuestContainerPage と呼ばれるコンテナページがログインの前に表示されます。

図 6-1 デフォルトのゲストコンテナページ

GuestContainerPage の内部は、次のようなレイアウトがなされています。

図 6-2 GuestContainerPage のレイアウト

GuestContainerPage のレイアウトは 3 つの領域に分けられており、それぞれ次のポートレットが \'95\'5c 示されます。

表 6-2 レイアウト領域

ポートレット	説明
HeaderPortlet	ユーザインタフェースのヘッダ情報およびトップレベルのタブ制御が \'95\'5c 示されます。
Shared Page Navigation	垂直のメニューが \'95\'5c 示され、ユーザはこのメニューから共有ページを選択して \'95\'5c 示できます。
Portal Page Controller	ユーザが共有ページナビゲーションポートレットで現在選択している共有ページが \'95\'5c 示されます。

デフォルトでは、ユーザがログインする前は、これらのポートレットに次のコンテンツのみ表示されます。

- ヘッダに 1 つのリンク : *ログイン*
- 1 つの共有ページ : *ようこそ*

ユーザがまだログインしていないため、共有ページナビゲーションポートレットは、［ゲストページ］カテゴリにある共有ページのみを表示し、他のカテゴリはすべて除外します。デフォルトでは［Welcome］ページだけが［Guest Pages］カテゴリに含まれています。

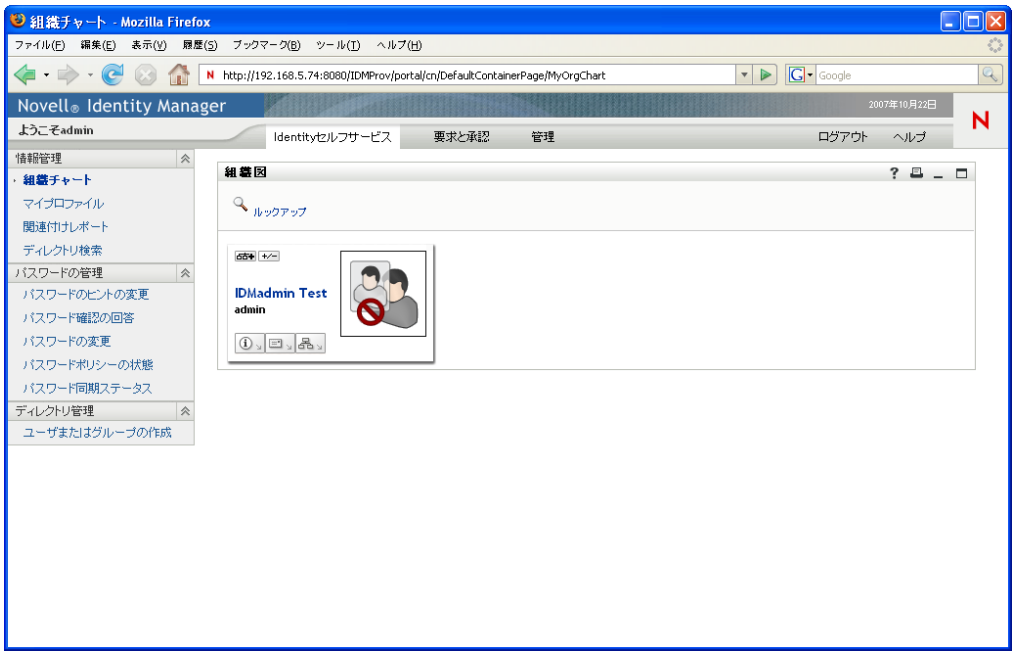
ログイン後、共有ページナビゲーションポートレットは［ゲストページ］カテゴリを除外します。代わって共有ページの他のカテゴリが \'95\'5c 示されます (初期設定に従います)。

共有ページナビゲーションポートレットの詳細については、209 ページの第 10 章「ポートレットについて」を参照してください。

DefaultContainerPage

デフォルトでは、ユーザが Identity Manager ユーザインタフェースにログインすると、図 6-3 に示すような DefaultContainerPage と呼ばれるコンテナページが表示されます。

図 6-3 *Default Container Page*



DefaultContainerPage のレイアウトを図 6-4 に示します。

図 6-4 デフォルトのコンテナページレイアウト

DefaultContainerPage のレイアウトは 3 つの領域に分けられており、表 6-3 に示すようなポートレットがそれぞれに表示されます。

表 6-3 デフォルトのコンテナページポートレット

ポートレット	説明
HeaderPortlet	ユーザインタフェースのヘッダ情報およびトップレベルのタブ制御が \'95\'5c 示されます。
Shared Page Navigation	垂直のメニューが \'95\'5c 示され、ユーザはこのメニューから共有ページを選択して \'95\'5c 示できます。
Portal Page Controller	ユーザが共有ページナビゲーションポートレットで現在選択している共有ページが \'95\'5c 示されます。
Session Timeout Warning	ユーザセッションのタイムアウトが近づくと警告メッセージを \'95\'5c 示します。

ユーザがログインすると、DefaultContainerPage の HeaderPortlet に *［Identity セルフサービス］* タブが自動的に表示されます。

管理コンテナページ

デフォルトでは、ユーザアプリケーション管理者 (および許可された他のユーザ) が Identity Manager ユーザインタフェースの *［管理］* タブをクリックすると、管理コンテナ

ページと呼ばれるコンテナページが表示されます。このページは、図 6-5 のように表示されます。

図 6-5 *デフォルトの管理コンテナページ*

管理コンテナページのレイアウトを図 6-6 に示します。

図 6-6 管理コンテナページのレイアウト

管理コンテナページのレイアウトは 2 つの領域に分けられており、表 6-4 に示すようなポートレットがそれぞれに表示されます。

表 6-4 デフォルトの管理コンテナページポートレット

ポートレット	説明
HeaderPortlet	ユーザインタフェースのヘッダ情報およびトップレベルのタブ制御が \'95\'5c 示されます。
Admin List Display	2 番目のレベルのタブが \'95\'5c 示され、ユーザはこの中から管理アクションを選択して実行できます。
Portal Page Controller	管理リスト \'95\'5c 示ポートレットでユーザが現在選択しているタブに対応する共有ページが \'95\'5c 示されます。
Session Timeout Warning	ユーザセッションのタイムアウトが近づくと警告メッセージを \'95\'5c 示します。

6.1.2 共有ページについて

Identity Manager ユーザインタフェースには、コンテナページに主なコンテンツを提供する数多くの共有ページが含まれています。これらの共有ページは、必要に応じて変更できます。また、独自の共有ページを追加することもできます。

共有ページの操作の詳細については、154 ページのセクション 6.3「共有ページの作成とメンテナンス」を参照してください。

一般的な共有ページ

これらの共有ページの一例として、ユーザが Identity Manager にログインすると DefaultContainerPage にデフォルトの共有ページの［組織チャート］ページが表示されます。このツールを「図 6-7」に示します。

図 6-7 サンプルの共有ページ

組織チャートのレイアウトを図 6-8 に示します。

図 6-8 デフォルトの組織チャートのレイアウト

組織チャートのレイアウトは 1 つの領域だけで \'8d\'5c 成されており、ポートレットは 1 つだけ \'95\'5c 示されます (組織チャートポートレット)。

6.1.3 ページの使用に関する例外

この節では、Identity Manager ユーザインタフェースの最上位レベルのタブが、次に示す各ページを基にしてどのように構成されているかを説明しました。

- *［Identity セルフサービス］*タブは DefaultContainerPage を使用します。
- *［管理］*タブは管理コンテナページを使用します。

ただし、*［要求と承認］*タブは別のアーキテクチャを基にしているため、［ページ管理］からは操作できません。

6.2 コンテナページの作成とメンテナンス

コンテナページの作成とメンテナンスには、次の手順が含まれます。

1 に従って、新しいコンテナページを作成するか、既存のコンテナページを 147 ページのセクション 6.2.1「コンテナページの作成」選択します。

2 に従って、ページに (ポートレット形式で) コンテンツを追加します。149 ページのセクション 6.2.2「コンテナページへのコンテンツの追加」

ページからコンテンツを削除することもできます (151 ページのセクション 6.2.3「コンテナページからのコンテンツの削除」を参照)。

3 に従って、ポータルレイアウトを選択します。152 ページのセクション 6.2.4「コンテナページのレイアウトの変更」

4 に従って、選択したレイアウトのコンテンツの順序と位置を決めます。152 ページのセクション 6.2.5「コンテナページへのコンテンツの配置」

5 コンテナページの URL をブラウザに指定して、新しいページをすぐに表示します (154 ページのセクション 6.2.6「コンテナページの表示」を参照)。

コンテナページのレイアウトを切り替えてもページのコンテンツは失われません。コンテナページに新しいレイアウトを適用すると、ページ内のポートレットは自動的に新しいレイアウトで表示されます。新しいレイアウトではコンテンツの位置調整が必要なこともあります。

6.2.1 コンテナページの作成

コンテナページは初めから作成することも、既存のページをコピーして作成することもできます。このセクションでは、両方の手順を説明します。

コンテナページを初めから作成する

1 ［Page Admin］ページで［*Maintain Container Pages*］*を選択します。*

［Maintain Container Pages］パネルが \'95\'5c 示されます。

2 *[新規]* ページアクションを選択します (パネルの左下にあります)。
タイトルとカテゴリが未設定のコンテナページが作成されます。

3 コンテナページの［page properties］を指定します。

プロパティ	操作

Page Link Name (URI)	ページの URI 名を指定します (ユーザインタフェースの URL 内に表示されます)。たとえば、次のように URI を指定すると：
	`MyContainerPage`
	URL では次のように \'95\'5c 示されます。
	`http://myappserver:8080/IDM/portal/cn/MyContainerPage`

Page Name	ページの表示名を指定します。例： `My Container Page` この名前を他の言語にローカライズする場合は、*[ローカライズ]* をクリックします。

プロパティ	操作
Navigation Priority	次のいずれかを指定します。 ◆ このコンテナページに優先度を割り当てる必要がない場合 *[なし]* を選択します。 ◆ このコンテナページに、他のコンテナページに対する優先度を割り当てる場合、*[値の設定]* を選択します。優先度は 0 ～ 9999 の間の整数を指定します。0 は優先度が最高で、9999 は優先度が最低です。 優先度順にページがリストされるときに特定の順序で \'95\'5c 示したい場合や、(ユーザが複数のグループに属しているため) デフォルトページが複数存在するときに特定のページを選択しておきたい場合、優先値を設定しておくと便利です。
Default Shared Page	詳細については、169 ページのセクション 6.6「コンテナページのデフォルト共有ページの選択」を参照してください。
Assign Categories	ページが所属するカテゴリを次から 0 個以上選択します。 ◆ 管理 ◆ 一般 カテゴリ順にページがリストされるときに適切に整理されるようにしたい場合や、ページがカテゴリ順にフィルタされるときに適切なサブセットが選択されるようにしたい場合は、カテゴリを割り当てておくと便利です。
説明	ページを説明するテキストを入力します。

4 ［*Save Page*］をクリックします (［Page Properties］セクションの下部)。

既存のページをコピーしてコンテナページを作成する

1 ［Page Admin］ページで［*Maintain Container Pages*］*を選択します。*

［Maintain Container Pages］パネルが \'95\'5c 示されます (前の手順と同じ)。

2 コンテナページのリストで、コピーするページを選択します。

リストが長い場合は、リストを (カテゴリ順や開始テキスト順に) 並べ替えると、目的のページを見つけやすくなります。

3 *[コピー]* のページアクションを選択します (パネルの左下にあります)。

新しいコンテナページが作成され、［Copy of OriginalPageName］という名前が付けられます。

4 コンテナページの［page properties］を指定します (前の手順と同じ)。

5 ［*Save Page*］*をクリックします (［Page Properties］セクションの下部)。*

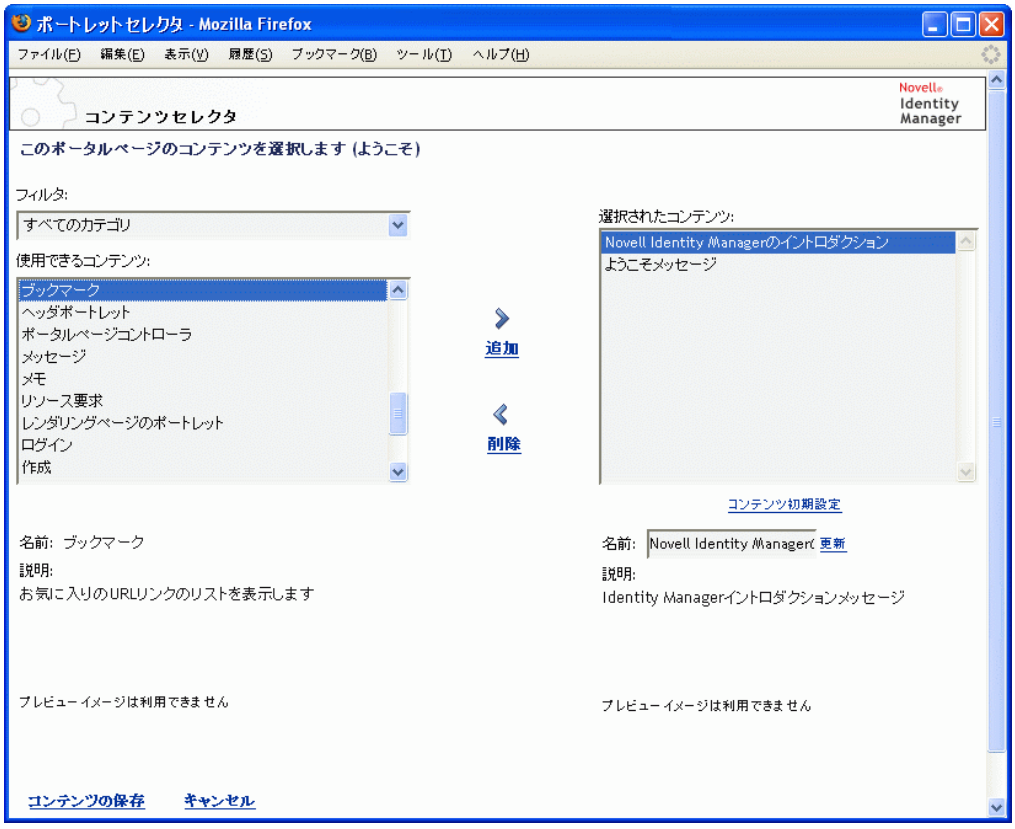
6.2.2 コンテナページへのコンテンツの追加

コンテナページを作成したら、次の手順として、ページに設定するポートレットを選択してコンテンツを追加します。Identity Manager ユーザアプリケーションに付属の作成済みポートレットを使用することも、登録した他のポートレットを使用することもできます。

コンテナページにコンテンツを追加する

1 ［Maintain Container Pages］パネルで新規または既存のページを開き、［*Select Content*］*ページタスクをクリックします(パネルの下部)*。

新しいブラウザウィンドウに［Content Selector］が \'95\'5c 示されます。



2 使用可能なコンテンツの中から特定のカテゴリのコンテンツを表示する場合は、*［フィルタ］*ドロップダウンリストからカテゴリを選択します。

3 *［使用できるコンテンツ］*リストからポートレットを 1 つまたは複数選択します。

リストから非連続のポートレットを複数選択する場合は Ctrl キーを押しながら選択します。連続した複数のポートレットを選択する場合は Shift キーを押しながら選択します。

4 *［追加］*をクリックして、選択したポートレットを *［選択されたコンテンツ］*リストに移動します。

5 *［コンテンツ初期設定］*をクリックすると、コンテナページのために選択したポートレットの初期設定を編集できます。指定した初期設定値は、ページに \'95\'5c 示されるポートレットのインスタンスに反映されます。

6 ［*Save Contents*］*をクリックします。*

これでコンテナページのコンテンツを選択しました。続いて 152 ページのセクション 6.2.4「コンテナページのレイアウトの変更」の説明に従って新しいレイアウトを選択するか、152 ページのセクション 6.2.5「コンテナページへのコンテンツの配置」の説明に従って現在のレイアウトのコンテンツを配置できます。

6.2.3 コンテナページからのコンテンツの削除

コンテナページの作成中、あるページからポートレットを削除してコンテンツを削除することも可能です。このような場合、次の手順に従って［Content Selector］または［Layout Selector］を使用します。

［Content Selector］を使ってコンテナページからコンテンツを削除する

1 ［Maintain Container Pages］パネルでページを開き、［*Select Content*］ページタスクをクリックします*(* パネルの下部*)*。

新しいブラウザウィンドウに、150 ページのステップ 1 に示すような［コンテンツセレクタ］が表示されます。

2 *［選択されたコンテンツ］*リストから削除するポートレットを選択し、*［削除］*をクリックします。

ポートレットがページから削除されます。

3 ［*Save Contents*］をクリックします。

［Layout Selector］を使ってコンテナページからコンテンツを削除する

1 ［Maintain Container Pages］パネルでページを開き、［*Arrange Content*］ページタスクをクリックします*(* パネルの下部*)*。

新しいブラウザウィンドウに［Layout Selector］が \'95\'5c 示され、そのページのポートレットが \'95\'5c 示されます。

2 削除するポートレットの *X\'83\'7b* タンをクリックします。

3 確認のメッセージが \'95\'5c 示されたら、［*OK*］をクリックします。

ポートレットがページから削除されます。

4 ［*Save Layout*］をクリックします。

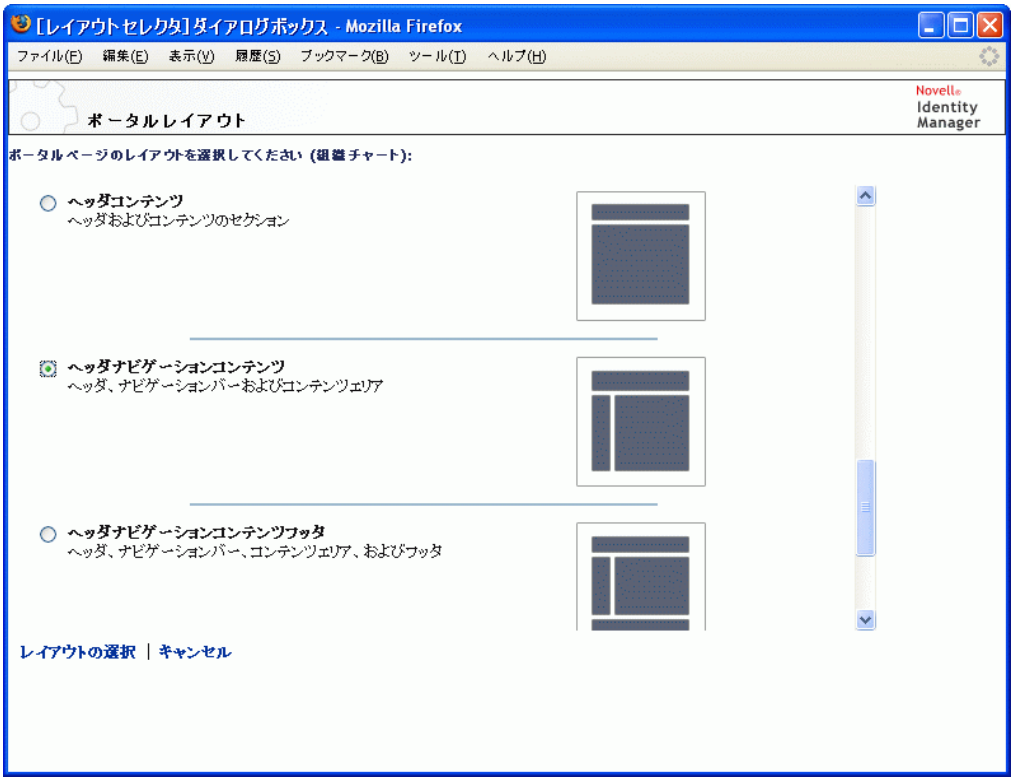
6.2.4 コンテナページのレイアウトの変更

コンテナページのレイアウトを変更すると、新しいレイアウトに合わせて既存のコンテンツが移動します。場合によっては最終結果を調整する必要があります。

コンテナページのレイアウトを変更する

1 ［Maintain Container Pages］パネルでページを開き、［*Select Layout*］ページタスクをクリックします(パネルの下部)。

新しいブラウザウィンドウに［Portal Layouts］リストが \'95\'5c 示されます。



2 選択項目をスクロールし、使用するレイアウトを選択します。

3 ［*Select Layout*］をクリックします。

6.2.5 コンテナページへのコンテンツの配置

コンテナページのコンテンツやレイアウトを指定した後、選択したレイアウトにコンテンツを配置できます。また、特定の場所に他のポートレットを追加したり、ポートレットを削除したりできます。

1 ［Maintain Container Pages］パネルでページを開き、［*Arrange Content*］ページタスクをクリックします(パネルの下部)。

新しいブラウザウィンドウに［Layout Selector］が \'95\'5c 示され、そのページのポートレットが \'95\'5c 示されます。

2 ページにポートレットを追加する

2a 目的のレイアウトフレームにある［*Add Content*］をクリックします。

新しいブラウザウィンドウに［Portlet Selector］が \'95\'5c 示されます。

2b 使用可能なコンテンツの中から特定のカテゴリのコンテンツを表示する場合は、*［フィルタ］*ドロップダウンリストからカテゴリを選択します。

2c *［使用できるコンテンツ］*リストから追加するポートレットを選択します。

2d ［*Select Content*］をクリックします。

［Portlet Selector］が閉じ、選択したポートレットが［Layout Selector］にある目的のレイアウトフレームに \'95\'5c 示されます。

3 レイアウト内の別の場所にポートレットを移動する場合は、次のブラウザ別の手順に従ってください。

ブラウザ	操作
Internet Explorer	1. ポートレットのタイトルバーにカー \'83\'5c ルを移動し、カー \'83\'5c ルが手の形になるようにします。 2. \'83\'7d ウスの左 \'83\'7b タンを押し、レイアウト内の希望の場所にポートレットをドラッグします。

ブラウザ	操作
Mozilla	1. 移動するポートレットをクリックします。 2. 移動先のレイアウトフレームの内側をクリックします。 ポートレットが目的の場所に移動します。

4 レイアウトからポートレットを削除する場合は、次の手順に従います。

4a 削除するポートレットの *X\'83\'7b タンをクリックします。*

4b 確認のメッセージが \'95\'5c 示されたら、［*OK*］*をクリックします。*

ポートレットがレイアウトから削除されます。

5 ポートレットの初期設定を編集する

5a 編集するポートレットの鉛筆型の \'83\'7b タンをクリックします。

ポートレットの［*Content Preferences*］*がブラウザに \'95\'5c 示されます。*

5b 必要に応じて初期設定値を変更します。

指定した初期設定値は、ページに \'95\'5c 示されるポートレットのインスタンスに反映されます。

5c ［*Save Preferences*］*をクリックします。*

6 ［*Save Layout*］をクリックして変更を保存し、［Layout Selector］を閉じます。

6.2.6 コンテナページの表示

コンテナページを表示するには、ブラウザでコンテナページの URL に移動します。Web ブラウザで次の URL を指定します。

`http://server:port/IDM-war-context/portal/cn/container-page-name`

たとえば、MyContainerPage というコンテナページを \'95\'5c 示するには、次のページに移動します。

`http://myappserver:8080/IDM/portal/cn/MyContainerPage`

6.3 共有ページの作成とメンテナンス

共有ページの作成とメンテナンスには、次の手順が含まれます。

1 に従って、新しい共有ページを作成するか、既存の共有ページを 155 ページのセクション 6.3.1「共有ページの作成」選択します。

2 に従って、ページに (ポートレット形式で) コンテンツを追加します。157 ページのセクション 6.3.2「共有ページへのコンテンツの追加」

ページからコンテンツを削除することもできます (159 ページのセクション 6.3.3「共有ページからのコンテンツの削除」を参照)。

3 に従って、ポータルレイアウトを選択します。160 ページのセクション 6.3.4「共有ページのレイアウトの変更」

4 に従って、選択したレイアウトのコンテンツの順序と位置を決めます。160 ページのセクション 6.3.5「共有ページへのコンテンツの配置」

5 共有ページの URL をブラウザに入力して、新しいページを表示します (162 ページのセクション 6.3.6「共有ページの表示」を参照)。

共有ページとレイアウト

共有ページはポータルレイアウトに完全にバインドされてるわけではありません。このため、共有ページのレイアウトを切り替えてもページのコンテンツは失われません。新しいレイアウトが適用されると、ページに追加されたポートレットは自動的に新しいレイアウトで表示されます。新しいレイアウトではコンテンツの位置調整が必要なこともあります。

6.3.1 共有ページの作成

共有ページは初めから作成することも、既存のページをコピーして作成することもできます。このセクションでは、両方の手順を説明します。

共有ページを初めから作成する

1 ［Page Admin］ページで［*Maintain Shared Pages*］*を選択します。*

［Maintain Shared Pages］パネルが \'95\'5c 示されます。

2 *［新規］*ページアクションを選択します (パネルの左下にあります)。

タイトルとカテゴリが未設定の共有ページが作成されます。

3 共有ページの［page properties］を指定します。

プロパティ	操作

Page Link Name (URI)	ページの URI 名を指定します (ユーザインタフェースの URL 内に表示されます)。たとえば、次のように URI を指定すると :
	`MySharedPage`
	URL では次のように \'95\'5c 示されます。
	`http://myappserver:8080/IDM/portal/cn/MyContainerPage/MySharedPage`

Page Name	ページの表示名を指定します。例 : `My Shared Page` この名前を他の言語にローカライズする場合は、［*Localize*］*をクリックします。*
Navigation Priority	次のいずれかを指定します。 ◆ この共有ページに優先度を割り当てる必要がない場合 *［なし］* を選択します。 ◆ この共有ページに、他の共有ページに対する優先度を割り当てる場合、*［値の設定］* を選択します。優先度は 0 ～ 9999 の間の整数を指定します。0 は優先度が最高で、9999 は優先度が最低です。 優先度順にページがリストされるときに特定の順序で \'95\'5c 示したい場合や、(ユーザが複数のグループに属しているため) デフォルトページが複数存在するときに特定のページを選択しておきたい場合、優先値を設定しておくと便利です。
Parent Page	この共有ページを他の共有ページの子として設定する場合は、*［親の選択］* **をクリックします。親ページと子ページが両方とも** *同じカテゴリに属していることを確認してください (\'95\'5c 示の問題を避けるため)*。 エンドユーザがランタイム時に共有ページナビゲーションポートレットを使用すると、この関係が表示されます。共有ページのリストを \'95\'5c 示すると、親ページの下に子ページがインデント \'95\'5c 示されます。 子ページは親ページのコンテンツ、初期設定、および設定を継承しません。逆に言えば、親ページがそれ自体のコンテンツと同時に子ページのコンテンツを自動的に表示することはありません。

プロパティ	操作
Assign Categories	ページが所属するカテゴリを次から 0 個以上選択します。 ◆ 管理 ◆ ディレクトリ管理 ◆ 一般 ◆ Guest Pages ◆ 情報管理 ◆ パスワード管理 カテゴリ順にページがリストされるときに適切に整理されるようにしたい場合や、ページがカテゴリ順にフィルタされるときに適切なサブセットが選択されるようにしたい場合は、カテゴリを割り当てておくと便利です。 **注 :**「ゲストページ」は特別なカテゴリで、ユーザのログイン前に表示される (ログイン後ではない) 共有ページの識別に使用されます。詳細については、209 ページの第 10 章「ポートレットについて」にある共有ページナビゲーションポートレットのセクションを参照してください。
説明	ページを説明するテキストを入力します。

4 ［*Save Page*］をクリックします (［Page Properties］セクションの下部)。

既存のページをコピーして共有ページを作成する

1 ［Page Admin］ページで［*Maintain Shared Pages*］*を選択します。*

155 ページの 「共有ページを初めから作成する」に示すような、［共有ページの保守］パネルが表示されます。

2 共有ページのリストで、コピーするページを選択します。

リストが長い場合は、リストを (カテゴリ順や開始テキスト順に) 並べ替えると、目的のページを見つけやすくなります。

3 ［*Copy*］*のページアクションを選択します (パネルの左下)。*

新しい共有ページが作成され、［Copy of OriginalPageName］という名前が付けられます。

4 共有ページのページプロパティを指定します (155 ページの 「共有ページを初めから作成する」 を参照)。

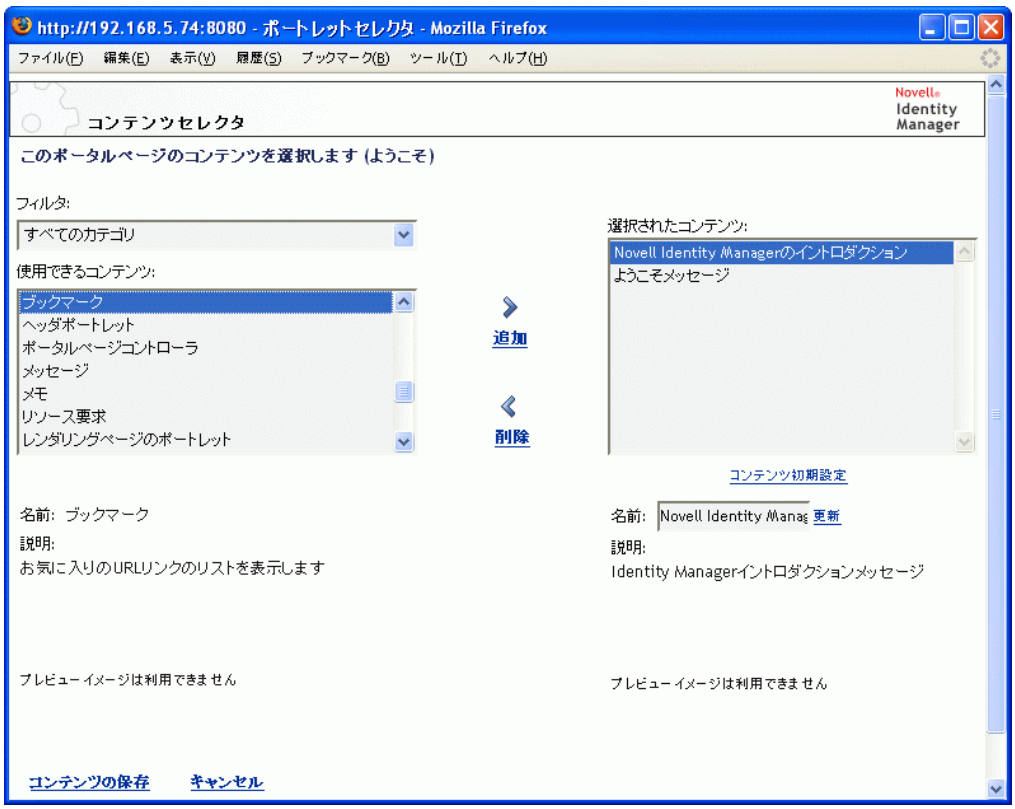
5 ［*Save Page*］*をクリックします (［Page Properties］セクションの下部)。*

6.3.2 共有ページへのコンテンツの追加

共有ページを作成したら、次の手順として、ページに設定するポートレットを選択してコンテンツを追加します。Identity Manager ユーザアプリケーションに付属の作成済みポートレットを使用することも、登録した他のポートレットを使用することもできます。

1 ［Maintain Shared Pages］パネルで新規または既存のページを開き、［*Select Content*］*ページタスクをクリックします (パネルの下部)。*

新しいブラウザウィンドウに［Content Selector］が \'95\'5c 示されます。



2 使用可能なコンテンツの中から特定のカテゴリのコンテンツを表示する場合は、*［フィルタ］*ドロップダウンリストからカテゴリを選択します。

3 *［使用できるコンテンツ］*リストからポートレットを 1 つまたは複数選択します。

リストから非連続のポートレットを複数選択する場合は Ctrl キーを押しながら選択します。連続したポートレットを複数選択する場合は Shift キーを押しながら選択します。

4 *［追加］*をクリックして、選択したポートレットを *［選択されたコンテンツ］*リストに移動します。

5 ［*コンテンツ初期設定*］をクリックすると、共有ページのために選択したポートレットの初期設定を編集できます。指定した初期設定値は、ページに \'95\'5c 示されるポートレットのインスタンスに反映されます。

6 ［*Save Contents*］*をクリックします。*

これで共有ページのコンテンツを選択しました。続いて 160 ページのセクション 6.3.4「共有ページのレイアウトの変更」の説明に従って新しいレイアウトを選択するか、160 ページのセクション 6.3.5「共有ページへのコンテンツの配置」の説明に従って現在のレイアウトのコンテンツを配置できます。

6.3.3 共有ページからのコンテンツの削除

共有ページの作成中、あるページからポートレットを削除してコンテンツを削除することも可能です。このような場合、次の手順に従って［Content Selector］または［Layout Selector］を使用します。

1 ［Maintain Shared Pages］パネルでページを開き、［*Select Content*］*ページタスクをクリックします(パネルの下部)*。

新しいブラウザウィンドウに、157 ページのセクション 6.3.2「共有ページへのコンテンツの追加」に示すような［コンテンツセレクタ］が表示されます。

2 *［選択されたコンテンツ］*リストから削除するポートレットを選択し、*［削除］*をクリックします。

ポートレットがページから削除されます。

3 ［*Save Contents*］*をクリックします。*

［レイアウトセレクタ］を使って共有ページからコンテンツを削除する

1 ［Maintain Shared Pages］パネルでページを開き、［*Arrange Content*］*ページタスクをクリックします(パネルの下部)*。

新しいブラウザウィンドウに［Layout Selector］が \'95\'5c 示され、そのページのポートレットが \'95\'5c 示されます。

2 削除するポートレットの *X\'83\'7b タンをクリックします。*

3 確認のメッセージが \'95\'5c 示されたら、［*OK*］*をクリックします。*

ポートレットがページから削除されます。

4 ［*Save Layout*］をクリックします。

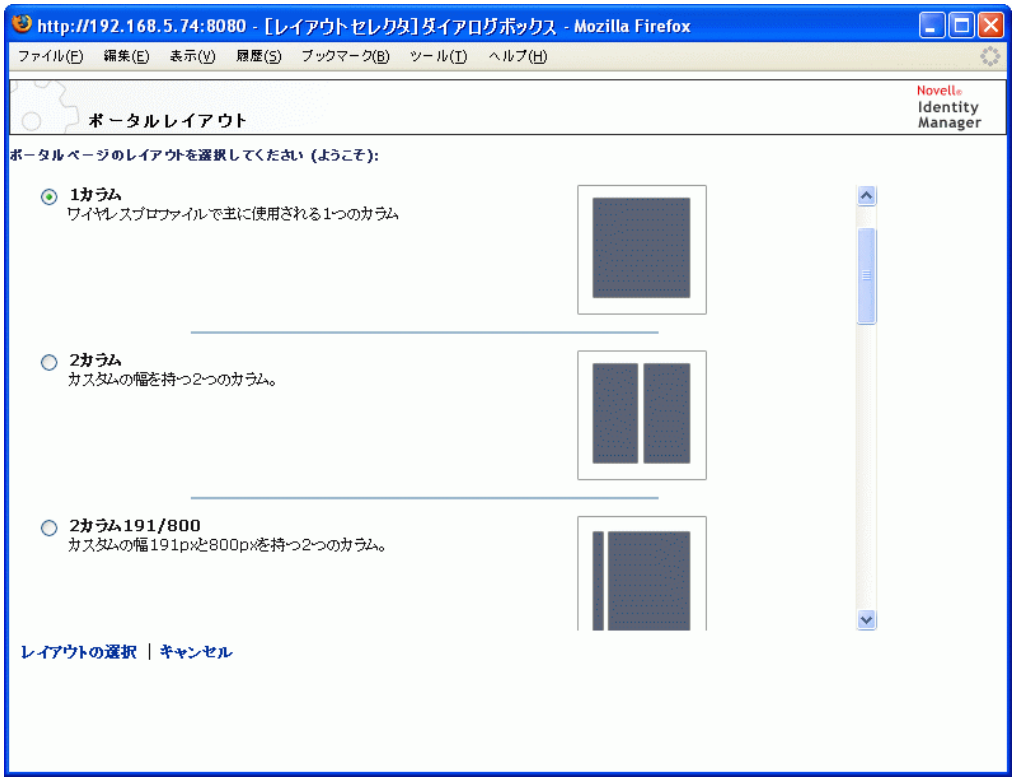
6.3.4 共有ページのレイアウトの変更

共有ページのレイアウトを変更すると、新しいレイアウトに合わせて既存のコンテンツが移動します。場合によっては最終結果を調整する必要があります。

共有ページのレイアウトを変更する

1 ［Maintain Shared Pages］パネルでページを開き、［*Select Layout*］ページタスクをクリックします*(* パネルの下部*)*。

新しいブラウザウィンドウに［Portal Layouts］リストが \'95\'5c 示されます。



2 選択項目をスクロールし、使用するレイアウトを選択します。

3 ［*Select Layout*］をクリックします。

6.3.5 共有ページへのコンテンツの配置

共有ページのコンテンツやレイアウトを指定した後、選択したレイアウトにコンテンツを配置できます。また、特定の場所に他のポートレットを追加したり、ポートレットを削除したりできます。

共有ページでコンテンツを配置する

1 ［Maintain Shared Pages］パネルでページを開き、［*Arrange Content*］ページタスクをクリックします*(* パネルの下部*)*。

新しいブラウザウィンドウに［Layout Selector］が \'95\'5c 示され、そのページのポートレットが \'95\'5c 示されます。

2 ページにポートレットを追加する場合：

2a 目的のレイアウトフレームにある［*Add Content*］をクリックします。

新しいブラウザウィンドウに［Portlet Selector］が \'95\'5c 示されます。

2b 使用可能なコンテンツの中から特定のカテゴリのコンテンツを表示する場合は、*［フィルタ］*ドロップダウンリストからカテゴリを選択します。

2c *［使用できるコンテンツ］*リストから追加するポートレットを選択します。

2d ［*Select Content*］をクリックします。

［Portlet Selector］が閉じ、選択したポートレットが［Layout Selector］にある目的のレイアウトフレームに \'95\'5c 示されます。

3 レイアウト内の別の場所にポートレットを移動する場合は、次のブラウザ別の手順に従ってください。

ブラウザ	操作
Internet Explorer	1. ポートレットのタイトルバーにカー \'83\'5c ルを移動し、カー \'83\'5c ルが手の形になるようにします。 2. \'83\'7d ウスの左 \'83\'7b タンを押し、レイアウト内の希望の場所にポートレットをドラッグします。

ブラウザ	操作
Mozilla Firefox	1. 移動するポートレットをクリックします。 2. 移動先のレイアウトフレームの内側をクリックします。 ポートレットが目的の場所に移動します。

4 レイアウトからポートレットを削除する場合 :

4a 削除するポートレットの *X\'83\'7b タンをクリックします。*

4b 確認のメッセージが \'95\'5c 示されたら、［*OK*］*をクリックします。*

ポートレットがレイアウトから削除されます。

5 ポートレットの初期設定を編集する場合 :

5a 編集するポートレットの鉛筆型の \'83\'7b タンをクリックします。

ポートレットの［Content Preferences］がブラウザに \'95\'5c 示されます。

5b 必要に応じて初期設定値を変更します。

指定した初期設定値は、ページに \'95\'5c 示されるポートレットのインスタンスに反映されます。

5c ［*Save Preferences*］*をクリックします。*

6 ［*Save Layout*］をクリックして変更を保存し、［Layout Selector］を閉じます。

6.3.6 共有ページの表示

共有ページを表示するには、Web ブラウザで次のリンクに移動します。

http://server:port/IDM-war-context/portal/pg/shared-page-name

たとえば、MyContainerPage という共有ページを \'95\'5c 示するには、次のページに移動します。

http://myappserver:8080/IDM/portal/pg/MySharedPage

6.4 ページの許可の割り当て

他のユーザ、グループ、およびコンテナが特定のコンテナページや共有ページを操作できるように、許可を割り当てることができます。次のように 2 種類のセキュリティレベルの許可を割り当てることができます。

表 6-5 *ページ許可*

許可	説明	割り当てる対象
表示	ユーザ、グループ、またはコンテナにページへのアクセスを許可します。また、使用可 \'94\'5c なページのリストにそのページが \'95\'5c 示されます。	コンテナページおよび共有ページ

許可	説明	割り当てる対象
所有権	ユーザ、グループ、またはコンテナにページのコンテンツやレイアウトの変更を許可します。また、そのユーザ、グループ、またはコンテナが他のユーザ、グループ、およびコンテナに \'95\'5c 示および所有権の許可を割り当てることを許可します。	共有ページ

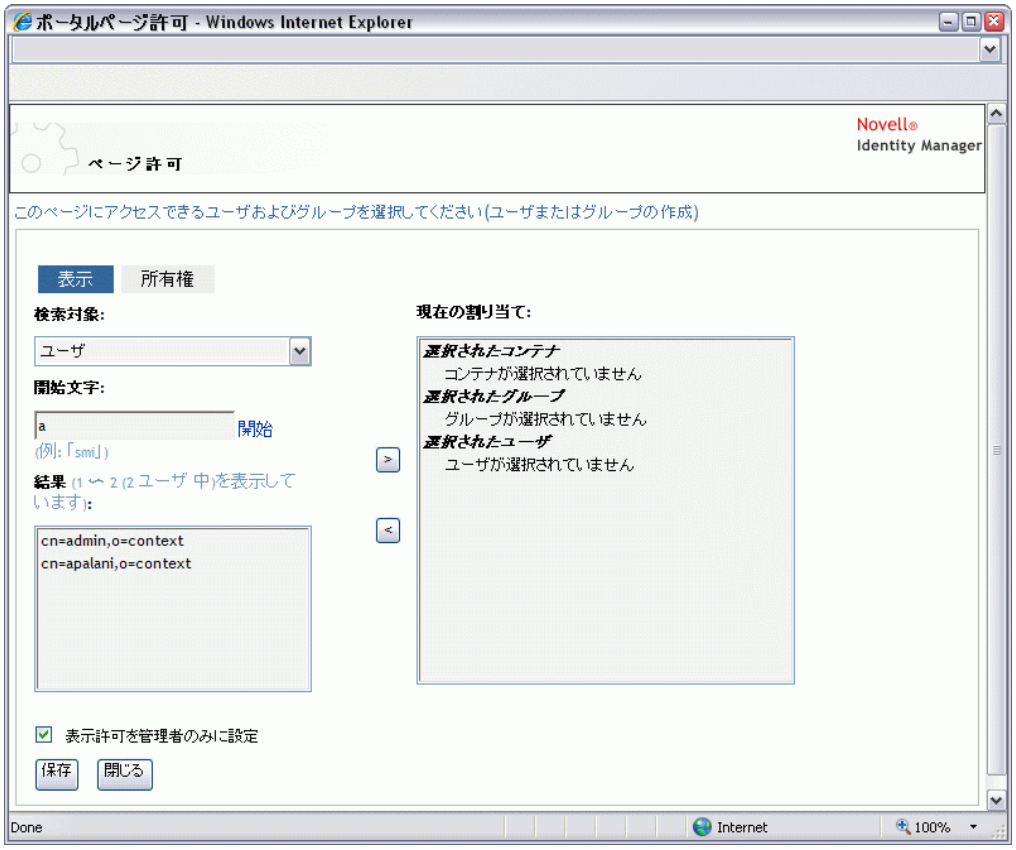
6.4.1 ページ表示許可の割り当て

ユーザにコンテナページや共有ページの \'95\'5c 示許可を割り当てると、ユーザはそのページにアクセスできます。また、使用可 \'94\'5c なページのリストにそのページが \'95\'5c 示されます。

コンテナページや共有ページの \'95\'5c 示許可を割り当てる

1 ［Maintain Container Pages］パネルまたは［Maintain Shared Pages］パネルでページを開き、［*Assign Permissions*］*ページタスクをクリックします(パネルの下部)*。

新しいブラウザウィンドウに［ページ許可］ダイアログボックスが表示されます。



2 ［*View*］*タブに移動します。*

3 次の検索設定の値を指定します。

設定	操作
検索対象	次のいずれかをドロップダウンメニューから選択します。 ◆ ユーザ ◆ グループ ◆ コンテナ
Starts with	必要な作業 : ◆ 指定したタイプ (ユーザ、グループ、またはコンテナ) で使用できるオブジェクトすべてを検索する場合は、この設定を空白にします。 ◆ これらのオブジェクトのサブセットを検索して、目的の CN 値の開始文字を入力します。(大文字小文字は区別されません。ワイルドカードはサポートされていません)。 たとえば、S から始まるグループを検索する場合、検索結果は次のようになります : cn=Sales,ou=groups,o=MyOrg cn=Service,ou=groups,o=MyOrg cn=Shipping,ou=groups,o=MyOrg Se から始まるグループを検索した場合は、次のようになります : cn=Service,ou=groups,o=MyOrg

4 *［Go］をクリックします。*

検索結果は、*［Results］リストに \'95\'5c 示されます。*

5 ページを割り当てるユーザ、グループ、またはコンテナを選択して、*［Add(>)］\'83\'7b タンをクリックします。*

複数選択するには、<Ctrl> キーを押しながら選択します。

6 次のようにページロックの有効または無効を設定します。

必要な作業	操作手順
ページをロックし、ユーザアプリケーション管理者だけが \'95\'5c 示できるようにする	*［表示許可を管理者のみに設定］* を選択します。
割り当てられたすべてのユーザ、グループ、およびコンテナがページを \'95\'5c 示できるようにする	*［表示許可を管理者のみに設定］* の選択を解除します。 **注 :** この設定の選択を解除した状態でユーザ、グループ、またはコンテナがページに対して明示的に割り当てられていない場合、全員にこのページへの表示許可が割り当てられることになります。

7 *［保存］、［閉じる］* の順にクリックします。

6.4.2 共有ページ所有者の割り当て

共有ページを所有するユーザは、所有しているページのコンテンツを変更できます。また、それらのページのポートレットの初期設定を変更できます。

共有ページに所有権の許可を割り当てる

1 ［Maintain Shared Pages］パネルでページを開き、［*Assign Permissions*］ページタスクをクリックします (パネルの下部)。

新しいブラウザウィンドウに、163 ページのステップ 1 に示すような［ページ許可］ダイアログボックスが表示されます。

2 ［*Ownership*］*タブに移動します。*

3 次の検索設定の値を指定します。

設定	操作
検索対象	次のいずれかをドロップダウンメニューから選択します。 ◆ ユーザ ◆ グループ ◆ コンテナ
Starts with	必要な作業 : ◆ 指定したタイプ (ユーザ、グループ、またはコンテナ) で使用できるオブジェクトすべてを検索する場合は、この設定を空白にします。 ◆ これらのオブジェクトのサブセットを検索して、目的の CN 値の開始文字を入力します。(大文字小文字は区別されません。ワイルドカードはサポートされていません)。 たとえば、S から始まるグループを検索する場合、検索結果は次のようになります : cn=Sales,ou=groups,o=MyOrg cn=Service,ou=groups,o=MyOrg cn=Shipping,ou=groups,o=MyOrg Se から始まるグループを検索した場合は、次のようになります : cn=Service,ou=groups,o=MyOrg

4 ［*Go*］*をクリックします。*

検索結果は、［*Results*］*リストに \'95\'5c 示されます。*

5 ページを割り当てるユーザ、グループ、またはコンテナを選択して、［*Add(>)*］*\'83\'7b タンをクリックします。*

複数選択するには、<Ctrl> キーを押しながら選択します。

6 次のようにページロックの有効または無効を設定します。

必要な作業	操作手順
ページをロックし、ユーザアプリケーション管理者だけが操作できるようにする	*［所有権許可を管理者のみに設定］*を選択します。
割り当てられたすべてのユーザ、グループ、およびコンテナがページを操作できるようにする	*［所有権許可を管理者のみに設定］*の選択を解除します。 **注 :** この設定の選択を解除した状態でユーザ、グループ、またはコンテナがページに対して明示的に割り当てられていない場合、全員にこのページへの所有権許可が割り当てられることになります。

7 *［保存］、［閉じる］*の順にクリックします。

6.4.3 ［ユーザまたはグループの作成］ページへのユーザアクセスを有効にする

デフォルトでは、ユーザアプリケーション管理者だけが［ユーザまたはグループの作成］ページを表示＜使用できます。このページは、Identity Manager ユーザインタフェースの *［Identity セルフサービス］*タブの共有ページです。ただし、ユーザアプリケーション管理者は状況に応じて、このページにアクセスするための許可を 1 人または複数のエンドユーザに割り当てることができます。たとえば、管理職にある、あるユーザが、ユーザ、グループ、またはタスクグループを作成する機能が必要なことがあります。

［Create User or Group］ページへのアクセスをユーザに許可する

1 ［Maintain Shared Pages］パネルで、［Create User or Group］という名前のページを開きます。

2 *［Assign Permissions］ページタスクを使用して、適切なユーザ、グループ、またはコンテナに、［Create User or Group］共有ページの*\'95\'5c 示許可を与えます。

3 ［ページ管理］から［ポートレット管理］に切り替え、CreatePortlet というポートレット登録を開きます (これは［ユーザまたはグループの作成］ページで使用します)。

4 ［Security］パネルを使用して、適切なユーザ、グループ、またはコンテナに、［CreatePortlet］ポートレット登録のリストおよび実行許可を与えます。

ポートレットに許可を割り当てる際の詳細については、173 ページの第 7 章「ポートレットの管理」を参照してください。

5 iManager に移動し、管理者アカウントを使用してアイデンティティボールトのツリーにログインします。

6 ［ユーザまたはグループの作成］を使用するユーザが、オブジェクト (ユーザ、グループ、およびタスクグループ) が作成されるコンテナの「Entry Rights」プロパティを作成する権利を持っていることを確認します。

たとえば、選択したコンテナのトラスティを変更し、適切なユーザ、グループ、またはコンテナをトラスティとして追加できます。それから各トラスティに対し次の権利を割り当てます。

プロパティ名	割り当てる権利	継承
[All Attributes Rights]	• 比較 • 読み込み • 書き込み	継承する (このチェック \'83\'7bックスをオンにします)
[Entry Rights]	• 参照 • 作成	継承する (このチェック \'83\'7bックスをオンにします)

必要な権利をアイデンティティボールトで割り当てなかった場合 (または何らかの理由でこうした権利が生成されなかった場合)、［ユーザまたはグループの作成］によってエンドユーザに対し次のようなエラーメッセージが表示されます。

```
User 'cn=mmackenzie,ou=users,ou=idmsample,o=novell' does not have
permission
to create 'cn=MyNewGroup,ou=groups,ou=idmsample,o=novell' or
```

```
modify related
objects.
```

(ページへのアクセス権利を持つユーザの) ［Create User or Group］ページの使用方法については、『*Identity Manager User Application: ユーザガイド*』を参照してください。

6.4.4 個々の［管理］ページへのユーザアクセスを有効にする

デフォルトでは、ユーザアプリケーション管理者だけが Identity Manager ユーザインタフェースの *［管理］*タブ、およびそのタブに含まれるページ (［アプリケーション環境設定］、［ページ管理］、［ポートレット管理］、［プロビジョニング］、［セキュリティ］) にアクセスできます。ただし必要であれば、ユーザアプリケーション管理者は 1 人または複数のエンドユーザに、*［管理］*タブの特定のページを表示および使用する許可を割り当てることができます。たとえば、ユーザアプリケーション管理者ではないユーザが、定期的にテーマを変更する必要がある場合があります。

個別の［Administration］ページへのユーザアクセスを有効にする

1 ［Maintain Container Pages］パネルで［Admin Container Page］を開きます。

これは、Identity Manager ユーザインタフェースの *［管理］*タブに移動したときに使用されるコンテナページです。

2 *［Assign Permissions］ページタスクを使用して、適切なユーザ、グループ、またはコンテナに、［Admin Container Page］の* \'95\'5c 示許可を与えます。

3 ［Maintain Shared Pages］パネルで、適切な［Administration］ページ (［Administration］カテゴリ内にある共有ページの 1 つ) を開きます。

4 *［Assign Permissions］ページタスクを使用して、適切なユーザ、グループ、またはコンテナに、*その共有ページの \'95\'5c 示および所有権の許可を与えます。

5 指定したユーザ、グループ、またはコンテナに、指定したページで使用される各ポートレットの実行許可があることを確認します (これらのポートレットを制限した場合)。

ポートレットに許可を割り当てる際の詳細については、173 ページの第 7 章「ポートレットの管理」を参照してください。

6.5 グループのデフォルトページの設定

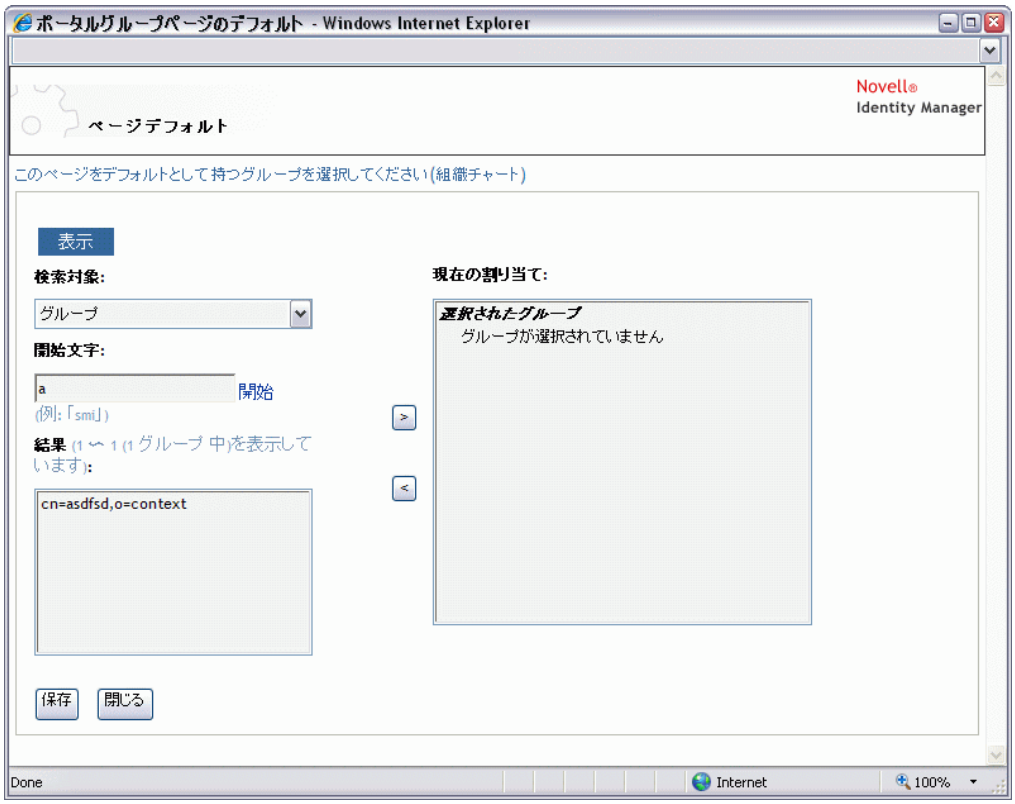
許可を受けたユーザのグループに対し、デフォルトのコンテナページおよびデフォルトの共有ページを割り当てることができます。ユーザがログインしたときに表示されるコンテナページ、およびコンテナページ上に表示される共有ページは、これらの設定によって決まります。

ユーザが複数のグループに属しており、デフォルトのページが複数割り当てられている場合、\'95\'5c 示されるコンテナページと共有ページは［Navigation Priority］を使用して決定されます。

グループにデフォルトのコンテナページやデフォルトの共有ページを割り当てる

1 ［Maintain Container Pages］パネルまたは［Maintain Shared Pages］パネルでページを開き、*［Set As Default］ページタスクをクリックします (パネルの下部)。*

ブラウザの新しいウィンドウに［ページデフォルト］のダイアログボックスが表示されます。



2 次の検索設定の値を指定します。

設定	操作
検索対象	［グループ］が自動的に選択されます。
Starts with	必要な作業 : ◆ 使用できるグループすべてを検索する場合は、この設定を空白にします。 ◆ これらのグループのサブセットを検索して、目的の CN 値の開始文字を入力します。(大文字小文字は区別されません。ワイルドカードはサポートされていません)。 たとえば、S から始まるグループを検索する場合、検索結果は次のようになります : cn=Sales,ou=groups,o=MyOrg cn=Service,ou=groups,o=MyOrg cn=Shipping,ou=groups,o=MyOrg Se から始まるグループを検索した場合は、次のようになります : cn=Service,ou=groups,o=MyOrg

3 *［Go］をクリックします。*

検索結果は、*［Results］リストに\'95\'5c 示されます。*

4 *このページをデフォルトとして設定するグループを選択して、［Add(>)］\'83\'7b タンをクリックします。*

複数選択するには、<Ctrl> キーを押しながら選択します。

5 *［保存］、［閉じる］*の順にクリックします。

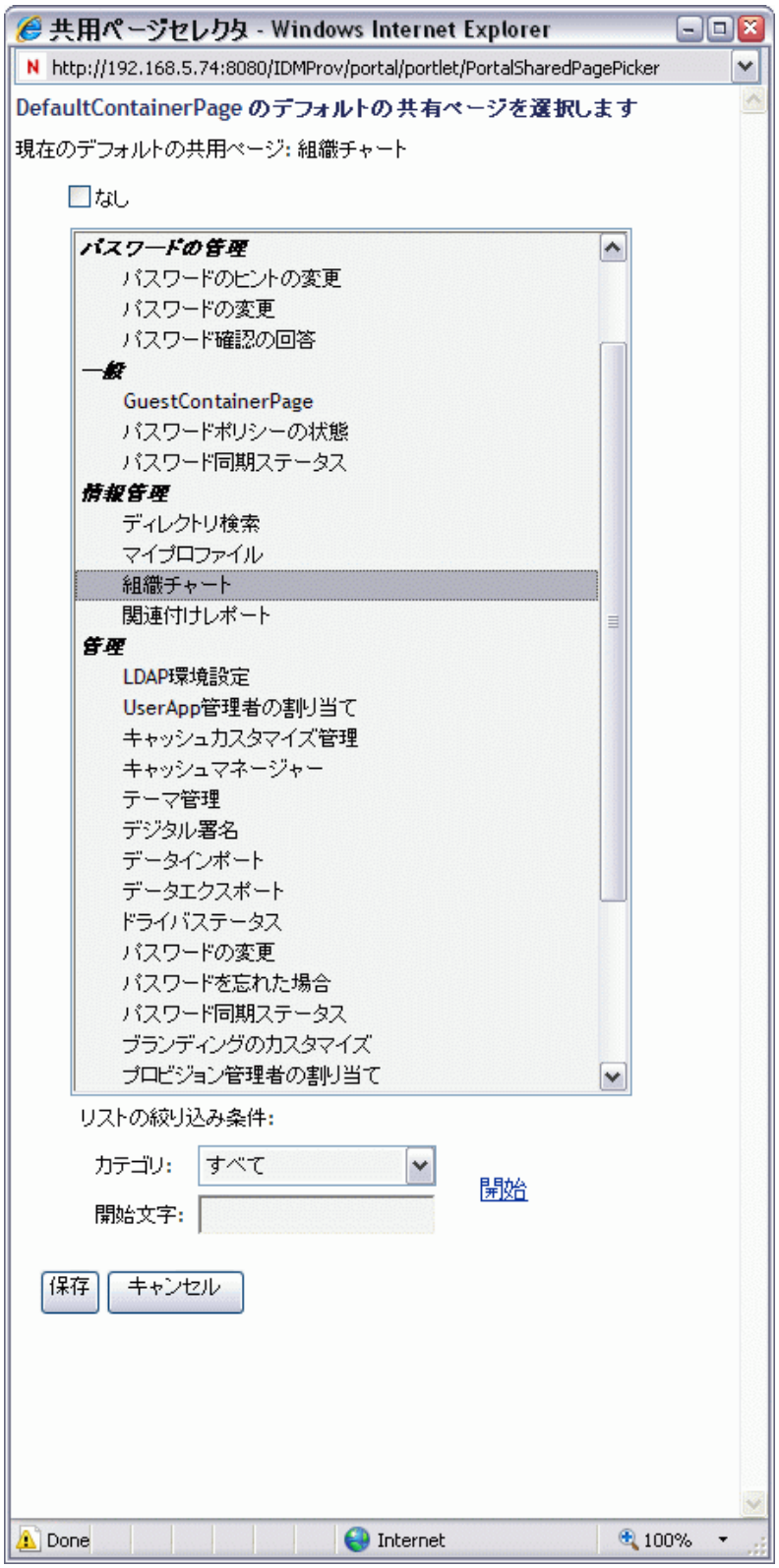
6.6 コンテナページのデフォルト共有ページの選択

使用する各コンテナページに対してデフォルトの共有ページを割り当てられます。ユーザインタフェースは、\'95\'5c 示内容を決定するときにこのページ割り当てを参照します。

1 ［Maintain Container Pages］パネルでコンテナページを開きます。

2 ［Page Properties］セクションで［Default Shared Page］を見つけ、［Select Default］をクリックします。

ブラウザの新しいウィンドウにデフォルトの共有ページを選択するためのダイアログボックスが表示されます。



3 共有ページのリストが長い場合は、リストをカテゴリ順や開始テキスト順に並べ替えると、目的のページを見つけやすくなります。

4 コンテナページのデフォルトとして使用する共有ページを選択します。デフォルトを設定しない場合は*［なし］*を選択します。

5 ［*Save*］*をクリックして選択を適用し、ダイアログ\'83\'7bックスを閉じます。*

6 ［*Save Page*］*をクリックします(［Page Properties］セクションの下部)。*

ポートレットの管理

7

この節では、Identity Manager ユーザインタフェースの *[管理]* タブの［ポートレット管理］ページを使用する方法について説明します。主なトピックは次のとおりです。

- 173 ページのセクション 7.1「ポートレットの管理について」
- 174 ページのセクション 7.2「ポートレット定義の管理」
- 178 ページのセクション 7.3「登録されたポートレットの管理」

[管理] タブにアクセスして操作する一般的な情報については、81 ページの第 4 章「［Administration］タブの使用」を参照してください。

7.1 ポートレットの管理について

［ポートレット管理］ページを使用すると、Identity Manager のユーザインタフェースで使用できるポートレット、およびそれらのポートレットへのアクセス許可を持つユーザを制御できます。ポートレットは、プラグ可能なユーザインタフェースエレメント (Java 標準に基づく) で、ユーザインタフェース内のページのコンテンツ (コンテナページや共有ページなど) を提供します。ポートレットの管理方法については、表 7-1 を参照してください。

表 7-1 *ポートレットの管理*

操作する対象	説明
ポートレット定義	ポートレット設定パラメータを指定する記述子です (portlet.xml から読み込まれます)。アプリケーション内の各ポートレットに対し 1 つの定義があります。 詳細については、**174 ページのセクション 7.2「ポートレット定義の管理」**を参照してください。
ポートレット登録	ポートレット定義に基づくポートレットの登録です。1 つのポートレットアプリケーションで、同じポートレットの複数の登録が存在できます。 詳細については、**178 ページのセクション 7.3「登録されたポートレットの管理」**を参照してください。

Identity Manager ユーザインタフェースに付属するポートレットの詳細については、207 ページのパート IV「ポートレットリファレンス」を参照してください。コンテナページや共有ページでポートレットを使用する場合の詳細については、139 ページの第 6 章「ページの管理」を参照してください。

7.2 ポートレット定義の管理

［Portlet Admin］ページでは、ポートレットアプリケーションのポートレット定義に関係した次のタスクを実行できます。

- 174 ページのセクション 7.2.1「展開されたポートレットアプリケーションのポートレット定義へのアクセス」
- 174 ページのセクション 7.2.2「ポートレット定義の登録」
- 176 ページのセクション 7.2.3「ポートレット定義の情報の表示」

7.2.1 展開されたポートレットアプリケーションのポートレット定義へのアクセス

[ポートレットアプリケーション] リストには、選択したポートレットアプリケーションのポートレット定義が表示されます。

[ポートレットアプリケーション] リストで、アクセスするポートレット定義のポートレットアプリケーションを展開します。

そのポートレットアプリケーションに含まれるポートレット定義がツリー \'95\'5c 示されます。

7.2.2 ポートレット定義の登録

ポートレットを使用する前に、ポータル (Identity Manager ユーザアプリケーション) にポートレット定義を登録する必要があります。登録されたポートレット定義は「*ポートレット登録*」と呼ばれます。1 つのポートレットに対し複数の登録を作成できるため、同じページにそのポートレットのインスタンスを複数設定できます。

ポートレット登録はポートレットクラスの初期設定と設定をすべて継承しますが、これらの値は次の方法で変更できます。

- ポートレット定義を登録する場合、178 ページのセクション 7.3「登録されたポートレットの管理」を参照してください。
- ポートレットのインスタンスをページに追加する場合、139 ページの第 6 章「ページの管理」を参照してください。

Identity Manager ユーザアプリケーションに付属するポートレットはすべて、自動的に登録されます。

ポートレット定義が編集モードを提供する場合、エンドユーザはランタイムにポートレット登録の特定の初期設定を変更できます。この場合、ポートレットの doEdit() メ \'83\'5cッドのロジックに従います。

Identity Manager ユーザアプリケーションでは、デフォルトの編集モードも実装しています。doEdit() メ \'83\'5cッドが明示的に実装されていない場合は、デフォルトの初期設定シートが \'95\'5c 示されます。

ポートレット定義を登録する

1 ［Portlet Applications］リストで、ポートレット登録を作成するポートレット定義を展開します。

［General］パネルが右側に \'95\'5c 示されます。

選択したポートレットの既存の登録が、［ポートレットアプリケーション］ツリー (左側) の対応するポートレット定義名の下にリストされます。

2 *［新規ポートレットインスタンスの登録］*テキストボックスでポートレット登録の固有の名前を指定し、*［登録］*をクリックします。

新しいポートレット登録が作成され、［Portlet Applications］ツリーにリストされます。

3 新しいポートレット登録の初期設定や設定を変更する場合は、178 ページのセクション 7.3「登録されたポートレットの管理」を参照してください。

7.2.3 ポートレット定義の情報の表示

リストされたポートレット定義に関する次の情報を \'95\'5c 示できます (読み込み専用)。

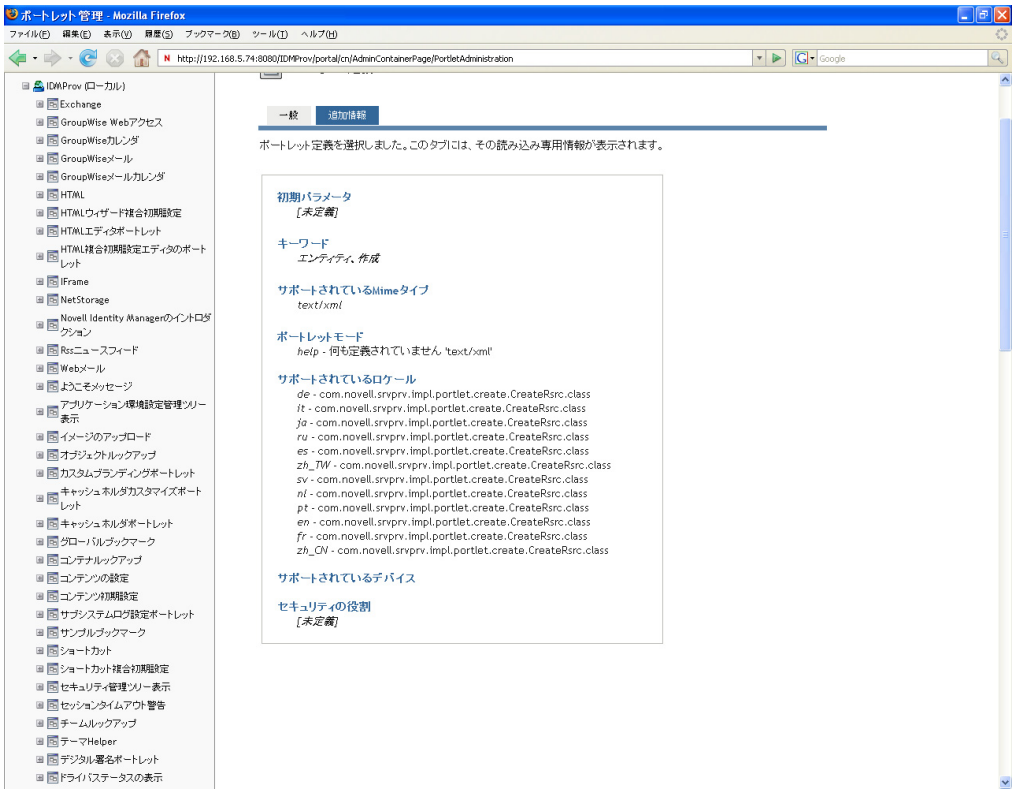
- \'95\'5c 示名
- クラス名
- ポートレットのタイトル
- 実行のタイプ (同期または非同期)
- 短いタイトル
- 登録のタイプ
- スタイル名
- キャッシュの有効期限
- 説明
- 初期化パラメータ
- キーワード
- サポートされている MIME タイプ
- ポートレットによってサポートされているモード
- サポートされているロケール
- サポートされているデバイス
- セキュリティの役割

ポートレット定義の情報を \'95\'5c 示する

1 ［Portlet Applications］リストで、情報を \'95\'5c 示するポートレット定義を選択します。

［General］パネルが右側に \'95\'5c 示され、選択したポートレット定義の情報が \'95\'5c 示されます。

2 ［Additional Information］パネルに移動し、選択したポートレットの詳細を \'95\'5c 示します。



7.3 登録されたポートレットの管理

［Portlet Admin］ページでは、ポートレットアプリケーションのポートレット登録に関係した次のタスクを実行できます。

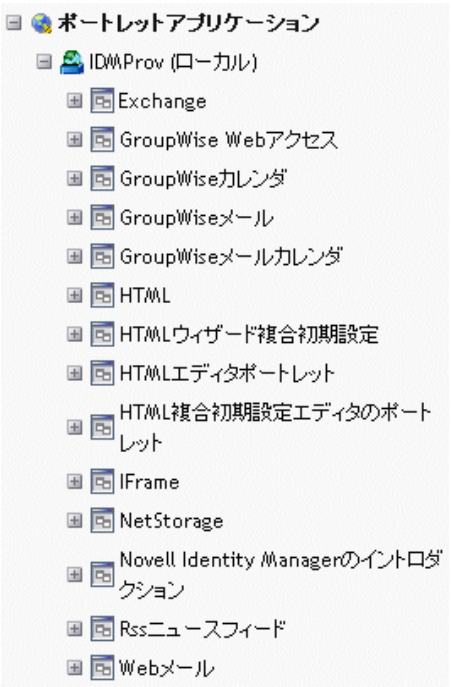
7.3.1 展開されたポートレットアプリケーションでのポートレット登録へのアクセス

［Portlet Applications］リストには、選択したポートレットアプリケーションにある各ポートレット定義のポートレット登録が \'95\'5c 示されます。

展開されたポートレットアプリケーションでポートレット登録にアクセスする

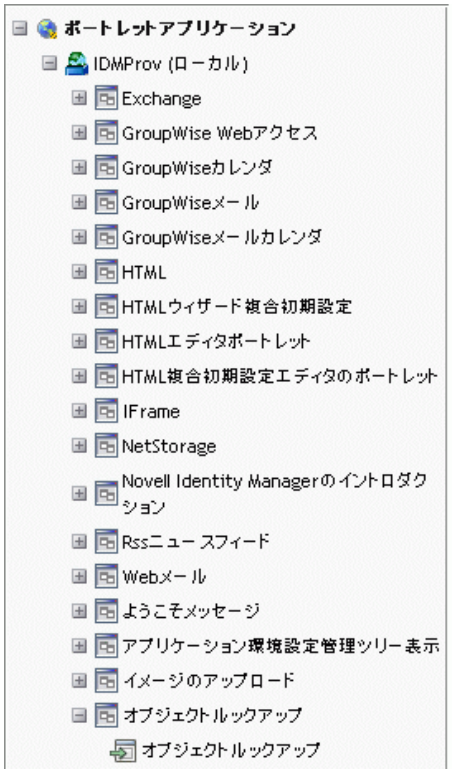
1 ［Portlet Applications］リストで、アクセスしたいポートレット定義と登録が含まれたポートレットアプリケーションを展開します。

そのポートレットアプリケーションに含まれるポートレット定義がツリー \'95\'5c 示されます。



2 アクセスしたいポートレット登録が含まれたポートレット定義を展開します。

そのポートレット定義に含まれるポートレット登録がツリー \'95\'5c 示されます。



7.3.2 ポートレット登録の情報の表示

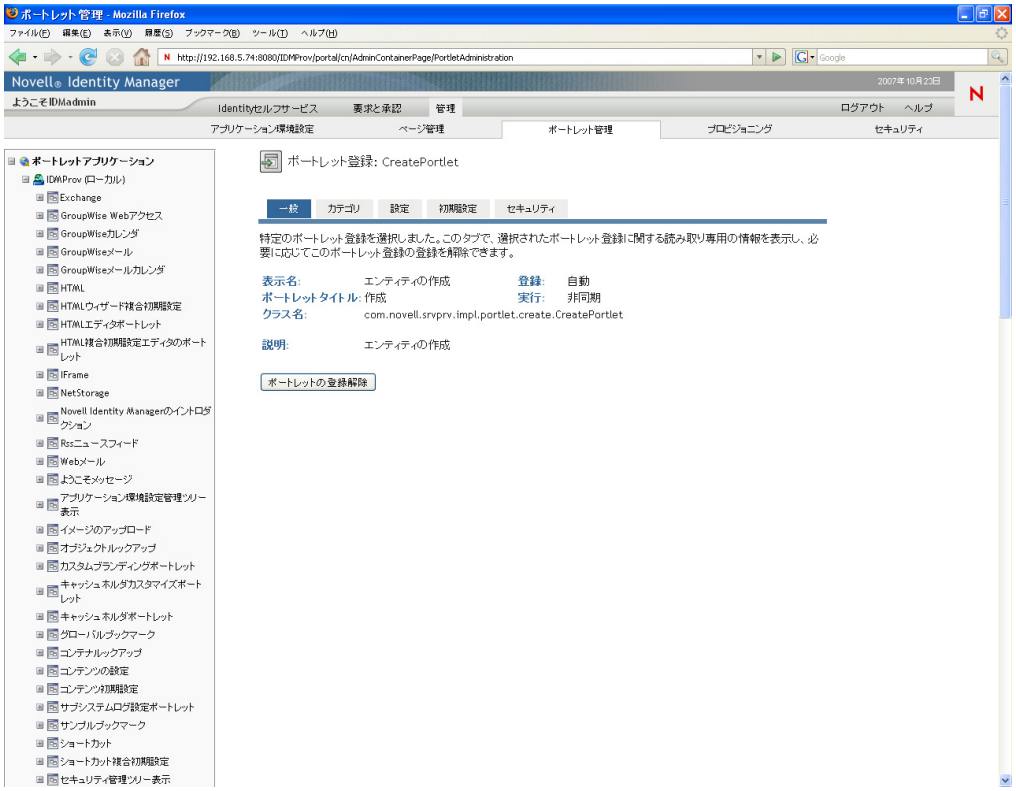
リストされたポートレット登録に関する次の情報を \'95\'5c 示できます (読み込み専用)。

- \'95\'5c 示名
- 登録のタイプ
- ポートレットのタイトル
- 実行のタイプ (同期または非同期)
- Class name
- 説明

[ポートレットアプリケーション] リストで、情報を表示するポートレット登録を選択します。

[一般] パネルが右側に表示され、図 7-1 のように選択したポートレット登録の情報が表示されます。

図 7-1 ポートレット登録 : 一般プロパティ



7.3.3 ポートレット登録へのカテゴリの割り当て

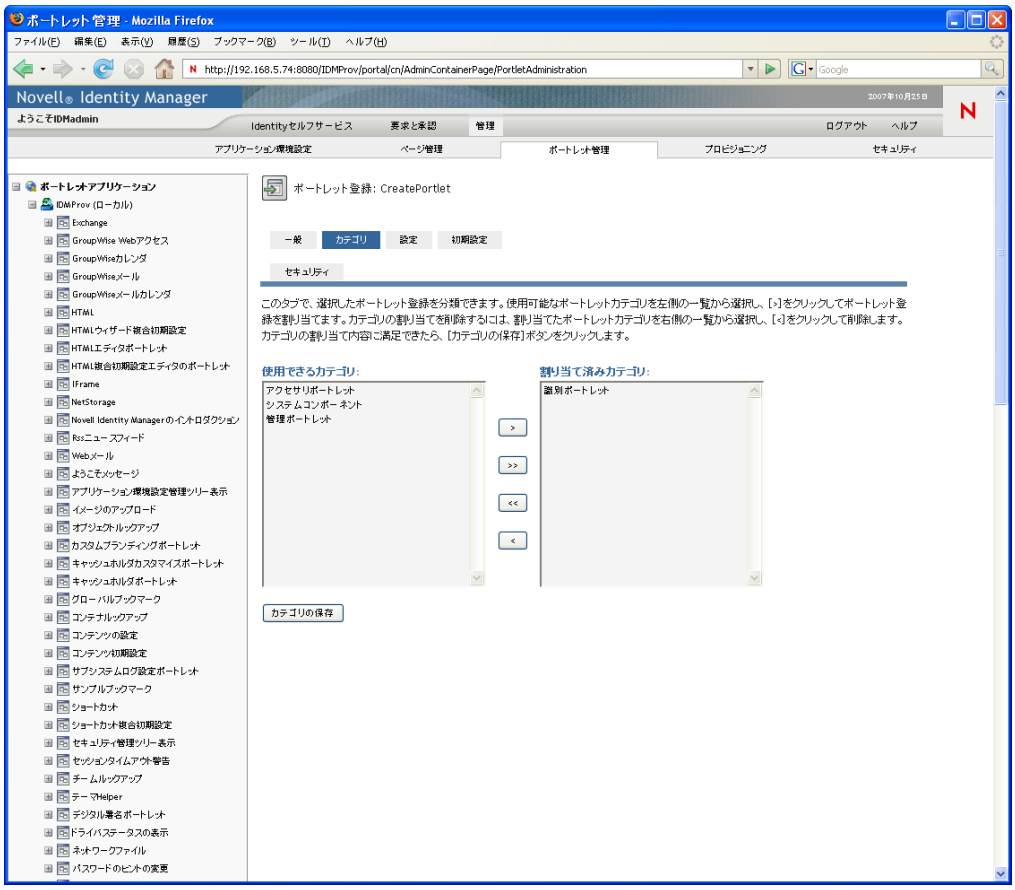
ポートレット登録をカテゴリ別に整理すると、ポートレットアプリケーションで特定のポートレットを容易に検索できます。

1 *[ポートレットアプリケーション]* リストで、カテゴリを設定するポートレット登録を選択します。

［General］パネルが右側に \'95\'5c 示されます。

2 ［Categories］パネルに移動します。

このパネルには、選択したポートレット登録で使用できるカテゴリのリスト、および割り当てられたカテゴリのリストが \'95\'5c 示されます。



3 次のように必要に応じて［*Assigned Categories*］リストを更新します。

必要な作業	操作手順
ポートレット登録に 1 つまたは複数のカテゴリを割り当てる	割り当てる各カテゴリを選択し、> をクリックします。
ポートレット登録にすべてのカテゴリを割り当てる	Click >>
1 つまたは複数のカテゴリの割り当てを削除する	削除する各カテゴリを選択し、< をクリックします。
すべてのカテゴリの割り当てを削除する	Click <<

4 ［*Save Categories*］をクリックします。

7.3.4 ポートレット登録の設定の変更

ポートレット設定では、ポータル (Identity Manager ユーザアプリケーション) が個別のポートレットと対話的にやり取りする方法を定義します。各ポートレットは次の内容が設定されています。

- 役職
- タイムアウトの最大時間

- 認証の必要性
- タイトルバーの \'95\'5c 示
- ユーザから隠す
- ポートレットアプリケーションで定義されたオプション

ポートレットアプリケーション WAR のポートレット展開記述子 (portlet.xml) で、標準 Java Portlet 1.0 の設定が定義されています。これらの設定値は、［ポートレット管理］ページを使用して登録別に変更できます。この場合、新しい値は選択したポートレット登録にのみ適用されます。

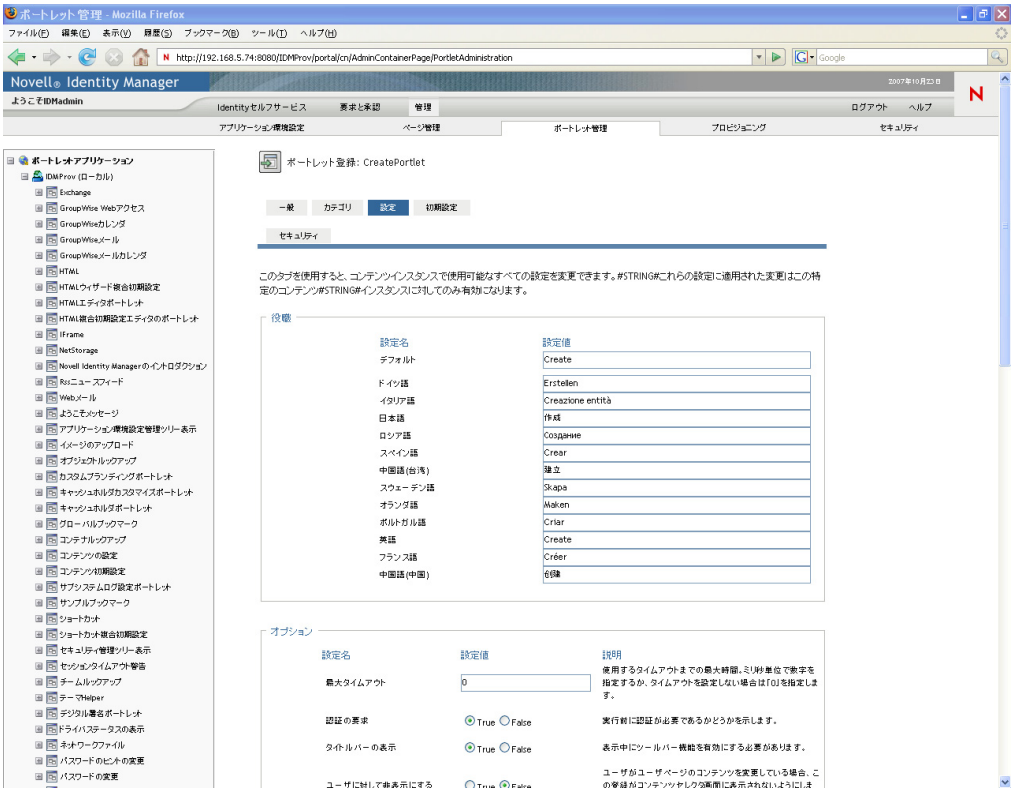
ポートレット登録の設定を変更する

1 ［Portlet Applications］リストで、設定を変更するポートレット登録を選択します。

［General］パネルが右側に \'95\'5c 示されます。

2 ［Settings］パネルに移動します。

このパネルには、選択したポートレット登録の現在の設定が \'95\'5c 示されます。



3 必要に応じて設定を変更します。

このパネルでの作業中、次のアクションも実行できます。

必要な作業	操作手順
保存していない変更を破棄する	［*Cancel*］をクリックします。

必要な作業	操作手順
このポートレット登録の設定をすべてデフォルト値に戻す (対応するポートレット定義に従います)。	［*Reset All*］をクリックします。
個別の設定をデフォルト値に戻す	設定の横にある［*Reset*］リンクをクリックします。

4 ［*Save Settings*］をクリックします。

7.3.5 ポートレット登録の初期設定の変更

ポートレットの初期設定は、ポートレットの設計時に開発者がポートレット展開記述子で定義します。初期設定は、ポートレットの開発者の実装に基づいてポートレットごとに異なります。

［ポートレット管理］ページを使用して、これらの初期設定値を登録ごとに変更できます。この場合、新しい値は選択したポートレット登録にのみ適用されます。

ポートレット登録の初期設定を変更する

1 *［ポートレットアプリケーション］*リストで、初期設定を変更するポートレット登録を選択します。

［General］パネルが右側に \'95\'5c 示されます。

2 ［Preferences］パネルに移動します。

このパネルには、選択したポートレット登録の現在の初期設定が \'95\'5c 示されます。

3 必要に応じて初期設定を変更します。

このパネルでの作業中、次のアクションも実行できます。

必要な作業	操作手順
初期設定の詳細情報を \'95\'5c 示する	［*Descriptions*］をクリックします。
保存していない変更を破棄する	［*Cancel*］をクリックします。
このポートレット登録の初期設定をすべてデフォルト値に戻す (対応するポートレット定義に従います)。	*［すべてリセット］*をクリックする
個別の初期設定をデフォルト値に戻す	各初期設定の横にある *［リセット］*リンクをクリックする

4 ポートレット定義で指定された各ロケールの初期設定のローカライズバージョンを変更する

4a 初期設定の横にある［*Detail*］*リンクをクリックします (リンクが \'95\'5c 示されている場合)。*

各ロケールの初期設定値がパネルに \'95\'5c 示されます。

4b 必要に応じて値を変更します。

4c ［*OK*］*をクリックして変更を適用し、初期設定のメインリストに戻ります。*

5 ［*Save Preferences*］*をクリックします。*

7.3.6 ポートレット登録のセキュリティ許可の割り当て

ポートレット登録のユーザ、グループ、およびコンテナに、表 7-2 で説明されているセキュリティ許可を割り当てられます。

表 7-2 *ポートレット登録のセキュリティ許可の割り当て*

許可	説明
リスト	ユーザは、選択したリストからポートレット登録を **\'95\'5c 示**できます。
Execute	ユーザは、ポータルページのポートレット登録を**実行**できます。

セキュリティ許可を変更した場合、新しい値は選択したポートレット登録にのみ適用されます。

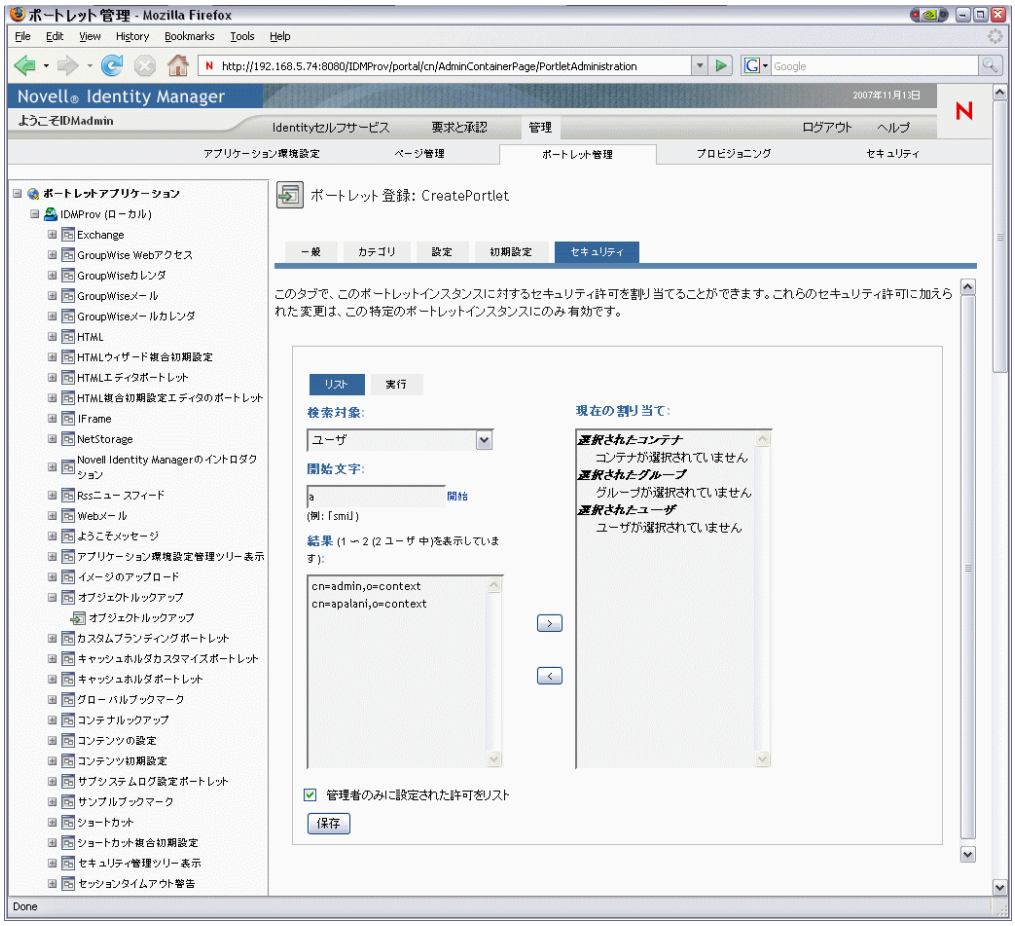
ポートレット登録のセキュリティ許可を割り当てる

1 *［ポートレットアプリケーション］*リストで、セキュリティ許可を変更するポートレット登録を選択します。

［General］パネルが右側に \'95\'5c 示されます。

2 ［Security］パネルに移動します。

このパネルには、選択したポートレット登録の現在のセキュリティ許可が \'95\'5c 示されます。



3 割り当てる許可のタイプに応じて、［*List*］または［Execute］タブに移動します。

4 次の検索設定の値を指定します。

設定	操作
Search for	次のいずれかをドロップダウンメニューから選択します。 ◆ ユーザ ◆ グループ ◆ コンテナ
Starts with	必要な作業 : ◆ 指定したタイプ (ユーザ、グループ、またはコンテナ) で使用できるオブジェクトすべてを検索する場合は、この設定を空白にします。 ◆ これらのオブジェクトのサブセットを検索して、目的の CN 値の開始文字を入力します。(大文字小文字は区別されません。ワイルドカードはサポートされていません)。 たとえば、S から始まるグループを検索する場合、検索結果は次のようになります : cn=Sales,ou=groups,o=MyOrg cn=Service,ou=groups,o=MyOrg cn=Shipping,ou=groups,o=MyOrg Se から始まるグループを検索した場合は、次のようになります : cn=Service,ou=groups,o=MyOrg

5 ［*Go*］*をクリックします。*

検索結果は、［Results］リストに \'95\'5c 示されます。

6 ポートレット登録に割り当てるユーザ、グループ、またはコンテナを選択して、［*Add(>)*］*\'83\'7b タンをクリックします。*

複数選択するには、<Ctrl> キーを押しながら選択します。

7 ポートレット登録のロックの有効または無効を次のように設定します。

必要な作業	操作手順
ポートレット登録をロックして、ユーザアプリケーション管理者だけがリスト／実行できるようにする	［管理者のみに設定された許可をリスト］/［管理者のみに設定された許可を実行］を選択します。
割り当てられたすべてのユーザ、グループ、およびコンテナがポートレット登録をリスト／実行できるようにする	［管理者のみに設定された許可をリスト］/［管理者のみに設定された許可を実行］の選択を解除します。 **注 :** この設定を解除した状態でポートレット登録に対して明示的に割り当てられたユーザ、グループ、またはコンテナがない場合、ユーザ全員がこのポートレット登録に対しリスト許可と実行許可を持つことになります。

8 ［*保存*］をクリックします。

7.3.7 ポートレット登録の解除

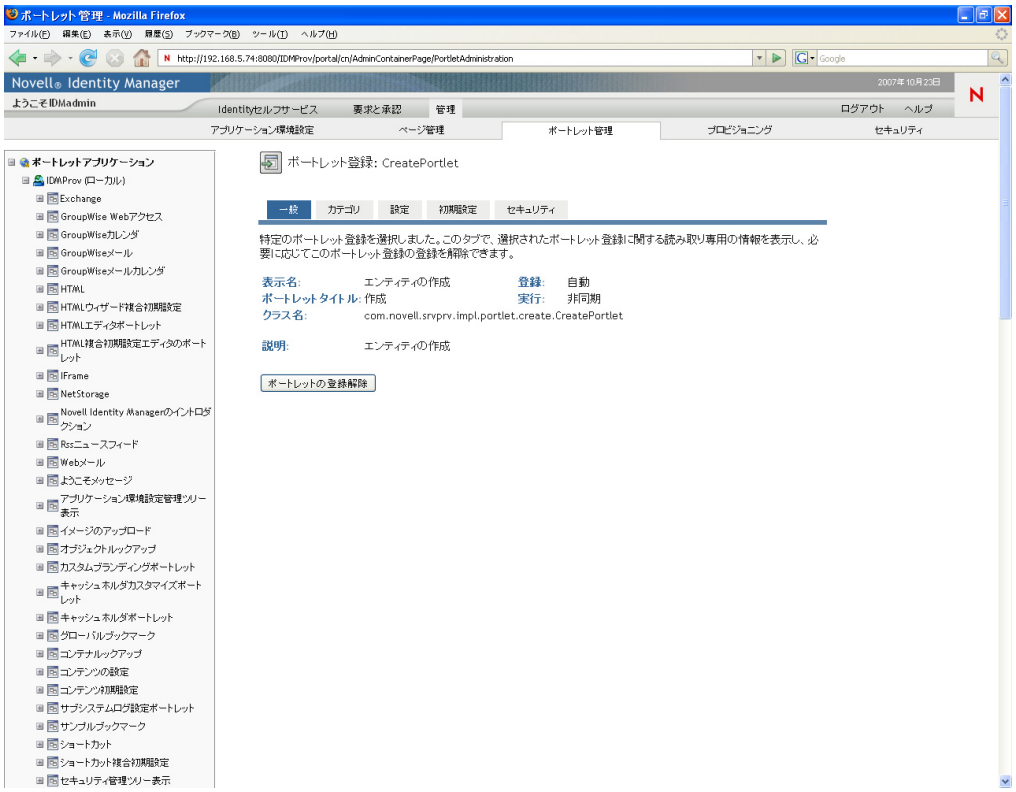
必要な場合は、［Portlet Admin］ページを使用してポートレットの登録を解除できます。

注 : 自動登録として定義されたポートレットは、登録を解除してもアプリケーションサーバの再起動時に自動的に再登録されます。

ポートレットの登録を解除する

1 ［Portlet Applications］リストで、登録を解除するポートレット登録を選択します。

［General］パネルが右側に \'95\'5c 示され、選択したポートレット登録の情報が \'95\'5c 示されます。



2 ［*Unregister Portlet*］をクリックします。

3 登録解除の操作を確認するメッセージが \'95\'5c 示されたら、［*OK*］をクリックします。

プロビジョニング環境設定

8

このセクションでは、［プロビジョニング環境設定］ページで行える作業について説明していきます。主なトピックは次のとおりです。

8.1 プロビジョニング環境設定について

この節では、［プロビジョニング］ページを使ったユーザアプリケーションのワークフローベースのプロビジョニング機能の管理方法について説明していきます。［プロビジョニング］ページにアクセスするには、Identity Manager のプロビジョニングモジュールが必要です。また、プロビジョニングアプリケーション管理者としてログオンする必要があります。

8.2 委任、代理人、タスクの環境設定

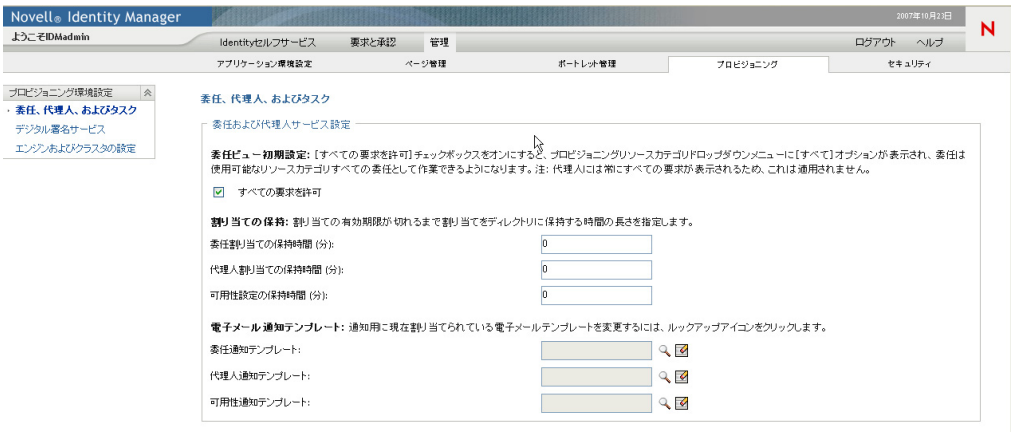
この節には、次の情報を記載しています。

8.2.1 委任と代理人サービスの環境設定

委任と代理人サービスを設定する

1. *［プロビジョニング］* タブを選択します。
2. 左側のナビゲーションメニューから、*［委任、代理人、およびタスク］* を選択します。

   ［委任、代理人、およびタスク］ページが表示されます。サービスを設定するには、［委任および代理人サービス設定］ボックスで一部のオプションを変更する必要があります。



3 [チームの代理人の割り当て] アクション用の [リソース検索条件] ドロップダウンリストに *[すべて]* のオプションを表示する場合は、*[すべての要求を許可]* オプションを選択します。*[すべて]* オプションを利用できる場合、すべてのリソースカテゴリに適用される委任割り当てを定義できます。

4 委任、代理人、および可用性割り当ての保持時間を定義します。

フィールド	説明
委任割り当ての保持時間	有効期限を過ぎた委任割り当てをディレクトリに保持する分数を指定します。デフォルトは 0 です。この場合、有効期限が切れた割り当ては、すぐに削除されます。
代理人割り当ての保持時間	有効期限を過ぎた代理人割り当てをディレクトリに保持する分数を指定します。デフォルトは 0 です。この場合、有効期限が切れた割り当ては、すぐに削除されます。
可用性設定の保持時間	有効期限を過ぎた可用性設定をディレクトリに保持する分数を指定します。デフォルトは 0 です。この場合、有効期限が切れた割り当ては、すぐに削除されます。

5 委任、代理人、および可用性通知に使用する電子メールテンプレートを選択します。

フィールド	説明
委任通知テンプレート	委任通知用電子メールに使用する、言語に依存しない名前のテンプレートを指定します。テンプレート名を指定すると、通知エンジンによりランタイム時に使用する言語固有のテンプレートが判断されます。 電子メールテンプレートの作成、編集方法については、336 ページのセクション 18.4「電子メールテンプレートに関する作業」を参照してください。

フィールド	説明
代理人通知テンプレート	代理人通知用電子メールに使用する、言語に依存しない名前のテンプレートを指定します。テンプレート名を指定すると、通知エンジンによりランタイム時に使用する言語固有のテンプレートが判断されます。 電子メールテンプレートの作成、編集方法については、336 ページのセクション 18.4「電子メールテンプレートに関する作業」を参照してください。
可用性通知テンプレート:	可用性通知用電子メールに使用する、言語に依存しない名前のテンプレートを指定します。テンプレート名を指定すると、通知エンジンによりランタイム時に使用する言語固有のテンプレートが判断されます。 電子メールテンプレートの作成、編集方法については、336 ページのセクション 18.4「電子メールテンプレートに関する作業」を参照してください。

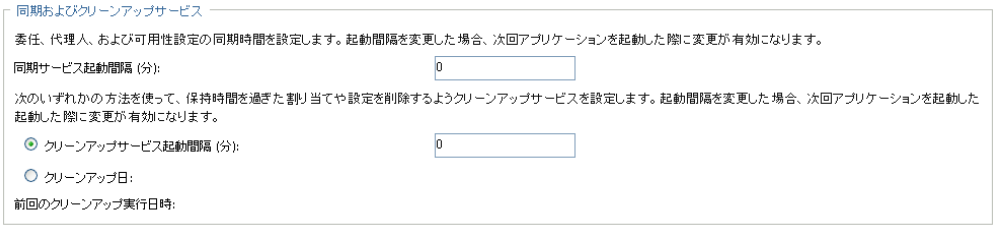
8.2.2 同期化とクリーンアップのスケジュール

同期およびクリーンアップサービスを環境設定する

1 *[プロビジョニング]* タブを選択します。

2 左側のナビゲーションメニューから、*[委任、代理人、およびタスク]* を選択します。

[委任、代理人、およびタスク] ページが表示されます。同期化とクリーンアップをスケジュールするには、[同期およびクリーンアップサービス] ボックスの各項目を設定します。



3 同期サービスの実施頻度を指定するには、[*同期サービス起動間隔*] フィールドに起動間隔 (分) を入力します。デフォルトは 0 です。この場合、サービスは実行されません。

同期サービスの実行時、委任割り当てに対して行われた変更 (または削除) は、対応するユーザの可用性設定と同期化されます。

4 クリーンアップサービスの実施頻度を指定するには、*[クリーンアップサービス起動間隔]* を選択して、起動間隔 (分) を入力します。代わりに *[クリーンアップ日 :]* を選択して、カレンダーからサービスを起動する日付を指定することもできます。デフォルトは 0 です。この場合、サービスは実行されません。

クリーンアップサービスが実行されると、無効になった代理／委任割り当てはすべてシステムから削除されます。

クリーンアップサービスが実行されると、*[前回のクリーンアップ実行日時:]* フィールドには前回クリーンアップが実行された日時が表示されます。

8.2.3 プロビジョニングインタフェース表示設定の環境設定

プロビジョニングインタフェース表示を設定する

1. *[プロビジョニング]* タブを選択します。
2. 左側のナビゲーションメニューから、*[委任、代理人、およびタスク]* を選択します。

   [委任、代理人、およびタスク] ページが表示されます。同期化とクリーンアップをスケジュールするには、[プロビジョニングインタフェース表示設定] ボックスの各項目を設定します。

3. デフォルトの表示ページを変更するには、*[デフォルトの表示ページ]* フィールドにページの URL を入力します。デフォルトページを次に示します。

   ```
   getAFTaskList.do?apwaLeftNavItem=JSP_MENU_TASKS&apwaActionType=user
   ```

   指定するページは、*[要求と承認]* タブで利用できるサーブレットを参照していなければなりません。表示ページを変更する場合、目的のページの左側のナビゲーションパネルの *[要求と承認]* をクリックして、アプリケーションコンテキスト (IDMProv) の後にある URL の最後の部分を切り取って、それを *[デフォルトの表示ページ]* フィールドに貼り付けることができます。たとえば、[マイ要求] を表示ページに設定する場合は、次の文字列をフィールドに貼り付けます。

   ```
   getAFProcessList.do?apwaLeftNavItem=JSP_MENU_REQUESTS&apwaActionScope=user&apwaNewSearch=true
   ```

4. 各クエリが返す行数を設定するには、*[クエリが返す結果の最大数:]* ボックスに行数の上限を入力します。デフォルトは 50 です。
5. 各ページに表示する行数を設定するには、*[1 ページに表示する結果の最大数:]* ボックスに表示する行数の上限を入力します。デフォルトは 5 です。

   [1 ページに表示する結果の最大数:] に指定した値は、*[要求と承認]* タブのさまざまな画面 ([マイタスク]、[マイ要求]、[マイ代理人割り当て]、[マイ委任割り当て] などを含む) に適用されます。
6. [チームのタスク] リストのビューを設定するには、*[テンプレート]* または *[表示]* ラジオボタンを選択します。

   ユーザに対してどのタスクビューを使用するかを判断するには、表 8-1 を参照してください。図 8-1 と図 8-2 に、表示内容が記載されています。

表 8-1 *［チームのタスク］のテンプレートビューと表示ビューの比較*

機能	テンプレートビュー	表示ビュー
ビューは表形式？	はい	はい
ビューはセクション 508 に準拠しているか？	はい	いいえ
デフォルトのテンプレートか？	はい	いいえ
1 回の検索で何件のタスクを表示できるか？	数千件以上。	数百件以上。数百件のアイテムで軽快なパフォーマンス。
フィルタ	［チームのタスク］ページには、［受信者］フィルタと［割り当て先］フィルタがあります。これらのフィルタのどちらかまたは両方を使って検索を行えます。	［チームのタスク］ページには、［受信者］フィルタと［割り当て先］フィルタがあります。これらのフィルタのどちらかまたは両方を使って検索を行えます。 また、新しく検索を実行せずに、取得したデータにフィルタを設定することもできます。表示ビューの右側にあるフィルタボックスから、1 つまたは複数のパラメータを選択してください。
列の値を基準にソート	はい . 列見出しをクリックするたびに、その列が昇順または降順でソートされます。	はい . 列見出しをクリックするたびに、その列が昇順または降順でソートされます。
列順のソート	［チームのタスク］ページで列見出しを選択した順序に列が表示されます。	［チームのタスク］ページで列見出しを選択した順序に列が表示されます。
ページ長	［チームのタスク］ページで、ページ長を 5、10、または 15 エントリに設定します。	ページ長は設定できませんが、次の作業を行えます。 ◆ フィルタを使ったタスクのサブセットの表示。 ◆ クリップボードにリストをコピーして、編集可能な .txt または .html レポートファイルを作成します。

図 8-1 *テンプレートビューの例*

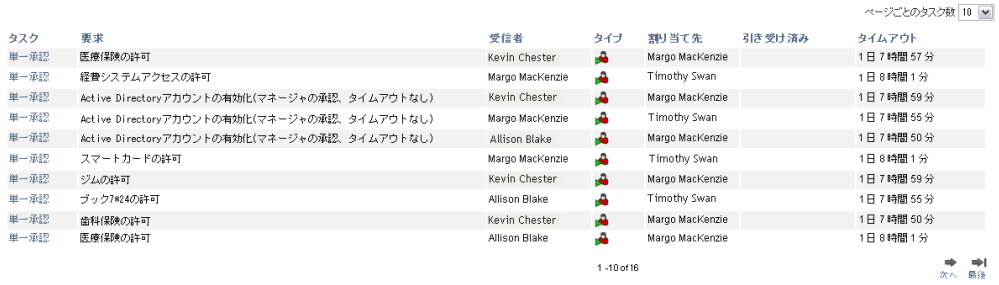


図 8-2 表示ビューの例

8.3 デジタル署名サービスの環境設定

この節では、デジタル署名サービスの環境設定について説明していきます。

デジタル署名サービスを設定する

1. *[プロビジョニング]* タブを選択します。
2. 左側のナビゲーションメニューから、*[デジタル署名サービス]* を選択します。

   [デジタル署名サービス] パネルが表示されます。

3. 次の手順で、デジタル署名サービスの環境設定を行います。

   3a *[デジタル署名サポートの有効化]* チェックボックスを選択します。

   このチェックボックスを選択しないと、デジタル署名を必要とするプロビジョニングリソースにユーザがアクセスした場合、エラーメッセージが表示されます。

デジタル署名サポートを有効にする前に、必要な JAR がすべて存在していることを確認してください。必要な JAR が 1 つでも存在しないときにこのチェックボックスを選択すると、エラーメッセージが表示されます。デジタル署名に必要な JAR の詳細は、37 ページのセクション 2.3「デジタル署名の環境設定」を参照してください。

3b XML 署名を使用する場合は、*[XML 署名の使用]* チェックボックスを選択します。(このオプションは、cryptvision を使用する場合に必要です)

3c 必要に応じて、ユーザに署名済みドキュメントのプレビューを許可する *[署名済みドキュメントのプレビューを有効にする]* を選択します。

3d *[クラス名]* フィールドに、デジタル署名サービスのクラス名を入力します。

署名検証プロバイダとして cryptovision を使用する場合の詳細は、http://www.cryptovision.com/idmdigsig.html を参照してください。

3e 必要に応じて、*[代替証明書の件名仮想エンティティキー]* フィールドにエンティティキーを指定します。エンティティキーは、データ抽象化層に定義されているエンティティにマップします。エンティティは、権限のあるユーザのみがデジタル署名を行えることを保証するための、LDAP 共通名の代わりに使用できる計算済み属性を提供します。Designer では、エンティティを定義して、任意のキー名を設定できます。［デジタル署名］環境設定パネルでは、定義したエンティティのキーを指定します。代替件名はオプション機能で、保護層を追加するために使用できます。

3f 必要に応じて *[証明書の認証]* チェックボックスを選択します。これにより、認証ユーザと、選択したユーザ証明書に対応するユーザが一致することが保証されます。*[証明書の認証]* を有効にした場合、現在のユーザが、別のユーザに対して与えられたスマートカードの証明書を使用することはできません。

3g 証明書の使用が有効かどうかを確認する前に、アプリケーションに証明書取り消しリスト (CRL) をチェックさせる場合は、*[取り消しチェックの有効化]* を選択します。何らかの理由で証明書が取り消されていることもあります。たとえば、認証局が特定の証明書が不適切に発行されたと判断することがあります。また、証明書の秘密鍵をなくした、または盗まれたなどの理由で証明書が取り消されることもあります。

3h 証明書を使用する前に、OCSP(Online Certificate Status Protocol) サーバにクエリを行う場合は、*[OCSP クエリの有効化]* チェックボックスを選択します。OCSP は、証明書取り消しリストの代替手段で、PKI での CRL の使用に関連する問題に対処するものです。サーバの OCSP アクセスポイントは、ユーザアプリケーション環境設定ユーティリティで指定します。

4 以前に設定したアプレットの設定を表示するには、*[署名アプレット]* ドロップダウンリストでアプレットを選択します。

cryptovision アプレットの環境設定の詳細は、http://www.cryptovision.com/idmdigsig.html を参照してください。

5 新しい署名アプレットの環境設定を追加するには、次の手順に従ってください。

5a ［*追加*］をクリックします。

［署名アプレット］パネルのフィールドが編集可能になります。

5b *［表示名］*フィールドに、このアプレットの環境設定名を指定します。

5c *［クラス ID］*フィールドに、アプレットのクラス ID を指定します。

5d *［アーカイブ名］*フィールドに、アプレットのある JAR のエントリを指定します。

5e *コンテキストルート*のアプレットアーカイブがある Web アプリケーションの、<コンテキストルートパス > を指定します。(コンテキストルートが別のアプリケーションを指している場合は、常に「/」文字で開始してください)

5f *［コールバック名］*フィールドに、コールバック名を指定します。

5g *［宣言テンプレート］*フィールドに、XML 宣言文字列を入力します。

5h *［起動テンプレート］*フィールドに、起動文字列を入力します。

5i *［コールバック関数テンプレート :］*フィールドに、コールバック関数を指定します。

5j *［ブラウザタイプ］*リストから、ブラウザタイプ (例 :Internet Explorer 6.0) を選択します。

6 *［保存］*をクリックして、設定内容を保存します。

8.4 ワークフローエンジンとクラスタの環境設定

この節では、ワークフローエンジンとクラスタの環境設定について説明していきます。これらの設定は、クラスタ内のすべてのエンジンに適用されます。いずれかの設定が変更されると、クラスタ内の他のエンジンはデータベースの変更を検出して、新しい値を使用します。エンジンは、［保留中のプロセス間隔］に指定された間隔で、これらの設定の変更をチェックします。

プロセスキャッシュの設定とハートビートの設定を変更した場合、変更内容を有効にするにはサーバを再起動する必要があります。

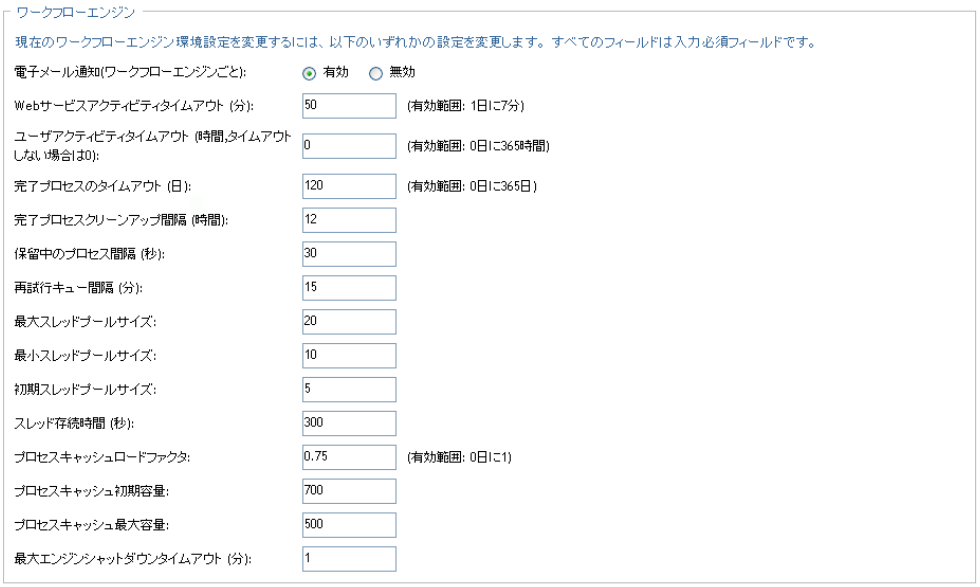
8.4.1 ワークフローエンジンの環境設定

ワークフローエンジンを設定する

1 *［プロビジョニング］タブ*を選択します。

2 左側のナビゲーションメニューから、*［エンジンおよびクラスタの設定］*を選択します。

［ワークフロー環境設定］ページが表示されます。エンジンを環境設定するには、［ワークフローエンジン］で設定項目を変更する必要があります。



3 エンジン設定を変更するには、目的のフィールドをクリックして、新しい値を入力します。エンジン設定については、下で説明しています。

エンジン設定	説明
電子メール通知 (ワークフローエンジンごと)	ワークフローエンジン全体に対する電子メール通知を有効／無効にします。デフォルトでは、このオプションは有効になっています。
Web サービスアクティビティタイムアウト (分)	デフォルトの Web サービスアクティビティタイムアウトを分で指定します。デフォルトは 50 分です。
ユーザアクティビティタイムアウト (時間、0 を指定するとタイムアウトなし)	デフォルトのユーザアクティビティタイムアウト時間を指定します。デフォルトは 0 日です。この場合、タイムアウトにはなりません。
完了プロセスのタイムアウト (日)	完了したプロセス状態をシステムに保持する日数を指定します。デフォルトは 120 日です。
完了プロセスクリーンアップ間隔 (時間)	エンジンが、完了プロセスのタイムアウトに指定されている時間が経過した完了プロセスを削除するための、チェック頻度を指定します。デフォルトは 12 時間です。

エンジン設定	説明
保留中のプロセス間隔 (秒)	プロセスがバインドされていないエンジンで実行されているユーザアクティビティは、保留状態に移行されます。この間隔は、アクティビティの実行を継続するために、保留中のアクティビティをチェックする頻度を表します。デフォルトは 30 秒です。
再試行キュー間隔 (分)	データベース接続上の問題が疑われるために失敗したアクティビティは、再試行キューに移動されます。この間隔は、エンジンがこれらのアクティビティの再試行を試みる間隔を表します。デフォルトは 15 分です。
最大スレッドプールサイズ	エンジンがアクティビティの実行に使用するスレッドの最大数です。デフォルトは 20 です。
最小スレッドプールサイズ	エンジンがアクティビティの実行に使用するスレッドの最小数です。スレッドの要求時にプールに最小値未満のスレッドしかない場合、プール内にアイドル状態のスレッドがある場合でもプールにスレッドが作成されます。デフォルトは 10 です。
初期スレッドプールサイズ	プール内でのスレッドの作成時に、事前開始されたスレッド数です。デフォルトは 5 です。
スレッド存続時間 (秒)	プールが最小サイズより大きい場合に、スレッド存続時間を超えてアイドル状態の超過スレッドは破棄されます。デフォルトは 5 分です。
プロセスキャッシュロードファクタ	ロードファクタには、キャッシュがどの程度一杯になったら容量を増やすのかを指定します。キャッシュ内のエントリ数が｛ロードファクタ x 現在の容量」を超えた場合に、容量が増やされます。デフォルトは 0.75 です。
プロセスキャッシュ初期容量	プロセスキャッシュは、ハッシュマップによりバックアップされています。この容量は、ハッシュマップ内のバケット数です。初期容量は、キャッシュ作成時のバケット数です。デフォルトは 700 です。

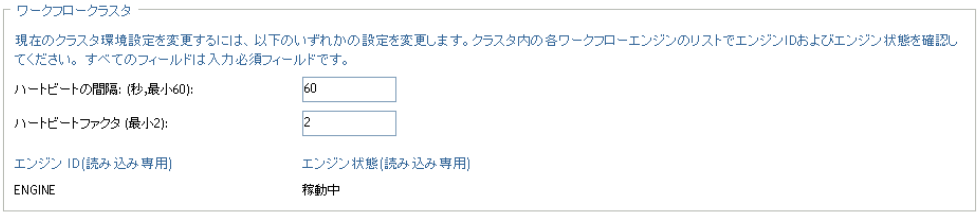
エンジン設定	説明
プロセスキャッシュ最大容量	キャッシュ内のプロセス数がここに指定した値以上の場合、キャッシュにプロセスを追加する前に、もっとも古いアクティブではないプロセスの削除が試みられます。この最大容量は緩やかな制限です。キャッシュ内にアクティブなプロセスしかない場合は、キャッシュ内のプロセス数がここに指定した値を超えることもあります。 ここには、プロセスキャッシュ初期容量とプロセスキャッシュロードファクタの積の値未満の値を設定することをお勧めします。そうすることにより、キャッシュ容量が増やされる前に、まずアクティブではない古いプロセスが削除されます。 次の例を参照してください。 プロセスキャッシュ初期容量 :700 プロセスキャッシュロードファクタ :0.75 プロセスキャッシュ最大容量 :500 キャッシュ内のプロセス数 :500 この場合、「初期容量 x ロードファクタ」は 525 になるため、キャッシュ内のプロセス数が 525 になると、キャッシュ容量が増やされ再ハッシュが行われます。キャッシュ内にはプロセスが 500 あるため、このままプロセスが増えるとまもなく (525 になると) サイズが増やされて再ハッシュが行われます。キャッシュに他のプロセスが追加されると、まず一番使われていない非アクティブなプロセスが削除されます。 デフォルトは 500 です。
最大エンジンシャットダウンタイムアウト (分)	エンジンは、一定時間の余裕を持ってシャットダウンを行います。シャットダウン時には、新しいアクティビティのキューへの格納が中止され、すでにキューに格納されているアクティビティの処理完了が試みられます。このタイムアウトには、キューに格納されているすべてのアクティビティとスレッドの実行が完了するまでにエンジンが待機する最大時間を指定します。ここに指定した時間が経過すると、キュー内のアクティビティの処理が停止され、アクティビティを実行しているすべてのスレッドの中断が試みられます。デフォルトは 1 分です。

8.4.2 ワークフロークラスタの環境設定

ワークフロークラスタを設定する

1. *[プロビジョニング] タブ*を選択します。
2. 左側のナビゲーションメニューから、*[エンジンおよびクラスタの設定]* を選択します。

   [ワークフロー環境設定] ページが表示されます。クラスタを設定するには、[ワークフロークラスタ] ボックスで設定項目を変更する必要があります。



3 クラスタ設定を変更するには、目的のフィールドをクリックして、新しい値を入力します。クラスタ設定については、下で説明しています。

クラスタ設定	説明
ハートビートの間隔 (秒、最小 60)	ワークフローエンジンのハートビートを更新する間隔を指定します。 ワークフローエンジンの起動時に、クラスタ内の他のノードがそのエンジン ID をすでに使用していないかどうかが確認され、その ID が使用中の場合は、起動が拒否されます。エンジン ID とエンジンの状態は、ユーザーアプリケーションデータベースが管理しています。エンジンがクラッシュして再起動された場合、データベース内の最終状態は、そのエンジンがまだ動作中であることを示します。そのため、ワークフローエンジンはハートビートを指定した間隔で書き込むハートビートタイマーを使用します。このタイマーは、その ID を持つエンジンがクラスタ内で動作しているかどうかを判断するために用いられます。そのエンジンがすでに動作している場合は、起動が拒否されます。 ハートビート間隔の最小値は 60 秒です。
ハートビートファクタ (最小 2)	ハートビートファクタを指定します。「ファクタ x ハートビート間隔」がハートビートタイムアウトになります。 タイムアウト時間を経過すると、ハートビートはタイムアウトとみなされます。 ハートビートファクタの最小値は 2 です。

セキュリティ設定

9

この節では、Identity Manager ユーザアプリケーションの *[管理]* タブの［セキュリティ］ページを使用する方法について説明します。主なトピックは次のとおりです。

[管理] タブにアクセスして操作するための一般的な情報については、81 ページの第 4 章「［Administration］タブの使用」を参照してください。

9.1 セキュリティの環境設定について

Identity Manager 3.5 ユーザアプリケーションは、プロビジョニングアプリケーション管理者とユーザアプリケーション管理者に管理作業を割り当てます。

表 9-1 *管理者のタイプ*

役割	実施できる作業
ユーザアプリケーション管理者	ユーザアプリケーションの *[管理]* タブでのアプリケーション管理作業。
プロビジョニングアプリケーション管理者	ユーザアプリケーションの *[要求と承認]* タブでの、プロビジョニングワークフロー管理作業。

これらの役割は、インストール時に、または Identity Manager ユーザアプリケーションの *[管理]* タブにある［セキュリティ］ページから割り当てられます。インストール時にこれらの役割を割り当てると、IDM が割り当て内容をユーザアプリケーション環境設定ファイルに書き込みます。このファイルは、configupdate ユーティリティを使って編集できます。ただし、WAR の展開時に、割り当てはユーザアプリケーションデータベースに書き込まれます。そのため、インストール後最初に JBoss アプリケーションサーバを起動した後は、configupdate ユーティリティを使ってこれらの割り当てを変更することはできません。［セキュリティ］ページから変更してください。226891

9.1.1 ユーザアプリケーション管理者

ユーザアプリケーション管理者は、Identity Manager ユーザアプリケーションの *［管理］* パネルから、Identity Manager ユーザアプリケーションの管理作業を行います。ユーザアプリケーション管理者にはプロビジョニング管理権はありません。*［要求と承認］* パネルの使用時には、一般ユーザとみなされます。複数のユーザアプリケーション管理者がいる場合もあります。

インストール時には、1 人のユーザをユーザアプリケーション管理者として割り当てる*必要があります*。インストール時に作成するユーザアプリケーション管理者は、プロビジョニングシステムを含めユーザアプリケーション内のすべてを管理できます。また、他の

ユーザをユーザアプリケーション管理者やプロビジョニングアプリケーション管理者として指名できます。

一般的に、ユーザアプリケーション管理者に指名するユーザは、ユーザアプリケーションの LDAP 環境設定に指定されているユーザルートコンテナになければなりません。こうすることにより、そのユーザはユーザ名だけでログインできます。毎回完全識別名を指定する必要はありません。

ユーザアプリケーション管理者となるユーザには、特別なディレクトリ権限は必要ありません。この役割は、アプリケーションレベルのアクセスを制御します。

注 : 必要に応じて、ユーザアプリケーション管理者は、1 人または複数のエンドユーザに対し、*[管理]* タブの特定のページへのアクセス許可を割り当てることができます。これらの許可の割り当てには、*[管理]* タブの［ページ管理］ページを使用します。(詳細については、139 ページの第 6 章「ページの管理」を参照してください)。

9.1.2 プロビジョニングアプリケーション管理者

プロビジョニングアプリケーション管理者は、ユーザアプリケーションではなく、プロビジョニングシステムを管理します。プロビジョニングアプリケーション管理者には、*[要求と承認]* パネル内のすべての機能に対する権限と許可があります (本質的にスーパユーザ)。

プロビジョニングアプリケーション管理者は、インストール時に割り当てます。システムを安全に保つためにも、インストール後には最低 1 人のプロビジョニングアプリケーション管理者を作成してください。プロビジョニングアプリケーション管理者がいない場合、ログインする各ユーザはすべてプロビジョニングアプリケーション管理者として扱われます。こうなると、システムの安全性は保てません。

プロビジョニングアプリケーション管理者は、他のユーザをプロビジョニングアプリケーション管理者として指名できます。ただし指名するには、管理コンソールの［プロビジョニング管理者の割り当て］ページにアクセスするために、ユーザアプリケーション管理者でなければなりません。

ユーザアプリケーションの LDAP 環境設定に指定されているユーザルートコンテナ内のユーザを、プロビジョニングアプリケーション管理者として指定できます。こうすることにより、そのユーザはユーザ名だけでログインできます。毎回完全識別名を指定する必要はありません。

9.2 ユーザアプリケーション管理者の割り当て

ユーザアプリケーション管理者を割り当てる場合、ユーザ、グループ、またはコンテナを指定できます。

1 ［Security］ページに移動します。

2 *［管理者の割り当て］で、［ユーザアプリケーション管理の割り当て］*を選択します。

3 次の検索設定の値を指定します。

設定	操作
検索対象	次のいずれかをドロップダウンメニューから選択します。 ◆ ユーザ ◆ グループ ◆ コンテナ
Starts with	必要な作業 : ◆ 指定したタイプ(ユーザ)で使用できるオブジェクトすべてを検索する場合は、この設定を空白にします。 ◆ これらのオブジェクトのサブセットを検索して、目的の CN 値の開始文字を入力します。(大文字小文字は区別されません。ワイルドカードはサポートされていません)。

4 *［Go］をクリックします。*

検索結果は、［Results］リストに \'95\'5c 示されます。

5 ユーザアプリケーション管理者を割り当てるユーザ、グループ、またはコンテナを選択して、*［追加 (>)］*をクリックします。

複数選択するには、<Ctrl> キーを押しながら選択します。

6 *［保存］*をクリックします。

ユーザアプリケーション管理者の割り当てを解除する

1 ［現在の割り当て］のリストで、ユーザアプリケーション管理者としての割り当てを解除するユーザ、グループ、またはコンテナを選択して、*［削除 (<)］* ボタンをクリックします。

   複数選択するには、<Ctrl> キーを押しながら選択します。

2 ［*保存*］をクリックします。

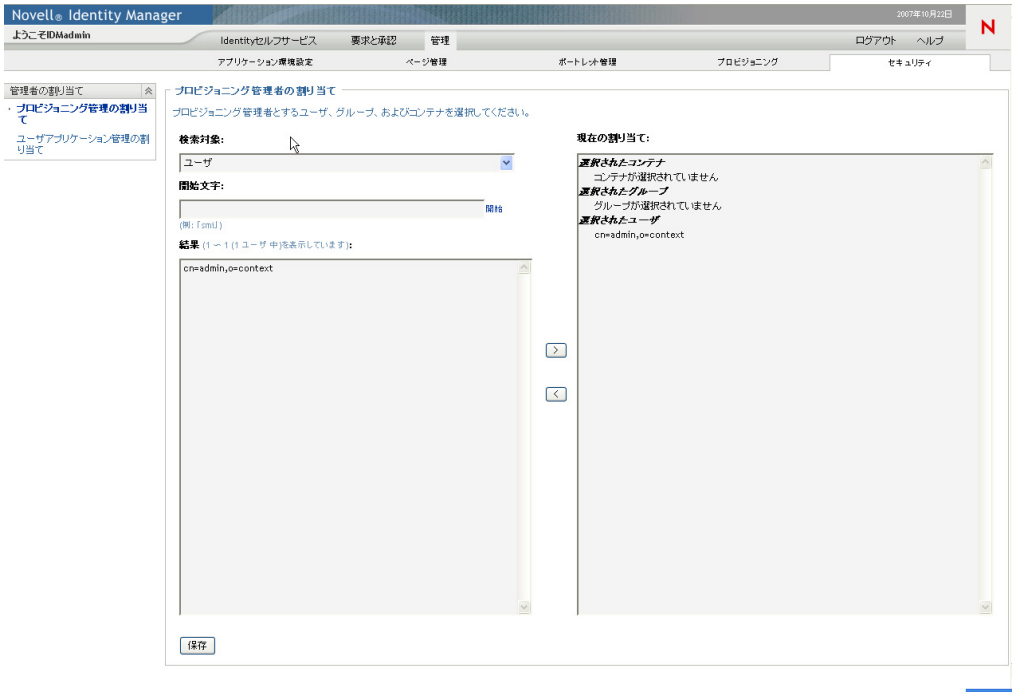
自分をユーザアプリケーション管理者から削除することはできません。これは、最低 1 人のユーザアプリケーション管理者が常に必要なためです。

9.3 プロビジョニング管理者の割り当て

プロビジョニング管理者を割り当てる場合、ユーザ、グループ、またはコンテナを指定できます。

1 ［セキュリティ］ページに移動します。

2 *［管理者の割り当て］* で、*［プロビジョニング管理の割り当て］* を選択します。



3 割り当てるユーザ、グループ、またはコンテナを検索します。次の検索設定の値を指定します。

設定	操作
検索対象	次のいずれかをドロップダウンメニューから選択します。 ◆ ユーザ ◆ グループ ◆ コンテナ

設定	操作
Starts with	必要な作業 : ◆ 指定したタイプ (ユーザ、グループ、またはコンテナ) で使用できるオブジェクトすべてを検索する場合は、この設定を空白にします。 ◆ これらのオブジェクトのサブセットを検索して、目的の CN 値の開始文字を入力します。(大文字小文字は区別されません。ワイルドカードはサポートされていません)。 たとえば、S から始まるグループを検索する場合、検索結果は次のようになります : cn=Sales,ou=groups,o=MyOrg cn=Service,ou=groups,o=MyOrg cn=Shipping,ou=groups,o=MyOrg Se から始まるグループを検索した場合は、次のようになります : cn=Service,ou=groups,o=MyOrg

4 *[開始]* をクリックします。検索結果は、[Results] リストに \'95\'5c 示されます。

5 プロビジョニング管理者を割り当てるユーザ、グループ、またはコンテナを選択して、*[追加 (>)]* をクリックします。

複数選択するには、<Ctrl> キーを押しながら選択します。

6 [*保存*] をクリックします。

プロビジョニングアプリケーション管理者の割り当てを解除する

1 [現在の割り当て] のリストで、ユーザアプリケーション管理者としての割り当てを解除するユーザ、グループ、またはコンテナを選択して、*[削除 (<)]* をクリックします。

複数選択するには、<Ctrl> キーを押しながら選択します。

2 [*保存*] をクリックします。

プロビジョニングアプリケーション管理者を削除する場合、最低 1 人のプロビジョニングアプリケーション管理者は残すようにしてください。システムの安全性を保つために、最低 1 人の管理者が必要です。最後の 1 人のプロビジョニングアプリケーション管理者を削除しようとした場合、警告メッセージが表示されます。

ポートレットリファレンス

IV

これらの節では、Identity Manager ユーザインタフェースで使用される識別情報およびシステムポートレットを設定する方法について説明します。

ポートレットについて

10

この節では、Identity Manager ユーザアプリケーションで使用できるポートレットについて説明しています。主なトピックは次のとおりです。

ポートレットの管理についての詳細は、173 ページの第 7 章「ポートレットの管理」を参照してください。

多くのポートレットには、その動作や外観をカスタマイズできる初期設定があります。これらの初期設定をローカライズするには、[*コンテンツ初期設定*] ページの [詳細] リンクをクリックします。一般的なガイドラインとして、初期設定の値が自由形式のテキスト入力フィールドの場合、値がユーザインタフェースに表示するメッセージの時以外はローカライズしないでください。ただし、初期設定名と説明はローカライズできます。メッセージ以外の値を持つ初期設定値をローカライズすると、ポーレットが誤動作する可能性があります。

10.1 アクセサリポートレット

アクセサリポートレットは、Identity Manager ユーザアプリケーションに追加できるさまざまな機能のセットを提供します。アクセサリポートレットは、電子メール、ファイルシステムなどの機能を提供します。詳細は、『*Identity Manager Accessory Portlet Reference Guide*』(アクセサリポートレットリファレンスガイド) を参照してください。

10.2 管理ポートレット

[Admin] カテゴリのポートレットは、ユーザインタフェースのレイアウトおよびコンテンツの制御に使用します。

重要 : これらのポートレットは使用、変更できません。これらのポートレットは、ユーザアプリケーションにフレームワークサービスを提供するものです。

管理ポートレットについては、表 10-1 を参照してください。

表 10-1 *管理ポートレット*

ポートレット名	説明
Header Portlet	ヘッダ情報およびユーザインタフェースのトップレベルのタブ制御を \'95\'5c 示します。 このポートレットには、環境設定はありません。

ポートレット名	説明
Shared Page Navigation	Identity Manager ユーザアプリケーションの共有ページを含むメニューを表示します。 初期設定は、表示内容および表示方法を定義しています。 詳細については、210 ページのセクション 10.2.1「共有ページナビゲーションポートレット」を参照してください。

10.2.1 共有ページナビゲーションポートレット

共有ページナビゲーションポートレットは、Identity Manager ユーザアプリケーションの共有ページへのリンクを生成します。表示する共有ページリンクは、初期設定に定義します。共有ページナビゲーションポートレットの初期設定の詳細は、210 ページの 表 10-2 を参照してください。

表 10-2 *共有ページナビゲーションポートレット : 初期設定*

初期設定	指定する内容
sharedpages-sorting	共有ページがカテゴリ内で表示される順序 : 昇順／降順。
sharedpages-sortmode	共有ページのソート方法 : アルファベット順／優先度。
sharedpages-category	共有ページの、1 つまたは複数のカテゴリを指定します。 カテゴリ名はヘッダとして表示され、そのカテゴリにあるすべての共有ページはリンクとして表示されます。カテゴリに共有ページがない場合は表示されません。カテゴリにない共有ページは、カテゴリ未分類として表示されます。
guest-category	ポータル待ち受けページに表示するポートレットの属するカテゴリを指定します。これは、すでに存在するカテゴリである必要があります。このカテゴリに含まれるページについては、ACL 読み込み制約があってはなりません。

10.3 ［Identity］ポートレット

識別ポートレットは、Identity Manager ユーザアプリケーションの *［Identity セルフサービス］* タブで使用されます。識別ポートレットについては、210 ページの 表 10-3 を参照してください。

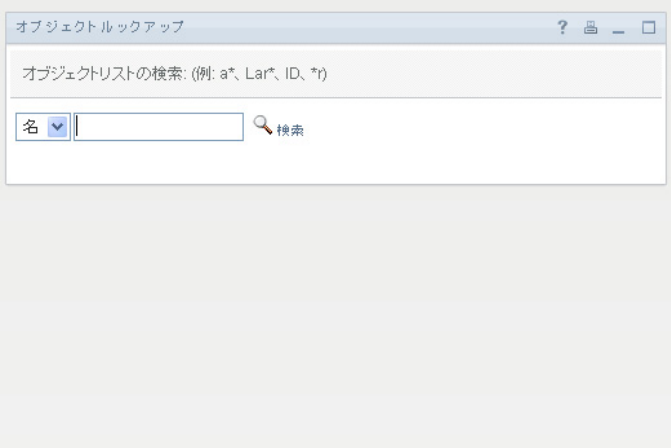
表 10-3 *識別ポートレット*

ポートレット名	説明
関連付けレポート	ログオンしているユーザの DirXML-Associations 属性を表示します。この属性は、ユーザと外部アプリケーションをマップします。このポートレットには、環境設定はありません。

ポートレット名	説明
作成	アイデンティティボールトにオブジェクトを作成するための、ウィザードベースのインタフェースを提供します。 詳細については、213 ページの第 11 章「[Create] ポートレットの環境設定」を参照してください。
詳細	エンティティの属性データを \'95\'5c 示したり、操作したりすることができます。 詳細については、223 ページの第 12 章「[Detail] ポートレットの環境設定」を参照してください。
組織図	アイデンティティボールトとのオブジェクト間の階層リレーションシップを表示したり、参照したりすることができます。 詳細については、239 ページの第 13 章「[Org Chart] ポートレットの環境設定」を参照してください。
リソース要求	匿名ユーザやゲストユーザへのリソース要求にアクセスできます。このポートレット用の共有ページを新たに作成し、そのページをゲストユーザや匿名ユーザが利用できるようにする必要があります。詳細については、267 ページの第 14 章「リソース要求ポートレット」を参照してください。
検索リスト	アイデンティティボールトにあるオブジェクトを検索できます。 詳細については、269 ページの第 15 章「[Search List] ポートレットの環境設定」を参照してください。

実行時に、識別ポートレットはユーザの操作に応じて、ContainerLookup ポートレットまたは ParamLookup ポートレットを呼び出すこともあります。ContainerLookup ポートレットは、ユーザがコンテナオブジェクトに対してルックアップを実行した場合に識別ポートレットにより起動されます。一方、ParamLookup ポートレットは、ユーザが属性に対してルックアップを実行した場合に起動されます。ユーザは、[ルックアップ] ボタンをクリックしてこれらのポートレットを起動します。これらのポートレットの実行時の外観は似ています。

図 10-1 *サンプルの ParamLookup ポートレット*



これらのポートレットはオブジェクトセレクタと呼ばれることもあります。ポートレットのコンテンツは、ディレクトリ抽象化層の DNLookup 定義に定義されています。これら

のポートレットに初期設定はありません。ページに追加することもできません。ただし、識別ポートレットにゲストアクセスを許可した場合にのみ、それらを変更できる場合があります。ゲストアクセスを許可した場合に必要な変更については、各識別ポートレットの参照セクションで説明しています。

10.4 システムコンポーネント

システムポートレットは、Identity Manager ユーザアプリケーションにサービスを提供します。

重要 : このカテゴリのポートレットを使用、変更することはできません。

システムポートレットを 212 ページの 表 10-4 に示します。

表 10-4 *システムポートレット*

ポートレット名	説明
Portal Page Controller	［Shared Page Navigation］ポートレットを介して現在選択している共有ページを \'95\'5c 示します。 このポートレットには、環境設定はありません。

［Create］ポートレットの環境設定 11

この節では、Identity Manager ユーザアプリケーションで作成ポートレットを使用する方法について説明します。主なトピックは次のとおりです。

11.1 ［Create］ポートレットについて

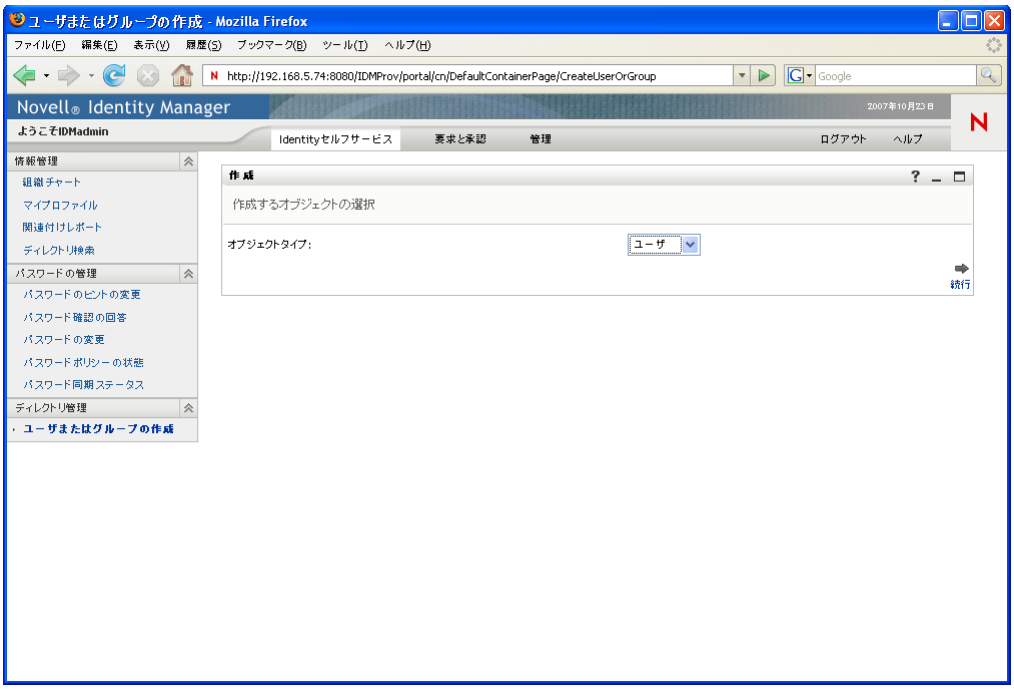
作成ポートレットは、さまざまなタイプのアイデンティティボールトブジェクトを作成するための、使いやすいウィザードを提供します。ポートレットの初期設定コントロールを次に示します。

- ユーザが作成するオブジェクトのタイプ
- ユーザが入力する属性

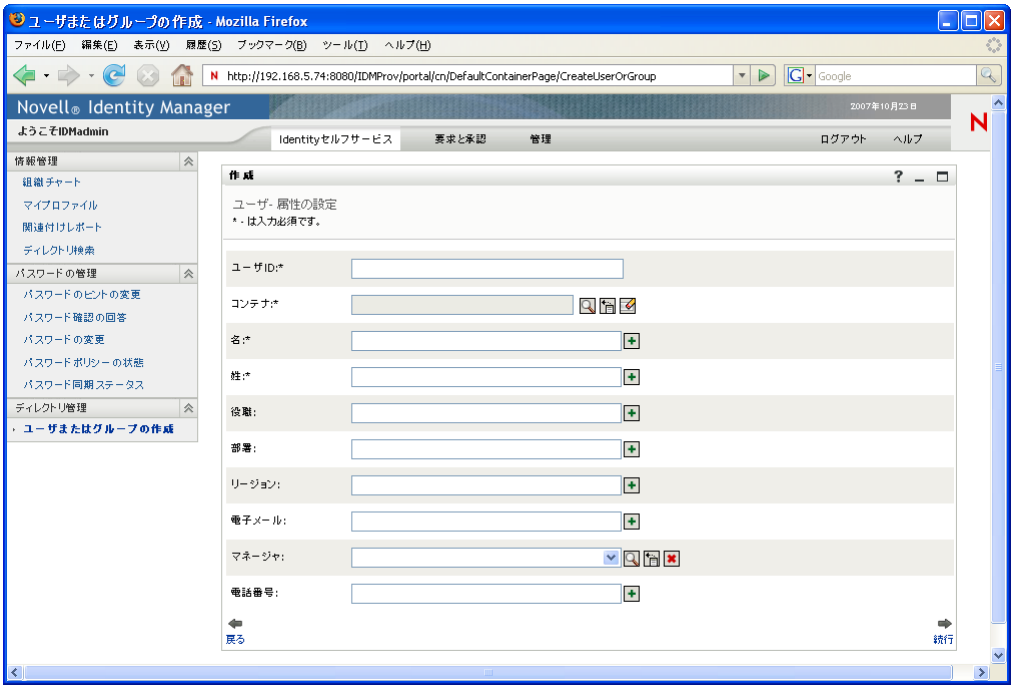
ゲストユーザが自己登録できるようにポートレットを設定することもできます。

作成ポートレットのデフォルト設定では、ユーザ、グループ、およびタスクグループを作成できるようになっています (作成ポートレットには、Identity Manager ユーザアプリケーションの *［ユーザまたはグループの作成］* アクションからアクセスできます)。デフォルトでは、このポートレットの操作はユーザアプリケーション管理者に限定されています。次の例は、デフォルトの［Create］ポートレットウィザードが \'95\'5c 示する内容を示します。

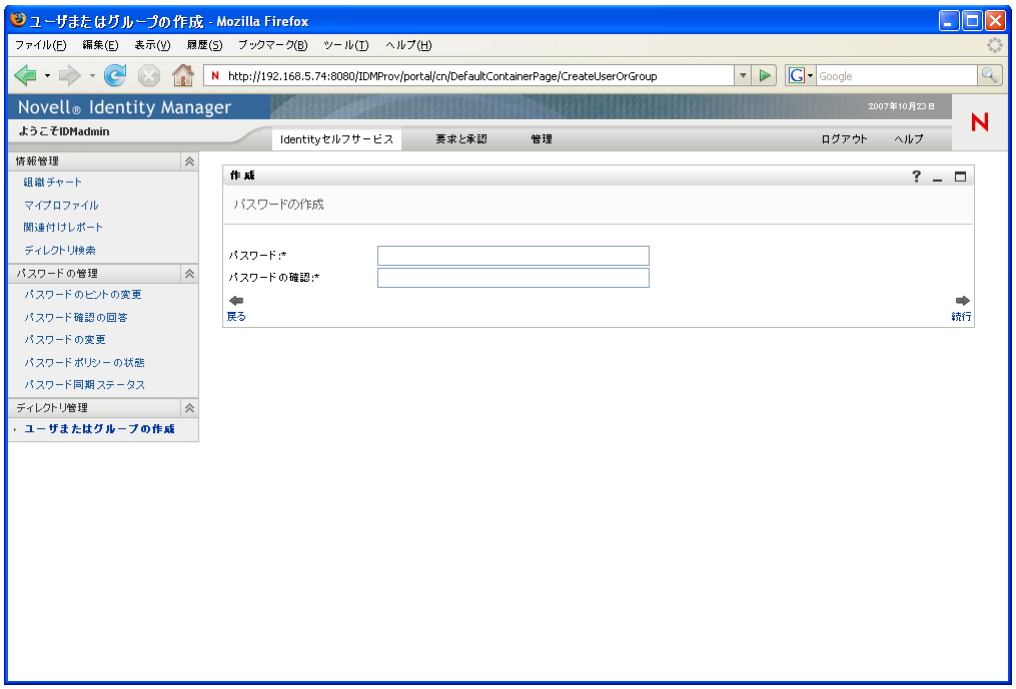
- 作成するオブジェクトのタイプを選択する



- オブジェクトの属性を入力する :



- オブジェクトタイプにより要求された場合にパスワードの入力を要求する



パスワードポリシーが割り当てられている場合、ポートレットにはカスタムポリシーメッセージが表示されます。

- オブジェクトを作成したら、参考情報となるメッセージを指定します。このメッセージには、そのオブジェクトをさらに編集するための、詳細ポートレットへのリンクを含めます (詳細ポートレットも同様に設定されている場合)。

11.2 ［Create］ポートレットの設定

作成ポートレットを環境設定するには、215 ページの 表 11-1 に説明されている手順に従ってください。

表 11-1 *作成ポートレットの環境設定手順*

手順	タスク	説明
1	デフォルトの「ユーザまたはグループの作成」機能が、要件に一致しているかどうかを判別します .	ニーズに合っている場合は、以降の作業は必要ありません。合わない場合は、残りの手順を完了してください。
2	ユーザに作成を許可するオブジェクトのタイプを定義します。	オブジェクトおよび属性をディレクトリ抽象化層に追加します。 詳細については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。

手順	タスク	説明
3	この新しいポートレットにユーザがアクセスする方法を指定します。	既存または新しいページから、ユーザがこのポートレットを起動できるようにしますか？ポートレットおよびページにアクセスできるのは、どのようなユーザとしますか？ ページの詳細については、139 ページの第 6 章「ページの管理」を参照してください。
4	ページおよびポートレットインスタンスにアクセスできるユーザを指定します。	ページセキュリティを編集し、ユーザをリストに追加します。ページへのユーザアクセスの制限の詳細については、139 ページの第 6 章「ページの管理」を参照してください。 ポートレットインスタンスを編集してセキュリティを変更します。ポートレットへのユーザアクセスの制限の詳細については、173 ページの第 7 章「ポートレットの管理」を参照してください。 匿名ユーザをこのポートレットにアクセスさせますか？匿名アクセス用の作成ポートレットの設定方法の詳細は、220 ページのセクション 11.4「作成ポートレットの自己登録の環境設定」を参照してください。
5	ポートレットの初期設定を指定します。	環境設定で定義する内容は、次のとおりです。 ◆ ユーザが作成できるオブジェクト ◆ 作成中に指定する属性 詳細については、218 ページのセクション 11.3「環境設定」を参照してください。
6	テスト .	オブジェクトが作成され、属性が適切に入力されていることを確認します。
7	ユーザに対する、eDirectory™ の有効な権利を設定します。	ユーザがオブジェクトを作成するために十分な権利を持っていることを確認してください。

11.2.1 ディレクトリ抽象化層の設定

作成ポートレットのユーザが作成可能なオブジェクト、および指定可能な属性は、ディレクトリ抽象化層で定義する必要があります。詳細は、217 ページの 表 11-2 を参照してください。

表 11-2 *ディレクトリ抽象化層の設定*

定義タイプ	プロパティ	値
entity	*作成*	選択済み .
	表示	選択済み . 選択しない場合、作成できるエンティティのリストにそのエンティティは表示されません。
	作成	*作成用コンテナ*: 有効なアイデンティティボールトコンテナを指定します。コンテナを割り当てない場合は、ユーザに対して選択を要求するメッセージが表示されます。ユーザは、ユーザアプリケーションのインストール時に指定されたルートコンテナから始まる、任意のコンテナを選択することができます。匿名ユーザの場合は、*作成用コンテナ*を指定することをお勧めします。指定しない場合は、［*ContainerLookupPortlet*］のセキュリティ設定を変更する必要があります。詳細は、220 ページのセクション 11.4「作成ポートレットの自己登録の環境設定」を参照してください。 *ネーミング属性の作成*: エンティティのネーミング属性を指定します。これは、作成ポートレットにオブジェクト ID として表示されます。*［名前付きラベルの作成］*を使って、別の文字列を指定できます。 **注 :** ネーミング属性はこのように定義されるため、ディレクトリ抽象化層に個別の属性として追加する必要はありません。
	パスワードの管理 : エンティティを作成するときにはパスワードが必要です	エンティティタイプが create のパスワードを必要とする場合、選択します。 ユーザを作成するように作成ポートレットが設定されており、そのユーザに対して iManager パスワードポリシーを割り当てる場合は、同じ iManager パスワードポリシーにこのコンテナも割り当てる必要があります。これにより、アプリケーション内で作成されたユーザは自動的にデフォルトの iManager パスワードポリシーが割り当てられます。 デフォルトでは、［ユーザまたはグループの作成］アクションにアクセスでき、OU に対するトラスティ権利がある任意のユーザが、ユーザを作成して初期パスワードを割り当てることができます。新しいユーザが初めてログインする際には、［パスワードの変更］ページが表示され、初期パスワードを変更することができます。初期設定 *［初回ログイン時にパスワードを有効期限切れにしますか］*を使って、デフォルトの動作を変更することができます。 この初期設定の詳細は、218 ページのセクション 11.3「環境設定」を参照してください。 ［パスワードの変更］ページの詳細は、120 ページのセクション 5.3.1「パスワード管理機能について」を参照してください。

定義タイプ	プロパティ	値
attribute	*対応*	選択済み .
	viewable	［enabled］または［viewable］を (誤って) 選択しない場合、ポートレットは属性を使用できません。

抽象化層を設定する場合の詳細については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。

11.3 環境設定

［初期設定］では、ユーザに表示するオブジェクトタイプと属性を設定できます。初期設定には、［一般］と［コンプレックス］の 2 種類があります。一般初期設定については 218 ページの 表 11-3 を、コンプレックス初期設定については 219 ページの 表 11-4 を参照してください。

表 11-3 *作成ポートレット : 一般初期設定*

初期設定	説明
詳細なポートレット名	オブジェクトを正しく作成した後に、ユーザが *［作成済みオブジェクト］* をクリックしたときに表示する、詳細ポートレットのインスタンスを指定します。デフォルトは標準の DetailPortlet です。詳細については、237 ページのセクション 12.6「詳細ポートレットの匿名アクセスの設定」を参照してください。
カスタムクラス名	作成イベントの処理に対するクラス名を指定します。デフォルトは、com.novell.srvprv.impl.portlet.create.CreateCustomEventDefaultHandler です。
初回ログイン時にパスワードを有効期限切れにしますか	初回ログイン時に新しく作成したユーザのパスワードを有効期限切れにするか (True)、またはアイデンティティボールトのパスワードポリシーの *GraceLogin* の設定をデフォルトにするかどうかを指定します。
パスワードを属性とともに表示しますか	パスワードを他の属性と同じページに表示するか (True)、または別のページに表示するか (False) を指定します。
Virtual Entity コンプレックス優先設定の作成	作成ポートレットのエンティティおよび属性定義にアクセスする場合は、*［カスタム初期設定の表示 / 編集］* をクリックします。初期設定の詳細は、219 ページの 表 11-4 を参照してください。

表 11-4 *作成ポートレット : コンプレックス初期設定*

初期設定	説明
エンティティ定義	作成するオブジェクトタイプの名前です。ここから、ポートレットのオブジェクト作成方法を定義するためのエンティティ定義ブロックが開始します。 初期設定内のオブジェクトは、ドロップダウンリストに表示されます。ユーザが作成できるオブジェクトを制限するには、［削除］ボタンを使用して、不要なオブジェクトをこの環境設定シートから削除します。他のエンティティを追加するには、*［エンティティ定義の追加］*をクリックしてウィザードを完了します。
Attributes	ユーザに入力を促す属性を指定します。オブジェクトに必要な属性はすべて含める必要があります。そうでない場合、実際のオブジェクトの作成が失敗します。また、必要な属性がない場合は、初期設定が適切には保存されません。 属性の追加または削除を実行する ◆ *［Modify Attributes (属性の変更)］*ボタンをクリックします。  ◆ 属性を追加するには、使用可能な属性のリストから対象の属性を選択します。<Ctrl> キーまたは <Shift> キーを使用して、複数の属性を選択できます。 ◆ 矢印をクリックして、属性を *［選択済み］*リストに移動します。属性を削除するには、逆の手順を実行します。 ◆ 属性リストを並べ替えるには、*［選択済み］***リストの右にある上下の矢印をクリックします。**［*送信*］をクリックします。 属性およびデータタイプ : 属性のデータタイプにより属性の表示方法が決まります。たとえば、属性がローカルまたはグローバルのリストサブタイプとして定義されている場合は、リストに表示されます。 **注 :** 作成ポートレットは、自動的にオブジェクト ID のプロンプトを表示します。(ラベルはエンティティタイプとして表示され、ユーザ ID やグループ ID などの文字列 ID を追加します) オブジェクト ID は、オブジェクトのネーミング属性です。属性として CN を追加する必要はありません。 詳細については、『*Novell Identity Manager 3.5 ユーザアプリケーション : 設計ガイド*』を参照してください。

初期設定パネルの設定の完了

有効なエントリを送信したことを確認するには、*［送信］*をクリックします。エントリが有効でない場合、初期設定ページの上部にエラーメッセージが表示されます。*［送信］*をクリックしても、エラーが発生しないようになったら、*［リストビューに戻る］*をクリックします。リストビューに戻った後は、*［設定の保存］*をクリックします。

11.4 作成ポートレットの自己登録の環境設定

ゲストユーザが自分で登録できるように、作成ポートレットを設定することができます。作成ポートレットへの匿名アクセスを有効にする作業は、2 段階に分かれています。まず、匿名アクセス用の作成ポートレットインスタンスを設定します。次に、新しいポートレットインスタンスのサービスを提供する共有ページを作成します。新しい登録ユーザにログインを強制したり、他のアイデンティティセルフサービス機能への匿名アクセスを許可することができます。ポートレットインスタンスを作成する

1. ［ポートレット管理］ページに移動します。
2. CreatePortlet の新しいインスタンスを登録し、名前を指定します (例 :*Self Registration*)。
3. 新しいポートレットインスタンスを選択して、*［設定］*をクリックします。
4. *［認証の要求］*に False を設定し、*［設定を保存］*をクリックします。
5. *［初期設定］*を選択して、初期設定を必要に応じて変更します。

   たとえば、匿名アクセスをサポートする DetailPortlet を指定したり、デフォルトのインスタンスが表示する属性セットを制限することができます。(デフォルトのインスタンスに対して行った変更内容は、そのインスタンスを使用するユーザアプリケーションの他の部分にも反映されます)

ヒント : デフォルトの DetailPortlet を指定した場合、ユーザが新しく作成されたオブジェクトの詳細を参照する際には、ログインが強制されます。詳細については、220 ページのセクション 11.4.1「ゲストアクセスに必要な設定」を参照してください。

共有ページを作成する

1. *［ページ管理］*タブに移動します。
2. 新しいページを作成します。
3. *［カテゴリの割り当て:］*で、*［ゲストページ］*を選択します。他のカテゴリを選択して、ログインしたユーザにそれを表示するように設定することもできます。
4. *［ページの保存］*をクリックします。
5. *［コンテンツの選択］*をクリックしてページにインスタンスを追加して、次に *［コンテンツの保存］*をクリックします。
6. *［許可の割り当て］*をクリックして、*［表示許可を管理者のみに設定］*の選択が解除されていることを確認します。
7. ページを保存します。

11.4.1 ゲストアクセスに必要な設定

他にも次のような設定が必要です。

- *コンテナの作成*: 各エンティティに作成コンテナが必要です。ディレクトリ抽象化層の置くエンティティタイプに対する、デフォルトの作成コンテナを定義したり、ユーザに作成コンテナを選択させることができます。各エンティティタイプに対するデフォルトの作成コンテナを指定した場合、ユーザにコンテナの選択を要求するメッセージが表示されることはありません。デフォルトを指定しない場合は、ユーザがコンテナを選択する必要があります。匿名ユーザの選択リストへのアクセスを許可するには、ContainerLookupPortlet の設定で *［認証の要求］*を False に設定する必要があり

ます。デフォルトの作成コンテナの詳細は、『*Identity Manager 3.5 ユーザアプリケーション: 設計ガイド*』のディレクトリ抽象化層について説明している項目を参照してください。

- *Identity Vault Rights(アイデンティティボールトの権限)*: 当初、ユーザはゲストユーザです。ユーザが自己登録する場合、オブジェクトは作成コンテナに書き込まれます。ユーザオブジェクトを作成するには、ゲストユーザは新しいユーザを作成するコンテナに対する [Entry rights] の作成権が必要です。この権利は、権利継承フィルタを使って継承したり、制限することができます。また、ゲストユーザには、作成を許可された属性に対する書き込み権も必要です。
- *DNLookup controls(DNLookup コントロール)*:DNLookupのコントロールタイプとして定義された属性の値をユーザが指定する必要がある場合、ParamlistPortlet の設定［*Requires authentication*(認証を要求する)］を False に設定する必要があります。
- *詳細ポートレット*: オブジェクトを正しく作成したら、ポートレットには表示されているオブジェクトへのリンクが表示されます (詳細ポートレット経由)。デフォルトの詳細ポートレットは認証が必要なため、ユーザは新しい ID 資格情報でログインしないと、詳細を表示することはできません。匿名ログイン用の詳細ポートレットインスタンスを別個に作成できます。また、デフォルトの詳細ポートレットを変更して、［Requires authentication(認証を要求する)］に False を設定することもできます。詳細については、237 ページのセクション 12.6「詳細ポートレットの匿名アクセスの設定」を参照してください。
- パスワード: 匿名ユーザにパスワードを要求するエンティティの作成を許可した場合は、匿名アカウントにパスワードの作成権を与えていることを確認してください。

［Detail］ポートレットの環境設定 12

この節では、エンティティの属性データの表示または操作が可能な、詳細ポートレットについて説明します。詳細ポートレットは、Identity Manager ユーザアプリケーションの *[Identity セルフサービス]* タブの *［マイプロファイル］* アクションの基本となります。主なトピックは次のとおりです。

12.1 ［Detail］ポートレットについて

詳細ポートレットはエンティティの属性およびその値を詳細に表示します。このポートレットには、表示と編集の 2 種類のモードがあります。［Detail］ポートレットにアクセスすると、組み込み機 \'94\'5c を使用して次のような操作をすることができます。

12.1.1 エンティティデータの表示

詳細ポートレットにアクセスすると、ユーザ、グループなど、選択したエンティティについての属性データが表示されます。たとえば、図 12-1 は Bill Brown が *［マイプロファイル］* アクションを選択した場合の詳細ポートレットの表示例です。

図 12-1 サンプルの MyProfile(マイプロファイル) データ

ユーザイメージ . デフォルトでは、詳細ポートレットには、ユーザの写真属性が含まれるようになっています。ただし、アイデンティティボールトにこの属性が含まれていない場合、または含まれてはいるが指定されていない場合、デフォルトイメージが表示されます。ユーザイメージを別の場所に格納する場合は、代わりにその場所からそれを表示するようにポートレットを設定できます。

詳細については、227 ページの 「イメージの動的なロード .」 を参照してください。

表示する属性の決定

詳細ポートレット (表示モード) には、次のような属性が表示されます。

- ディレクトリ抽象化層データ定義により、\'95\'5c 示可 \'94\'5c と設定されている .

  ディレクトリ抽象化層の環境設定の詳細については、16 ページのセクション 1.2.2 「ディレクトリ抽象化層」 を参照してください。
- *［ビューモードで表示する属性］* 初期設定に指定されています。

  ［Detail］ ポートレットが \'95\'5c 示する属性の指定については、234 ページのセクション 12.5 「環境設定」 を参照してください。
- 現在のユーザが、\'95\'5c 示のための権利を持っている .

  たとえば、属性を付与する権利を持つ \'83\'7d ネージャはデータを \'95\'5c 示できますが、他のユーザは \'95\'5c 示することができません。

  詳細については、233 ページのセクション 12.2.2 「エンティティへの権利の割り当て」 を参照してください。
- 現在、値が入力されている .

属性の表示方法の決定

属性を表示する場合、データはテキストとして表示されますが、一部例外があります。例外は、225 ページの 表 12-1 に記載されています。

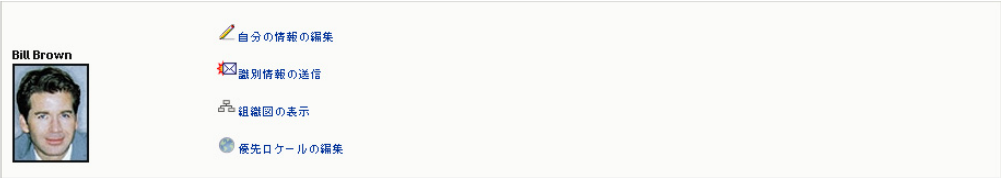
表 12-1 詳細ポートレット : テキストとして表示されない属性

抽象化層定義の形式の仕様	表示方法
Format: email	mail-to link として
書式 : ◆ groupwise-im ◆ aol-im ◆ yahoo-im	チャットの開始およびユーザの追加を行うアイコンとして
Data type: Binary Format: image	イメージとして
Data type: Boolean	［true］または［false］を示す、無効になっているラジオ \'83\'7b タンとして この \'83\'7b タンは、デフォルト値を示さないで \'95\'5c 示されます。値が指定されるまでは、属性が実際にユーザについて作成されないためです。
Multivalue: Selected	カンマ区切りリスト
Control type: DNLookup	リンクとして 前の例では、Terry Mellon というリンクが、Bill Brown のマネージャの詳細データへのアクセスを示しています。
Control type: ◆ Local List ◆ Global List	実際の (キー) 値ではなく \'95\'5c 示ラベルとして たとえば、［EmployeeType］属性は、実際の値 ft ではなく、Full Time が \'95\'5c 示されます。

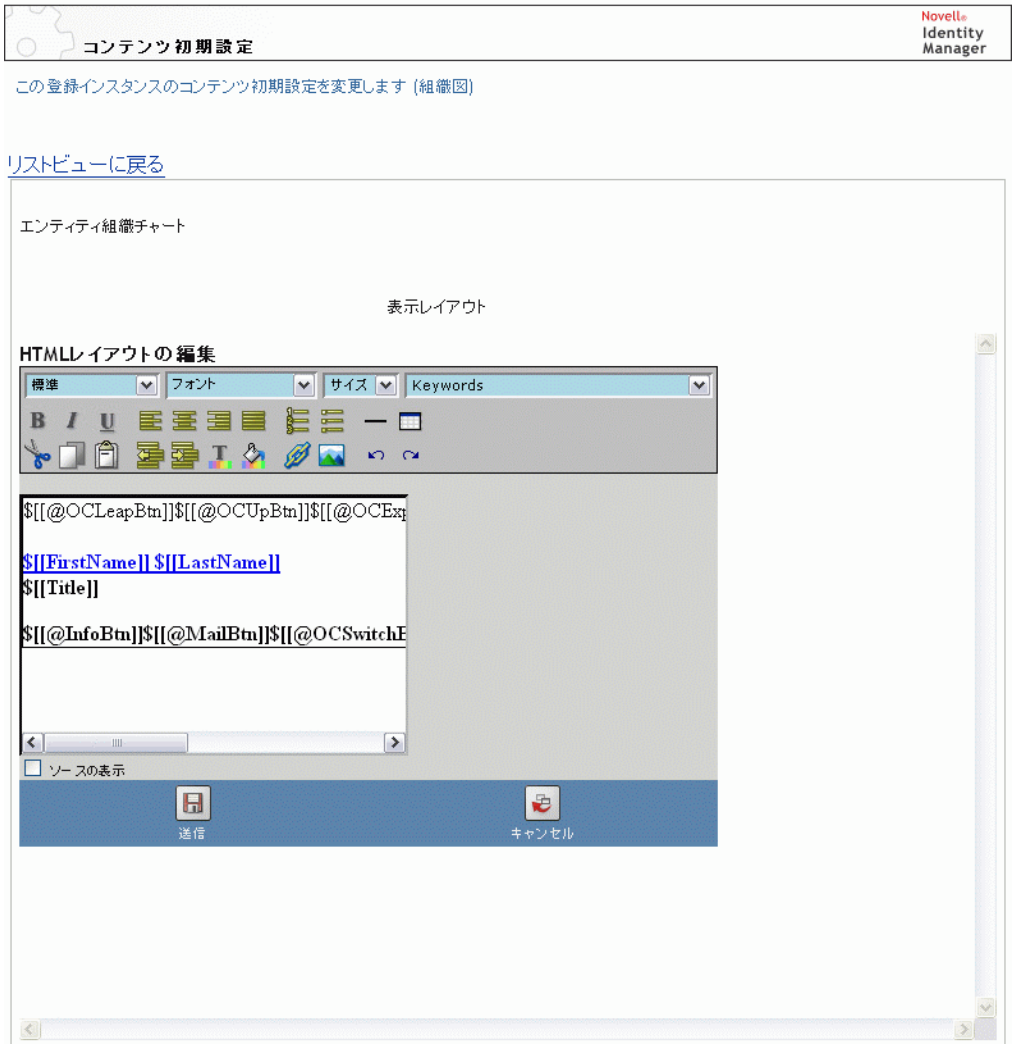
見出し領域の内容の決定

HTML の標準機能を使用して、詳細ポートレットの見出し領域のレイアウトを編集できます。

図 12-2 詳細ポートレット : 見出し領域



［Detail］環境設定には、\'95\'5c 示およびコンテンツの作成に使用する HTML Layout Editor が用意されています。



HTML Layout Editor の使用

HTML レイアウトエディタは、テキスト形式やリストを定義したり、アンカーやイメージなどを指定したりする HTML エディタの標準的な機能を提供します。

Keywords. レイアウトの設計時、*[キーワード]* ドロップダウンリストから、詳細ポートレットの見出し領域内に変数を挿入し、ランタイム時に特定の属性値で置き換えられるように設定できます。また、次の \'8d\'5c 文を使用して入力することもできます。

`$[[keyword]]`

keyword *は、LastName など、属性の値を \'95\'5c します。*

次の \'8d\'5c 文を使用すると、属性を連結できます。

`$[[keyword+keyword]]`

例 :

`$[[FirstName+LastName]]`

任意の数の属性を連結できます。また次のように、引用符で囲まれた文字列を含めることもできます。

```
$[[keyword+”sample text”+keyword]]
```

これにより、キーワードの値と引用符で囲まれた値がレンダリングされます。

注 : ドロップダウンリストから選択する代わりにキーワードのプレースホルダを手動入力する場合、HTML フォーマットが含まれていないことを確認してください。キーワードの手動入力時には、ソース表示モードを使用することをお勧めします。レイアウト中にキーワードの入力ミスがある場合、ランタイム時にはそのままの形でレンダリングされます ($[[]] を含む)。

イメージの動的なロード . アイデンティティボールトに保管されているイメージ (写真など) を表示するために、HTML レイアウトエディタを使って属性名を追加できます。たとえば、ユーザの写真を表示する場合は［ユーザの写真］属性を追加します。イメージをアイデンティティボールト外に格納している場合は、次のように、HTML エディタのソースの表示モードから IMG: タグを使用する必要があります。

1. ポートレットの環境設定に移動し、HTML Editor にアクセスします。
2. ［View Source*］をクリックします。*
3. 次のような \'8d\'5c 文で IMG: タグを使用し、場所、属性キー、ファイル拡張子を連結します。
   ```
   $[[IMG:”URL”+ attribute-key-name + ”fileextension”]]
   ```

次の例は、従業員の写真をアプリケーションサーバの /images サブディレクトリに Last Name (姓) ごとに JPG イメージとして格納している場合の構文です。

```
$[[IMG:"http://myhost:8080/images/"+LastName+".jpg"]]
```

ランタイム時、ポートレットは URL を LastName 属性およびファイル拡張子 .jpg と連結します。

HTML エディタは柔軟な構文をサポートしています。次の構文のようなテキストおよび属性の組み合わせをサポートしています。

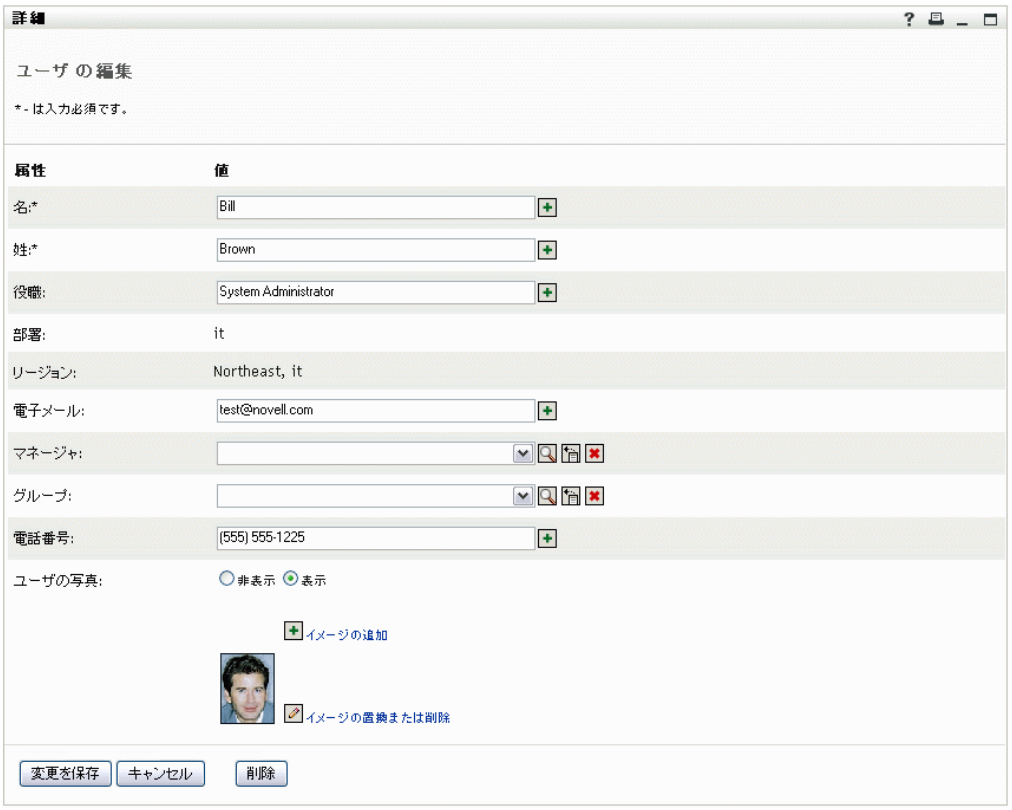
```
$[[IMG:”some text”+ attribute-key-name + ...]]
```

12.1.2 エンティティデータの編集

詳細ポートレットには、*編集*リンク (*［自分の情報の編集］*、*［ユーザの編集］* など) があり、表示モードから編集モードに切り替えられるようになっています。これにより、適切な権利を持つユーザは、現在のエンティティの属性を変更したり、変更を保存したりすることができます。

たとえば、(必要な権利を持っている)Bill Brown というユーザが自分自身の情報を編集するときに［Detail］ポートレットにより \'95\'5c 示される内容は次のとおりです。

図 12-3 MyProfile 編集モード



注 : ブール属性については、両方のラジオボタンがオフになっている場合、その属性が現在のユーザには存在しないことを示します。*True* または *False* を選択すると、ユーザの属性と値が作成、設定されます。

表示する属性の決定

編集モードでは、詳細ポートレットの初期設定 *[編集モードで表示する属性]* を使って、表示する属性や表示順序を指定できます。また、詳細ポートレットには次の属性のみを表示します。

- ディレクトリ抽象化層データ定義では、表示可能として定義されています。

  データ定義の詳細な説明については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。

- 現在のユーザが、\'95\'5c 示のための権利を持っている .

  たとえば、属性を付与する権利を持つ \'83\'7d ネージャはデータを \'95\'5c 示できますが、他のユーザは \'95\'5c 示することができません。

  詳細については、233 ページのセクション 12.2.2「エンティティへの権利の割り当て」を参照してください。

属性の表示方法の決定

編集モードでは、［Detail］は編集可 \'94\'5c な各属性をテキスト \'83\'7bックスに形式変換します。ただし次の場合を除きます。

表 12-2 *詳細ポートレット : 非テキストボックス編集可能属性の認識*

属性タイプの仕様 (ディレクトリ抽象化層で)	表示方法
Data type: Binary Format: image	イメージの \'95\'5c 示、更新、および追加を行うための［Entity Image Upload］ポートレットへの \'83\'7b タンおよびリンクとして
Data type: Boolean	［t rue］または［false］を示すラジオ \'83\'7b タンとして
hide: Selected	*［非表示］*と *［表示］*のラベルを持つラジオボタン
multivalue=Selected	属性の値の編集、追加、削除を制御するセットとして
Control type: DNLookup	DN の検索および選択のための［Param List］ポートレットを起動する \'83\'7b タンとして
Control type: ◆ Local list ◆ Global list	ドロップダウンリストとして (複数選択が可能)

定義により、またはユーザの権利が不十分なために編集できない属性は、*［無効］*または *［読み込み専用］*と表示されます。

変更の検証

編集時、次の属性タイプ指定についてはデータ検証が自動的に実行されます。

- Format: email
- Data type: Integer
- Control type: Range

ローカルリストまたはグローバルリストのコントロールタイプを使用する場合は、指定した属性の範囲外の値を表示リストに含めることができます。ただし、このような値は範囲外のフラグが設定され、検証により送信が中止されます。

12.1.3 エンティティデータの電子メールによる送信

詳細ポートレットには、*［識別情報の送信］*リンクがあります。このリンクをクリックすると、現在のエンティティの［詳細］画面の URL を、1 人または複数のユーザに電子メールで送信できます。実際の情報ではなく［詳細］画面の URL を電子メールで送信することにより、セキュリティを確保できます。これは、URL を受信したユーザがその URL を使用するには、適切な権限が必要になるためです。

12.1.4 組織チャートへのリンク

詳細ポートレットには、*[組織図の表示]* リンクがあります。このリンクをクリックすると、現在のエンティティの［Org Chart］ポートレットを \'95\'5c 示できます。

たとえば、Bill Brown というユーザの［Detail］が \'95\'5c 示されている場合、このリンクをクリックすると、次のように \'95\'5c 示されます。

図 12-4 *マイプロファイル : 組織図へのリンク*

組織図への自動リンクを抑制するには、詳細ポートレットの *[組織図の表示を有効にする]* に False を設定します。詳細については、234 ページのセクション 12.5「環境設定」を参照してください。

12.1.5 他のエンティティの詳細情報へのリンク

詳細ポートレットの設定時、ユーザが現在のエンティティから関連エンティティにリンクできるように設定しなければならない場合があります。そのためには、コントロールタイプ DNLookup(ディレクトリ抽象化層内) で定義された属性を入れます。

Manager 属性がユーザの［詳細］画面に表示される場合、この属性はリンクとして表示されます。このリンクをクリックすると、Manager の［Detail］が \'95\'5c 示されます。

図 12-5 マイプロファイルから他のエンティティへのリンク

ディレクトリ抽象化層の詳細については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。

［Detail］ポートレットが \'95\'5c 示する属性の指定については、234 ページのセクション 12.5「環境設定」を参照してください。

12.1.6 エンティティデータの印刷

デフォルトの詳細ポートレットの表示設定では、ポートレットのタイトルバー上の *［印刷］* オプションが有効になっています。*［印刷］* を有効にした場合、このオプションクリックすると、詳細コンテンツのプリンタフレンドリバージョンが表示されます。

詳細ポートレットのこれらの設定を変更するには、*［管理］* タブを使用して、*DetailPortlet* のポートレット登録を更新します (［ポートレット管理］ページ上で行います)。

詳細については、173 ページの第 7 章「ポートレットの管理」を参照してください。

12.1.7 優先ロケールの設定

詳細ポートレットには、*[優先ロケールの編集]* という名前のリンクが自動的に表示されます。これは、各自の情報を編集する管理者やユーザに対して表示されます。ユーザはこのリンクをクリックして設定項目を表示して、必要に応じて設定内容を変更することができます。優先ロケールを変更した場合、いったんログアウトしてから再度ログインしないと変更内容は反映されません。そうしないと、引き続き元の言語で表示されます。たとえば、Bill Brown というユーザの［詳細］画面が表示されている場合、このリンクをクリックすると、次の画面が表示されます。

図 12-6 *サンプルの使用ロケールの編集ダイアログ*

このリンクを表示しない場合は、*[優先ロケールの編集を有効にする]* に False を設定します。

12.2 前提条件

詳細ポートレットの使用を開始する前に、次の情報を確認してください。

- 233 ページのセクション 12.2.1「ディレクトリ抽出化層の設定」
- 233 ページのセクション 12.2.2「エンティティへの権利の割り当て」

12.2.1 ディレクトリ抽出化層の設定

詳細ポートレットは、ディレクトリ抽象化層の定義に依存します。特定の詳細ポートレット機能をサポートするための抽象化層データ定義の設定方法については、次の節を参照してください。

- 223 ページのセクション 12.1.1「エンティティデータの表示」
- 227 ページのセクション 12.1.2「エンティティデータの編集」
- 234 ページのセクション 12.4「ページからの詳細ポートレットの使用」

設定の詳細な説明については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。

12.2.2 エンティティへの権利の割り当て

詳細ポートレットのエンティティおよびその属性にアクセスするには、eDirectory™ の権利を適切に割り当てる必要があります。

操作	ユーザに必要な権利
属性の \'95\'5c 示	読み込み
属性の編集	書き込み

ユーザをオブジェクト (エンティティ) のトラスティに指定することで権利を割り当てることができます。また、詳細ポートレットで利用できる各属性に対して割り当てる権利を指定することもできます。

12.3 他のポートレットからの詳細ポートレットの起動

一般的に詳細ポートレットは、他の識別ポートレットからエンティティを選択した後に起動します。詳細ポートレットは、リスト検索ポートレットや組織図ポートレットから起動できます。

- 233 ページのセクション 12.3.1「リスト検索ポートレットからの詳細ポートレットの起動」
- 234 ページのセクション 12.3.2「組織図ポートレットからの起動」

12.3.1 リスト検索ポートレットからの詳細ポートレットの起動

リスト検索ポートレットで、検索結果内のエンティティ行をクリックすると、そのエンティティの［詳細］画面が表示されます。たとえば、次のリストの Bill Brown 行をクリックすると、詳細ポートレットが属性データとともに表示されます。

図 12-7 ディレクトリ検索からの詳細ポートレットの起動

［Search List］ポートレットの詳細については、269 ページの第 15 章「［Search List］ポートレットの環境設定」を参照してください。

12.3.2 組織図ポートレットからの起動

組織図ポートレットでは、エンティティの *［識別アクション］* アイコンをクリックして *［情報を表示］* を選択すると、そのエンティティの［詳細］画面が表示されます。

［Org Chart］ポートレットの詳細については、239 ページの第 13 章「［Org Chart］ポートレットの環境設定」を参照してください。

12.4 ページからの詳細ポートレットの使用

ユーザ自身の属性データを表示したり編集したりするためのセルフサービスをユーザに提供する場合、詳細ポートレットを共有ページに追加します。共有ページで詳細ポートレットを使用した場合、このポートレットは自動的に現在のユーザのデータにアクセスします。

12.5 環境設定

詳細ポートレットの内容と外観を定義するには、初期設定値を変更します。［Detail］ポートレットの使用方法は、環境設定の方法を決定します。

- 共有ページまたはコンテナページからのポートレット環境設定へのアクセスについては、139 ページの第 6 章「ページの管理」を参照してください。
- ポートレット登録のポートレット初期設定にアクセスする場合については、173 ページの第 7 章「ポートレットの管理」を参照してください。

12.5.1 初期設定について

詳細ポートレットには、一般設定用のページ (235 ページの 図 12-8 を参照) とコンプレックス設定用のページの 2 種類の初期設定用ページがあります。

図 12-8 詳細ポートレットの初期設定 : 一般設定

表 12-3 詳細ポートレット : 一般設定

初期設定	説明
組織図ポートレット名	［組織図の表示を有効にする］に True を設定した場合に起動する組織図ポートレットの、登録されたインスタンスの名前です。
エンティティ詳細のコンプレックス優先設定	詳細ポートレットのコンプレックス初期設定ページを表示するには、［カスタム初期設定の表示／編集］をクリックします。

この詳細環境設定を開くと、［Detail］の各環境設定は次のように \'95\'5c 示されます。

図 12-9 詳細ポートレット : コンプレックス設定

表 12-4 詳細ポートレット : コンプレックス設定

初期設定	詳細
エンティティ定義	(ユーザ、デバイス、グループなどの) 特定のエンティティタイプに［Detail］が使用された場合に \'95\'5c 示する、属性リストおよび HTML レイアウトを指定します。 追加エンティティタイプの［Detail］サポートを指定するには、［*Add Entity Definition*］をクリックします。
ビューモードで表示する属性	ビューモードの場合に、選択したエンティティについて、ポートレットに表示する属性を指定します。これらの属性は、選択した順にリストに表示されます。 \'83\'7b タンを使用すると、必要に応じて属性を追加または削除できます。
編集モードで表示する属性	編集モードの場合に、選択したエンティティについて、ポートレットに表示する属性を指定します。これらの属性は、選択した順にリストに表示されます。 \'83\'7b タンを使用すると、必要に応じて属性を追加または削除できます。

初期設定	詳細
HTML Layout	\'83\'7b タンを使用して［HTML Layout Editor］を開きます。ここでは、選択したエンティティについて、［Detail］ポートレットが \'95\'5c 示する見出しエリアを設計できます。 詳細については、225 ページの 「見出し領域の内容の決定」を参照してください。
エンティティの編集を有効にする	詳細ポートレットの見出しの *［自分の情報の編集］* リンクを有効にする場合、True を選択します。
エンティティ情報の送信を有効にする	詳細ポートレットの見出しの *［識別情報の送信］* リンクを有効にする場合、True を選択します。
組織図の表示を有効にする	詳細ポートレットの見出しの *［組織図の表示］* リンクを有効にする場合、True を選択します。
優先ロケールの編集を有効にする	詳細ポートレットの見出しに *［優先ロケールの編集］* リンクを表示する場合、True を選択します。

12.6 詳細ポートレットの匿名アクセスの設定

作成ポートレットの完了後、または検索の実行後、匿名ユーザが詳細ポートレットにアクセスする場合があります。匿名ユーザやゲストユーザ専用の、特別な詳細ポートレットのインスタンスを設定することができます。匿名アクセス用のインスタンスを別に設定しない場合、匿名ユーザには、アイデンティティボールトオブジェクトの詳細にアクセスするにはログインする必要があることを知らせるメッセージが表示されます。ゲストアクセス専用のインスタンスを設定する代わりに、標準の詳細ポートレットの認証要件を変更することもできます。

匿名アクセス用に詳細ポートレットを設定する

1. *［管理］*>*［ポートレット管理］* を選択します。
2. 詳細ポートレットの新しいインスタンスを登録し、名前を指定します (例 : 公開用詳細)。
3. 新しい詳細ポートレットのインスタンスを選択します。
4. *［設定］* に移動します。*［認証の要求］* に False を設定します。
5. ［*Save Settings*］*をクリックします。*
6. *［初期設定］* で初期設定を必要に応じて変更します。たとえば、ビューモードや編集モードに表示するエンティティや属性を変更します。

匿名ユーザに、ログインしない状態での詳細情報の表示を許可する場合、ユーザがログインしておらず、編集権もないため、詳細ポートレットに *［ユーザの編集］* や *［自分の情報の編集］* は表示されません。匿名ユーザにログインを強制する場合、eDirectory でコンテナ内の新規ユーザ用に設定されているポリシーセットにより、匿名ユーザの編集権が決められます。

［Org Chart］ポートレットの環境設定

13

この節では、Identity Manager ユーザアプリケーションを使った既存の組織図機能の変更方法、および新しい組織図機能の追加方法について説明します。主なトピックは次のとおりです。

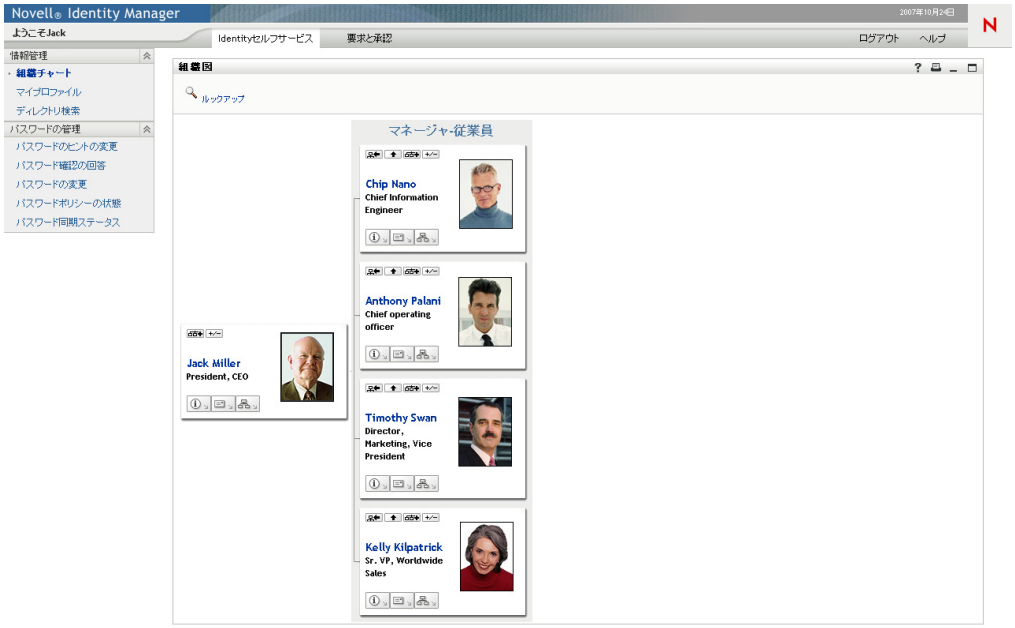
13.1 ［Org Chart］について

組織図ポートレットを使用すると、ユーザは、アイデンティティボールトのオブジェクト間の関係を表示したり、参照したりすることができます。たとえば、次のような関係を表す組織図ポートレットを定義できます。

- 組織 (従業員、\'83\'7d ネージャなど)
- グループのメンバーシップ (グループ内のすべての従業員など)
- ユーザに割り当てられたデバイス (携帯電話、ラップトップなど)

デフォルトでは、Identity Manage ユーザアプリケーションの *［Identity セルフサービス］* タブには *［組織図］* アクションが含まれています。このアクションは、アイデンティティボールト内のユーザオブジェクト間の関係を表示する組織図ポートレットです。次の例は、デフォルトの［Org Chart］ポートレットが (サンプルデータを使用して) この関係をレンダリングする方法を示します。

図 13-1 *デフォルトの組織図*



組み込みリンク . 組織図ポートレットには、次のリンクが組み込まれています。この組み込みリンクは、255 ページの 「組織図のレイアウトの初期設定」で説明している組織図のレイアウト初期設定で設定できます。

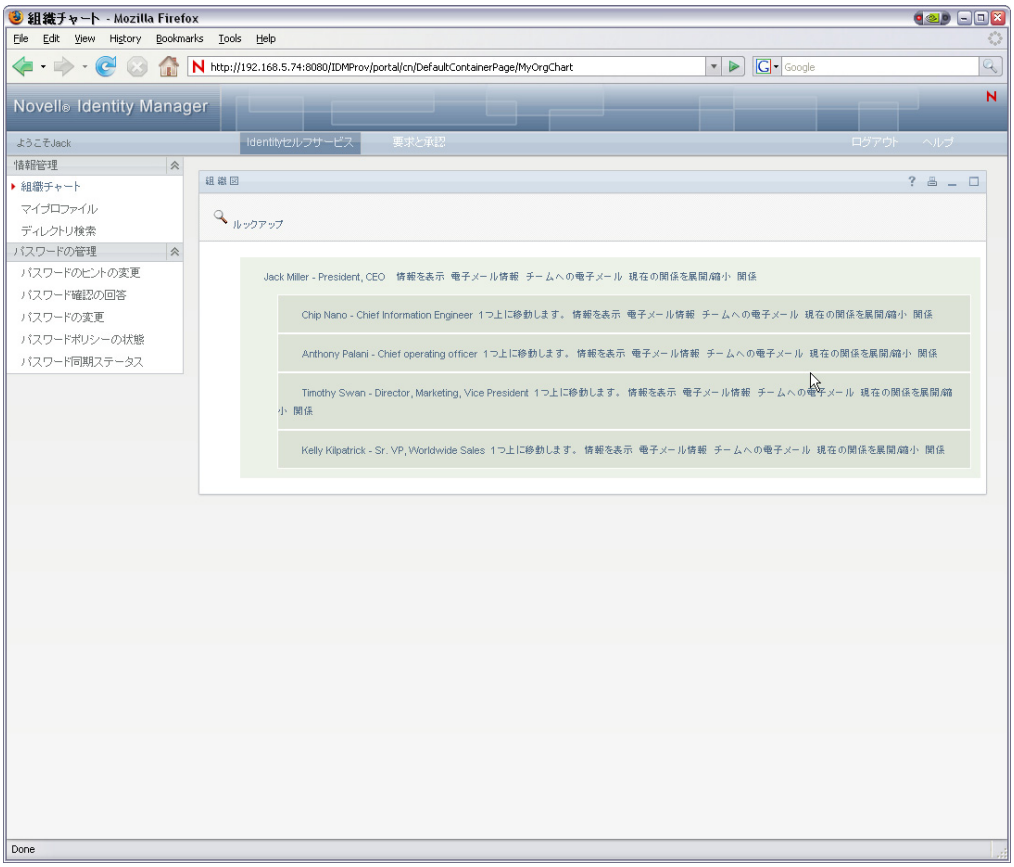
リンク	説明
	次の上位レベルに移動できます。これは、ターゲットエンティティとソースエンティティが同じタイプ (ユーザなど) の関係を表示している場合にのみ利用できます。関係は、ディレクトリ抽象化層エディタで定義します。
	デフォルトの関係を展開、縮小できます。 デフォルトの関係は、初期設定内に定義されます。これは、最初に表示される関係です。
	現在表示している組織図のルートをリセットできます。ルートは組織図の開始点です。
	ドロップダウン リストから、展開、縮小する関係を選択します。関係を展開した場合、どちらの方向 (左または右) に展開するかを選択できます。
	[Detail] ポートレットを起動します。
	組織図のリストを表示します。表示する 1 つまたは複数の組織図を選択できます。 この組織図のリストは動的なリストです。同じソースエンティティタイプを共有する他の組織図が表示されます。たとえば、マネージャの従業員組織図 (ソースエンティティはユーザ) を表示する場合、このアイコンをクリックすると、表示される組織図のリストには、ソースエンティティもユーザである関係のみが含まれます。 .

リンク	説明
	次の目的で電子メールツールを起動します。 ◆ 現在選択しているユーザの識別詳細を送信する . ◆ 電子メールを作成する。
	ユーザにエンティティ検索の実行を許可します。検索の結果、見つかったエンティティは、表示された図のトップノードとなります。(初期設定では環境設定できません)

組織チャートの組み込みリンクの追加および制限の詳細については、255 ページの 「組織図のレイアウトの初期設定」を参照してください。

組織図には、508 に準拠した形式で関係を表示することもできます。このビューをデフォルトで表示したり、オプションで表示するように初期設定を変更することができます。図 13-2 には、図 13-1 と同じ組織図データが表示されています。ただし、こちらは 508 に準拠した形式です。

図 13-2 *組織図アクセス可能ビュー*



13.1.1 ［Org Chart］の関係について

組織図ポートレットは、ディレクトリ抽象化層で定義されている関係を表示します。Identity Manager ユーザアプリケーションのインストール後に使用できる関係は、次のとおりです。

- グループのメンバーシップ
- \'83\'7d ネージャ - 従業員
- ユーザ - グループ

［Org Chart］関係の作成および変更については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。

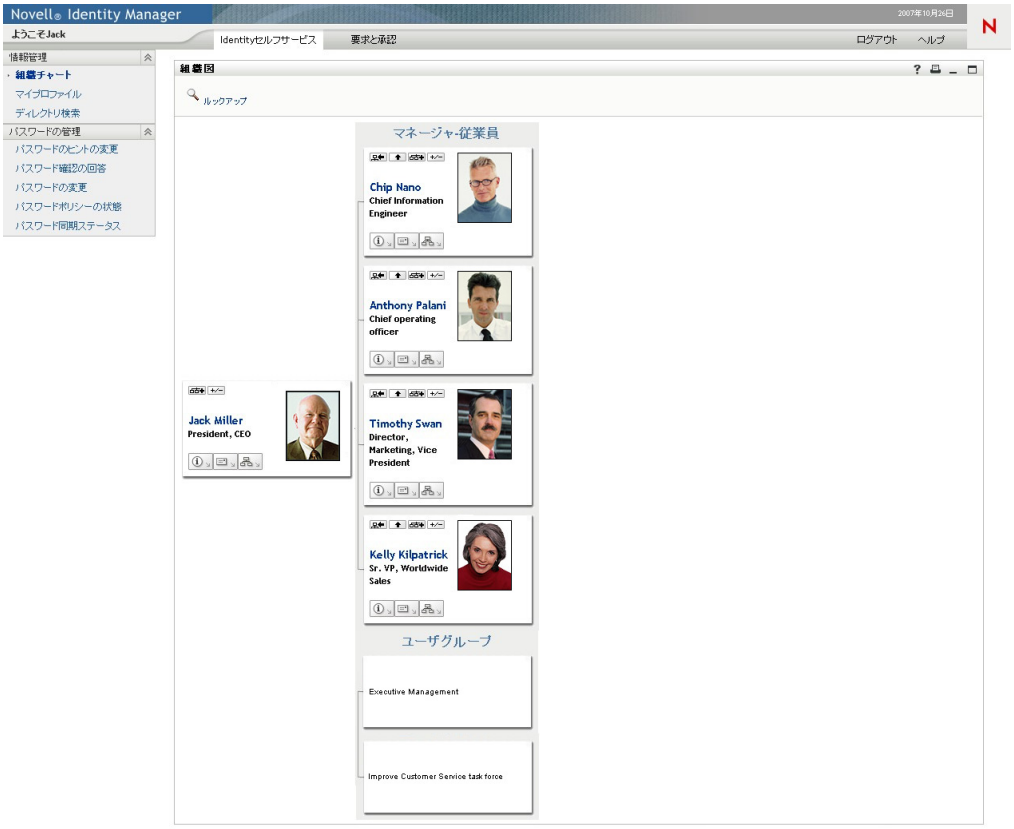
注 : 組織チャートポートレットでは、ダイナミックグループが完全にはサポートされていません。ダイナミックグループは、関係のソースエンティティとしては定義できませんが、ターゲットエンティティとしては定義できます。

13.1.2 組織図の表示について

組織図ポートレットでは、HTML モードのビューを表示したり (デフォルト)、508 に準拠した形式のアクセス可能モードのビューを表示することができます。これらのビューは、ポートレットの初期設定で有効／無効にできます。両方のモードを有効にすると、タブで各ページが表示されます。タブのに表示するタイトルも初期設定で変更できます。

複数の関係が同じソースエンティティを共有している限り、1 つの組織図に複数の関係を表示することができます。たとえば、図 13-3 の場合、ルートエンティティの「マネージャ - 従業員」と「ユーザ - グループ」の両方を組織図に表示しています。

図 13-3 複数の関係を表示した組織図



マネージャ属性が複数値の場合、どのマネージャの組織図を表示するかをユーザが選択できます。243 ページの 図 13-4 を参照してください。

図 13-4 複数値マネージャ属性の表示

ユーザイメージ

ユーザオブジェクトの組織図の HTML レイアウトには、デフォルトで「ユーザの写真」属性が含まれています。ただし、アイデンティティボールトにこの属性が含まれていない場合、または含まれていても指定されていない場合、ランタイム時に組織図で無視されま

す。フォトを別の場所に格納する場合は、それらのフォトを \'95\'5c 示するよう組織チャートを設定できます。

詳細については、264 ページのセクション 13.2.3「イメージの動的なロード」を参照してください。

13.2 組織図ポートレットの設定

組織図ポートレットを設定するには、表 13-1 の手順に従ってください。

表 13-1 *組織図 : 環境設定手順*

手順	タスク	説明
1	表示する関係を定義します。	Identity Manager ユーザアプリケーションのインストール時に事前定義された関係のいずれか 1 つを使用するか、独自に作成します。 関係の定義の詳細については、**16 ページのセクション 1.2.2「ディレクトリ抽象化層」**を参照してください。
2	関係で使用するエンティティおよび属性が、ディレクトリ抽象化層で使用できることを確認します。	関係の定義の詳細については、**245 ページのセクション 13.2.1「ディレクトリ抽象化層の設定」**を参照してください。
3	関係を表示する場所を決定します。	組織チャートの起動のための新しいページを作成しますか？［Detail］ポートレットまたは別の組織チャートから起動しますか？ ページの作成およびこれらのページへのポートレットの追加の詳細については、**139 ページの第 6 章「ページの管理」**を参照してください。
4	ポートレットの初期設定を指定します。	環境設定で定義する内容は、次のとおりです。 ◆ 表示する属性 . ◆ 表示方法 (HTML レイアウト). 詳細については、**245 ページのセクション 13.2.2「環境設定」**を参照してください。
5	テスト .	関係の定義およびレイアウトをテストします。
6	eDirectory™ の権利を設定し、パフォーマンス向上にインデックスが有効であれば、インデックスを構築します。	**有効な権利** : ポートレットにより定義された属性を表示するには、そのポートレットの読み込みの権利が必要です。 **パフォーマンスの向上** : 組織図表示のパフォーマンスを向上させるには、eDirectory の値インデックスを関係の子属性に追加します。子属性は LDAP 検索に使用されます。

13.2.1 ディレクトリ抽象化層の設定

組織図で表示するエンティティおよび属性は、ディレクトリ抽象化層で定義する必要があります。245 ページの 表 13-2 に、組織図で表示するそれぞれのエンティティおよび属性について、設定する必要がある属性およびプロパティを示します。

表 13-2 *組織図ポートレット : エンティティと属性の設定*

定義タイプ	設定	値
entity	表示	Selected(true)
attribute	読む	Selected(true)
	検索	Selected(true)

［ルックアップ］リンクの要件 *［ルックアップ］* リンクを使用して、ソースエンティティキーと同じタイプの他のオブジェクトの検索を実行することにより、組織図を操作できます。［ルックアップ］リンクを使用するには、ソースエンティティキーに、*require* アクセスプロパティと *search* アクセスプロパティに［true］が設定された属性が (ディレクトリ抽象化層エディタで選択) 少なくとも 1 つ必要です。そうしないと、ルックアップリンクの［オブジェクトルックアップ］ダイアログに表示できず、空になってしまいます。

エンティティおよび属性の設定の詳細な説明については、16 ページのセクション 1.2.2 「ディレクトリ抽象化層」を参照してください。

13.2.2 環境設定

関係、プレゼンテーション (属性やその並び順など)、および一般表示設定の初期設定値を定義できます。詳細については、以下の項を参照してください。

- 245 ページの 「組織図の一般初期設定」
- 252 ページの 「組織図のデータ／関係の初期設定」
- 255 ページの 「組織図のレイアウトの初期設定」

組織図の一般初期設定

このカテゴリには、メイン初期設定ページの初期設定が含まれますが、カスタム初期設定は除外しています。初期設定ページについては、図 13-5 と図 13-6 を参照してください。

図 13-5 組織図の初期設定

図 13-6 *組織図の初期設定 (続き)*

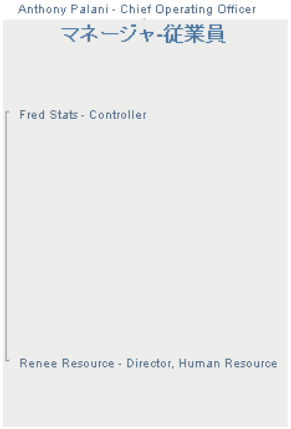
表 13-3 *組織図ポートレット : 初期設定*

初期設定	操作
データ	組織図の関係を定義する初期設定にアクセスするには、*［カスタム初期設定の表示 / 編集］* **をクリックします。詳細については、**252 ページの「組織図のデータ／関係の初期設定」を参照してください。
HTML ペインを有効にする	*True* を選択すると、関連するオブジェクトが HTML 表示されます。これがデフォルトの表示です。この場合、関連オブジェクトがビジネスカードとして表示されます。
HTML ペインタイトル	*［HTML ペイン］* **タブに表示する文字列を入力します。***［アクセス可能なペイン］* **と** *［HTML ペイン］* の表示を有効にした場合、ここに指定した文字列が HTML 表示のあるタブのタイトルとして表示されます。

初期設定	操作
アクセス可能なペインを有効にする	*True* を選択すると、関連するオブジェクトがアクセス可能ビューで表示されます。アクセス可能なペインには、オブジェクトとリンクが文字列として表示されます。このビューは、508 に準拠した形式で表示されます。
アクセス可能なペインタイトル	［アクセス可能なペイン］タブに表示する文字列を入力します。HTML ペインとアクセス可能なペインを有効にしている場合、ここに指定した文字列がアクセス可能ビューがあるタブのタイトルとして表示されます。
デフォルトのペイン	ユーザが *［組織図］* アクションをクリックした時に表示するデフォルトのペインを選択します。これは有効にする必要があります。
詳細なポートレット名	ユーザが *［情報を表示］* リンクをクリックした時に起動する、デフォルトのポートレットインスタンス名を指定します。
Presentation Layouts	*［カスタム初期設定の表示 / 編集］* をクリックしてレイアウト初期設定にアクセスします。を参照してください。255 ページの 「組織図のレイアウトの初期設定」
Maximum Depth	組織図でドリルダウンできる最大の深さを定義します。これは、有効な権利によって制限されている組織図内で移動できるかどうかを示すものではありません。
最大の初期階層	初期表示階層を定義します。
スクロールバーを表示	*True* を選択すると、スクロールバーが有効になります。

初期設定	操作
OrgChart Skin	次のいずれかの組織図用スキンを選択します。 *Business Card*:  *eGuide*:  *Novell.com*:  *Wired*:  *True Blue*: 
Connect wires to items	組織カードを回線接続するかどうかを指定します。［False］は接続しないことを示します。

初期設定	操作
Tree Presentation	組織図の向き (水平または垂直)、および図をビジネスカードで表示するか、またはテキストで表示するかを定義します。0 ～ 5 の値を指定できます。0、2、4 を指定すると、ビジネスカードで表示されます。1、3、5 を指定すると、テキストで表示されます。 ツリー表示値に 0、2、4 を指定すると、ビジネスカードで表示されます。 項目の垂直リストの上にカードを配置する場合は、*0* を指定します。  項目の水平リストの上にビジネスカードを配置する場合は、*2* を指定します。  項目の垂直リストの前にカードを配置する場合は、*4* を指定します。 

初期設定	操作
	ツリー表示値に 1、3、5 を指定すると、ラインを使って組織図が表示されます。 項目の垂直リストの上にラインを表示する場合は、*1* を指定します。  項目の水平リストの上にラインを表示する場合は、*3* を指定します。  項目の垂直リストの前にラインを表示する場合は、*5* を指定します。 

初期設定	操作
Leaf Presentation	許可されている最大の深さで、組織図のエンティティの外観 (向きと分布) を定義します。たとえば、最大の深さに 10 を指定した場合、リーフ表示では組織図の 10 番目のレベルのエンティティ表示が制御されます。最大の深さに 1 を指定した場合は、1 番目のレベルのエンティティのレイアウトが制御されます。 ツリー表示内の任意のリーフ表示を表示できます。
Minimum item width	ビジネスカード表示 (HTML モード) の最小の幅 (ピクセル) を指定します。この値は、ラウンドと等しくなります ('item min height' * 1.618)。
項目の最小の高さ	ビジネスカード表示の最小の高さ (ピクセル) を指定します。この値は、ラウンドと等しくなります ('item min width' * 1.618)。
Separator for multi-valued attributes	この文字は、複数の値を持つ属性の区切りとして使用します。

組織図のデータ／関係の初期設定

組織図の関係の初期設定にアクセスするには、*［データ］の［カスタム初期設定の表示／編集］*リンクをクリックします。初期設定ページを次に示します。このページには、デフォルトの組織図で使用されているデフォルトの関係が表示されます。

図 13-7 *組織図のデータ／関係の初期設定*

組織図で利用できるエンティティと関係を編集するには、編集ボタンをクリックします。参照先 (253 ページ) データ／関係の初期設定の編集 . 展開したノードの表示設定を変更するには、変更ボタンをクリックします。詳細については、(254 ページ) 展開ノードの変更を参照してください。

データ／関係の初期設定の編集

ここの初期設定項目は、ユーザが関係の展開／縮小ボタンをクリックした時に表示される、組織図と関係の初期表示に影響します。 . 任意の数の関係レベルを定義できます。

図 13-8 *デフォルトのデータ／関係初期設定の編集*

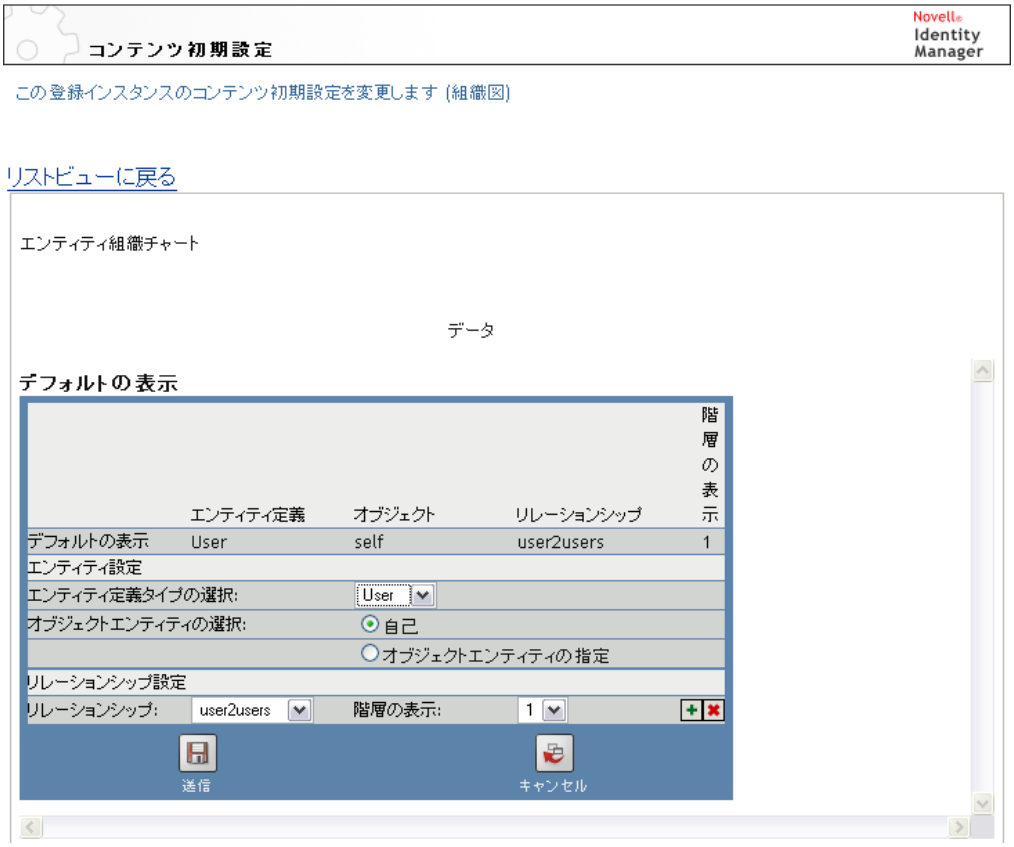


表 13-4 *組織図のデータ／関係の初期設定*

初期設定	説明
エンティティ設定	*[エンティティ定義タイプの選択]* では、関係を表示するエンティティを選択できます。このドロップダウンリストには、ディレクトリ抽象化層に定義されているエンティティのみが表示されます。 *[オブジェクトエンティティの選択]* では、組織図のルートエンティティを選択できます。オブジェクトのオブジェクトセレクタボタンをクリックしてください。選択したエンティティタイプ定義がユーザの場合、オブジェクトの代わりに［自己］を選択できます。［自己］を選択すると、組織図のルートがログオンしているユーザであることを表します。

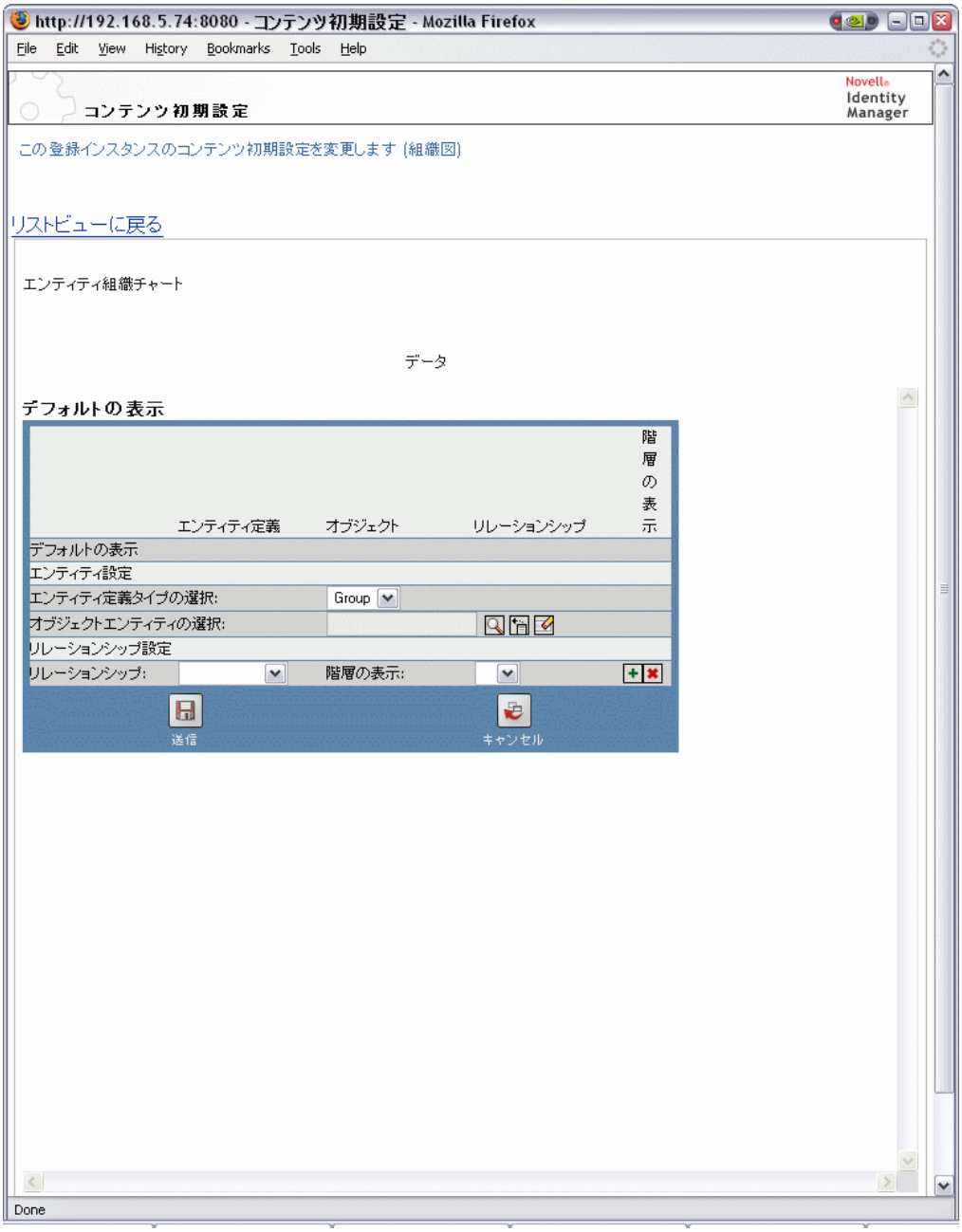
初期設定	説明
リレーションシップ設定	このカテゴリ内の設定項目では、デフォルトの組織図に表示する関係の詳細を指定できます。 *[リレーションシップ]* では、ドロップダウンリストから関係を選択できます。このリストには、選択したエンティティにとって意味のある関係のみが表示されます。 *[階層の表示]* には、表示する関係のレベル数を指定します。選択した関係で許可されている階層のみが表示されます。

拡張ノードの初期設定も同じですが、ユーザがが展開／縮小ボタン をクリックした後に表示される関係に関する設定項目であるということが異なっています。

展開ノードの変更

［展開ノード］では、ユーザが組織図の展開／縮小ボタン をクリックした場合に表示される項目を定義できます。

図 13-9 展開ノードの変更に関する初期設定



組織図のレイアウトの初期設定

*組織図のレイアウト*では、組織図エントリを表示するときの HTML レイアウトを定義できます。初期設定シートで利用できる HTML エディタを使用することも、好みの HTML エディタを使用することもできます。詳細については、264 ページの 「外部 HTML エディタの使用」を参照してください。

初期設定ページで利用できる HTML エディタは、組織図でリーフレイアウトを定義するための WYSIWYG インタフェースを提供しています。テキスト形式やリストを定義したり、アンカー、イメージなどを指定したりするための標準的な HTML エディタ機能を備

えています。属性、コマンド、ナビゲーション URL をレイアウト領域に配置するには、*[キーワード]* ドロップダウンリストを使用します。ドロップダウンリストからキーワードを選択すると、そのキーワードが適切な構文で挿入されます。レイアウト領域内に HTML を追加することもできます。

図 13-10 *組織図のレイアウトの初期設定*

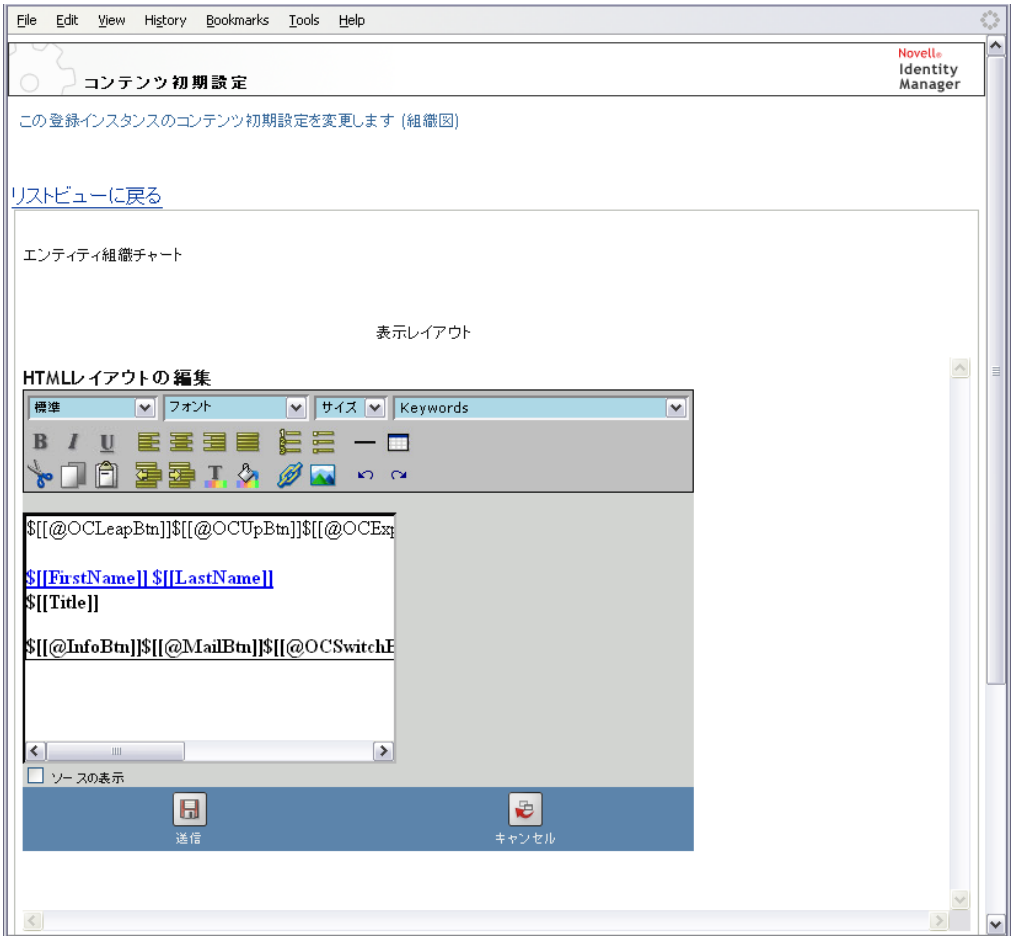
表 13-5 HTML レイアウト定義

レイアウトセクション	説明
ビジネスカードの HTML レイアウト	デフォルトのレイアウト。ツリー表示に 0、2、4 が設定された場合に表示されるレイアウトです。
セクション 508 ディスプレイのための HTML レイアウト	アクセス可能なペインのデフォルトのレイアウトです。
単純表示の HTML レイアウト	ツリー表示に 1、3、5 が設定された場合に表示されるレイアウトです。

HTML Editor の使用

[編集] ボタンをクリックすると、HTML エディタにアクセスできます。HTML エディタを図 13-11 に示します。

図 13-11 HTML Editor -



HTML エディタの機能とキーワード

HTML エディタの機能と *[キーワード]* ドロップダウンリストについては、表 13-6 を参照してください。レイアウトを保存するには、*[送信]* をクリックします。

表 13-6 *HTML エディタの機能*

機能	ヒント
［Insert Link］\'83\'7b タン	Mozilla の場合 : 1. リンクするテキストを選択して、*［リンクの挿入］*をクリックします。 2. URL を入力し、［***Create Link***］をクリックします。 3. 設定を保存します。 IE の場合 : 1. *［リンクの挿入］*をクリックします。 2. ポップアップウィンドウに、URL を入力します。 3. リンクするテキストを選択して、ポップアップウィンドウの *［リンクの作成］*をクリックします。 4. 設定を保存します。 **注 :** イメージまたは URL が HTML エディタの左上方角にある場合、その上にポップアップウィンドウが重なります。ポップアップウィンドウは移動できないため、その場合はエディタの任意の場所でテキストを作成してから正しい場所に切り貼りする必要があります。
［Add Image］\'83\'7b タン	Mozilla の場合 : 1. イメージを挿入する場所にカーソルを移動してから、*［イメージの追加］*> をクリックします。 2. URL およびテキストを入力してから、ポップアップウィンドウの *［イメージの作成］*をクリックします。 3. 設定を保存します。 IE の場合 : 1. ［***Add Image***］をクリックします。 2. ポップアップウィンドウに URL とテキストを入力し、イメージを挿入する場所にカーソルを移動してから、*［イメージの作成］*をクリックします。 3. 設定を保存します。 **注 :** イメージまたは URL が HTML エディタの左上方角にある場合、その上にポップアップウィンドウが重なります。ポップアップウィンドウは移動できないため、その場合はエディタの任意の場所でテキストを作成してから正しい場所に切り貼りする必要があります。

機能	ヒント

[キーワード] ドロップダウンリスト : 属性	このエンティティで利用できる属性セットです。レイアウトを設計する場合、[キーワード] ドロップダウンリストを使用して変数を挿入できます。この変数は、ランタイム時に、特定の属性値で置き換えられます。次の構文を使って、直接エディタに属性を入力することもできます。
	`$[[keyword]]`
	keyword は、LastName などエンティティ属性の値を表します。
	次の \'8d\'5c 文を使用すると、属性を連結できます。
	`$[[keyword+keyword]]`
	`$[[FirstName+LastName]]`
	たとえば、任意の数の属性を連結できます。また次のように、引用符で囲まれた文字列を含めることもできます。
	`$[[keyword+Åhsample textÅh+keyword]]`
	これにより、キーワードの値と引用符で囲まれた値がレンダリングされます。
	注 : レイアウトでキーワードを誤入力した場合は、それがそのまま ($[[]] を含む) レンダリングされます。

[キーワード] ドロップダウンリスト : コマンド	これらのコマンドを使って、240 ページの 「組み込みリンク .」に説明されているビルトインリンクに対するリンクやボタンを組織図ポートレットに表示できます。 キーワードコマンドは、次のものを生成します。 ◆ ナビゲーション URL。詳細については、260 ページの ï 13-7§ 「組織図キーワード : ビルトインのアクション URL」を参照してください。 ◆ アクションリンク。詳細については、262 ページの ï 13-8§ 「組織図キーワード : ビルトインのアクションリンク」を参照してください。 ◆ ナビゲーションボタン。263 ページの ï 13-9§ 「組織図ボタンのビルトインアクションボタン」。 HTML 表示のボタンを生成するコマンドセットや、アクセス可能なビューのリンクを生成するコマンドセットがあります。リンクには、リンク属性は表示されません。詳細については、262 ページの 表 13-8 を参照してください。

表 13-7 *組織図キーワード: ビルトインのアクション URL*

メニュー項目	ソース作成日時	使用率
組織図ナビゲーションのクリック > (リンク)	@OCNavClick	onClick イベントにはこのキーワードを使用します。これにより、クリックされたエンティティが新しい組織図ルートになります。 このキーワードを使用する 1. ［View Source］をクリックします。 2. 次の構文で、「@NavClick」というキーワードを入力します。 `<A href="javascript:return false;" onClick="$[[@NavClick]]">$[[SomeAttribute]]</A>` SomeAttribute *は、クリック可 \'94\'5c なリンクとなるエンティティ属性を \'95\'5c します。* 「javascript:return false」は必須です。省略すると、エラーが発生します。
組織図アップナビゲーション (リンク)	@OCUpClick	onClick イベントにはこのキーワードを使用します。現在のエンティティの親に移動します。複数の親が存在する場合、選択オプションがポップアップメニューに表示されます。 キーワードを使用するには、次の手順が必要です。 1. ［View Source］をクリックします。 2. 次の構文で、「@OCUpClick」を入力します。 `<A href="javascript:return false;" onClick="$[[@OCUpClick]]">$[[SomeAttribute]]</A>` SomeAttribute *は、クリック可 \'94\'5c なリンクとなるエンティティ属性を \'95\'5c します。* 「javascript:return false」は必須です。省略すると、エラーが発生します。

メニュー項目	ソース作成日時	使用率
	@OCExpCollClick	onClick イベントにはこのキーワードを使用します。クリックしたエンティティの既存の関係を展開／縮小できます。キーワードを使用するには、次の手順が必要です。 1. ［View Source］をクリックします。 2. 次の構文で、「@OCExpCollClick」を入力します。 `<A href="javascript:return false;" onClick="$[[@OCExpCollClick]]" >$[[SomeAttribute]]</A>` SomeAttribute は、クリック可 *\'94\'5c* なリンクとなるエンティティ属性を *\'95\'5c* します。 「javascript:return false」は必須です。省略すると、エラーが発生します。

メニュー項目	ソース作成日時	使用率

組織図ナビゲーションの URL(リンク)	@OCNavURL	リンクとして表示する URL またはエンティティ属性を指定します。クリックすると、そのエンティティが組織図にルートノードとして表示されます。これは、ソースエンティティとターゲットエンティティが同じオブジェクトの場合にのみ有効になります。たとえば、\'83\'7d ネージャ - 従業員の関係の場合、両者がユーザであることが必要です。
		
		このキーワードは、次のように使用します。
		
		1. ［View Source］をクリックします。
		2. 次の構文で、「@NavUrl」というキーワードを入力します。
		
		`<a href="javascript:$[[@NavUrl]]" >someText</a>`
		
		someText は、テキストまたはエンティティ属性です。次の例では、Click here が、クリックできるリンクとなります。
		
		`<a href="javascript:$[[@NavUrl]]" >Click here</a>`
		
		ここでは、FirstName 属性がクリックできるリンクとなります。
		
		`<a href="javascript:$[[@NavUrl]]" >$[[FirstName]]</a>`
		
		Internet Explorer の場合、次の構文は使用しないでください。Internet Explorer では @NavURL の前にコンテキストが追加されるため、正しく表示されません。
		
		`<a href="$[[@NavUrl]]">someText</a>`

表 13-8 にあるキーワードは、HTML ペインで使用するローカライズされたテキストリンクを生成します。

表 13-8 *組織図キーワード: ビルトインのアクションリンク*

メニュー項目	ソース作成日時	このテキストのローカライズされたリンクとしてレンダリング
現在の関係を展開 / 縮小 (リンク)	@OCLazyExpCollLink	*現在の関係を展開 / 縮小* 最初の再入可能な関係を検索して、それを縮小します。

メニュー項目	ソース作成日時	このテキストのローカライズされたリンクとしてレンダリング
組織図アップボタン (リンク)	@OCUpLink	*1 つ上に移動します* 現在のエンティティの親に移動します。複数の親がある場合は、親を選択するためのポップアップウィンドウが表示されます。
情報を表示 (リンク)	@ShowInfoLink	*情報を表示* 選択したエンティティの詳細ポートレットを起動します。
電子メール情報 (リンク)	@SendInfoLink	*電子メール情報* クリックしたエンティティの情報がある電子メールを起動します。
チームへの電子メール (リンク)	@MailTeamLink	*チームへの電子メール* 選択したエンティティのチームに送信する電子メールを起動します。

表 13-9 にあるキーワードは、HTML ペインで使用するイメージボタンを生成します。

表 13-9 *組織図ボタンのビルトインアクションボタン*

メニュー項目	構文	レンダリング内容
組織図リープ (アクションボタン)	@OCLeapBtn	 このボタンは、クリックされたエンティティを新しいルートにします。
組織図アップボタン (アクションボタン)	@OCUpButton	 現在のエンティティの親に移動します。複数の親がある場合は、親を選択するためのポップアップウィンドウが表示されます。
展開 / 縮小する関係を選択 (アクションボタン)	@OCExpColBtn	 クリックしたエンティティから、既存の関係を展開／縮小します。
現在の関係を展開 / 縮小 (アクションボタン)	@OCLazyExpColBtn	 最初の再入可能な関係を検索して、それを縮小します。

メニュー項目	構文	レンダリング内容
組織図 (アクションボタン)	@OCSwitchBtn	 クリックしたエンティティから、利用可能な関係を表示します。ユーザが関係を選択すると、クリックされたエンティティが新しいルートになり、選択した関係が展開されます。
情報 (アクションボタン)	@InfoBtn	 選択したエンティティの詳細ポートレットを表示します。
IM(アクションボタン)	@IMBtn	 ユーザにインスタントメッセージの送信と連絡先の追加を許可します。エンティティには、適切な属性を入れる必要があります。そうしないと、利用できるデータがないことを知らせるメッセージが表示されます。
メール (アクションボタン)	@MailBtn	 クリックしたエンティティの情報がある電子メールを起動します。

外部 HTML エディタの使用

外部 HTML エディタで作業を行うには、次の手順に従ってください。

1. 初期設定で使用できる *HTML レイアウトエディタ*を使用して、エンティティ属性、コマンド、キーワードの HTML ソースを作成します。
2. HTML ソースを、選択したエディタにコピーします。
3. 必要に応じて変更を行います。
4. 編集が終了したら、HTML ソースを HTML レイアウトエディタの初期設定にコピーします。

13.2.3 イメージの動的なロード

アイデンティティボールトに保管されているイメージ (写真など) を表示するために、属性名をビジネスカードに追加できます。たとえば、ユーザのフォトを \'95\'5c 示する場合は User Photo 属性を名刺レイアウトに追加します。

イメージをアイデンティティボールトの外部に格納している場合は、次のように、HTML エディタのソースの表示モードから IMG: タグを使用する必要があります。

1. ［Org Chart］ポートレットの環境設定に移動し、HTML Editor にアクセスします。
2. ［View Source］*をクリックします。*

3 次のような \'8d\'5c 文で IMG: タグを使用し、場所、属性キー、ファイル拡張子を連結します。

```
$[[IMG:”URL” + attribute-key-name + ”fileextension”]]
```

次の例は、従業員の写真をアプリケーションサーバの /images サブディレクトリに Last Name (姓) 別に JPG イメージとして格納している場合の構文です。

```
$[[IMG:"http://myhost:8080/images/"+LastName+".jpg"]]
```

ランタイム時、組織図ポートレットは URL を LastName 属性とファイル拡張子 .jpg に連結します。

HTML エディタは柔軟な構文をサポートしています。次の \'8d\'5c 文のような、テキストおよび属性の組み合わせをサポートしています。

```
$[[IMG:”some text” + attribute-key-name + ...]]
```

13.3 ゲストアクセス用の組織図の環境設定

ゲストアクセス用に組織図ポートレットを環境設定するには、組織図初期設定とユーザアプリケーション WAR ファイルを編集する必要があります。手順は以下を参照してください。

- 265 ページのセクション 13.3.1「組織図の初期設定の変更」
- 265 ページのセクション 13.3.2「ユーザアプリケーション WAR の変更」

13.3.1 組織図の初期設定の変更

1 *［管理］*>*［ポートレット管理］*を選択します。

2 OrgChartPortlet の新しいインスタンスを登録し、名前を指定します (例 : 公開用組織図)。

3 新しいインスタンスを選択し、［*設定*］タブに移動します。

4 *［認証の要求］*に False を設定して、*［設定を保存］*をクリックします。

5 *［初期設定］*タブで、初期設定を必要に応じて変更します。

6 この組織図のインスタンスを、匿名アクセス用に定義した作成ポートレットや詳細ポートレットから参照します。

13.3.2 ユーザアプリケーション WAR の変更

組織図ポートレットは、ユーザアプリケーション WAR の UIControlRegistry.xml ファイルに定義されているコントロールを使用します。デフォルトでは、これらのコントロールには認証が必要です。組織図ポートレットへのゲストアクセスを許可するには、WEB-INF\UIControlRegistry.xml ファイル内の portal サービスと vdm サービス定義で、認証要件に False を指定する必要があります。これを実働環境のユーザアプリケーションで使用する前に、まずテスト環境でこれらの指示に従って作業を行ってください。また、作業を開始する前に、ファイルをバックアップしてください。

portal サービスと vdm サービス定義の認証要件を変更する

1 ユーザアプリケーション WAR を表示して、コンテンツを抽出します。

2 WAR の WEB-INF ディレクトリにある UIControlRegistry.xml ファイルを探します。

3 UIControlRegistry.xml ファイルで、portal サービスのサービス定義を探します。これを次に示します。

```
<service resultType="json" authenticated="true" config="false">
  <key>portal</key>
<classname>com.novell.srvprv.impl.servlet.service.PortalBridge
   </classname></service>
```

4 *authenticated* の値を *false* に変更します。

5 UIControlRegistry.xml ファイルで、vdm サービス定義のサービスを探します。これを次に示します。

```
<service resultType="json" authenticated="false" config="false">
<key>vdm</
key><classname>com.novell.srvprv.impl.servlet.service.VDMBridge
   </classname></service>
```

6 *authenticated* の値を *false* に変更します。

7 変更を保存します。

8 ユーザアプリケーション WAR ファイルを再パッケージします。

9 更新した WAR をテスト環境に展開します。

リソース要求ポートレット 14

この節では、ユーザアプリケーションで使用するリソース要求ポートレットの設定およびカスタマイズの方法について説明します。ここは、次のトピックで構成されています。

14.1 リソース要求ポートレットについて

リソース要求ポートレットでは、ゲストや匿名ユーザがリソース要求を実行することができます。たとえば、ユーザが完了して承認されたワークフローを登録 (アイデンティティボールトに追加される) できるリソース要求を設定できます。

14.2 リソース要求ポートレットの環境設定

リソース要求ポートレットを環境設定するには、次の手順に従ってください。

表 14-1 *リソース要求の環境設定手順*

手順	タスク	説明
1	システムの匿名ユーザを定義します。	LDAP ゲストを使用していますか？それとも、アイデンティティボールトに特別に定義されたユーザを使用していますか？ワークフローを実行するためにこのユーザに必要な権限は？ワークフローに正しいプロパティ属性セットが定義されていますか？ 匿名ユーザの詳細は、**13 ページの第 1 章「ユーザアプリケーションの紹介」**を参照してください。
2	このポートレットで実行するリソース要求を指定します。	詳細については、**268 ページのセクション 14.2.1「環境設定」**を参照してください。
3	新しいリソース要求を入れるために、新しいページを作成します。このページのセキュリティでは、ゲスト／匿名アクセスを許可する必要があります。	詳細については、**154 ページのセクション 6.3「共有ページの作成とメンテナンス」**を参照してください。 新しい共有ページを作成したら、ゲストカテゴリを指定し、ページの *[表示許可を管理者のみに設定]* の選択を解除していることを確認してください。
4	匿名ユーザとして、リソース要求をテストします。	ワークフローが期待通りに完了することを確認してください。

ヒント : リソース要求ポートレットで使用するワークフローを作成し、電子メール通知で To トークンを _default_ として定義した場合、宛先の式は ID ボールトの式でなければなりません。

14.2.1 環境設定

環境設定には、次のものが含まれます。

表 14-2 *リソース要求ポートレット : 一般設定とカスタム設定*

初期設定	説明
リソース要求	ページに追加するリソース要求のリストを表示するには、*［カスタム初期設定の表示 / 編集］*をクリックします。このリストには、ユーザアプリケーションドライバに展開されたリソース要求が表示されます。 1 つのリソース要求を選択します。このリストには、ユーザアプリケーションドライバに展開されたリソース要求が表示されます。

15 ［Search List］ポートレットの環境設定

この節では、Identity Manager ユーザアプリケーションで使用するリスト検索ポートレットの設定およびカスタマイズの方法について説明します。主なトピックは次のとおりです。

15.1 ［Search List］ポートレットについて

リスト検索ポートレットを使用して、アイデンティティボールトのコンテンツを検索したり表示したりすることができます。これは、Identity Manager ユーザアプリケーションの *[Identity セルフサービス]* タブのディレクトリ検索アクションの基本となります。ディレクトリ検索アクションは、ユーザやグループの検索用に設定されます。ディレクトリ検索アクションを変更して、検索可能なオブジェクトおよび属性の範囲を変更することもできます。

ディレクトリ検索アクションがどのようにユーザによる検索条件の指定を許可するかを、269 ページの 図 15-1 に示します。

図 15-1 *基本検索*

表 15-1 *ディレクトリ検索条件*

ユーザインタフェース要素	説明
検索対象	検索するオブジェクトのタイプを選択します。 このリストのコンテンツの定義の詳細な説明については、275 ページのセクション 15.2.2「［Search List］の環境設定」を参照してください。

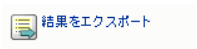
ユーザインタフェース要素	説明
With this criteria	ドロップダウンから属性および検索演算子を選択することにより、検索条件を定義します。 ［*高度な検索*］を選択した場合、複数の行と複数のブロックを、包含的 (AND) または排他的 (OR) のいずれかの検索条件グルーピングとして指定できます。 検索可 \'94\'5c な属性の定義の詳細については、275 ページの「［Search List］の環境設定」を参照してください。
検索	指定した検索条件を実行します。 デフォルト検索の定義の詳細については、275 ページの 「［Search List］の環境設定」を参照してください。
My Saved Searches マイ保存済み検索	以前に保存した検索の実行、編集、または削除ができます。
Advanced Search 高度な検索	行またはブロックを検索条件として追加できます。詳細検索では、複数の行および複数のブロックを、包含的 (AND) または排他的 (OR) のいずれかの検索条件グルーピングとして指定できます。 検索可 \'94\'5c な属性の定義の詳細については、275 ページの「［Search List］の環境設定」を参照してください。

次の例は、検索条件［First name starts with A］が指定された後の、ポートレットの \'95\'5c 示方法を示します (サンプルデータを使用)。

図 15-2 *サンプルのリスト検索結果*

271 ページの 表 15-2 に記載されている機能を使用できるように、リスト検索ポートレットを設定することもできます。

表 15-2　リスト検索ポートレットの機能

ユーザインタフェース要素	説明
［識別子］、［位置］、［組織］タブ	これらのタブのいずれか 1 つをクリックすると、リストがそれぞれ別の方法で \'95\'5c 示されます。 形式の詳細な説明については、271 ページの 「結果リストの表示形式について」を参照してください。
My Saved Searches 	以前に保存した検索を選択できます。
検索の保存 	検索条件を保存し、必要に応じて保存した条件を再実行できます。検索は、現在ログオンしているユーザの srvprvQueryList 属性に保存されます。
結果のエクスポート 	検索結果を異なる形式にエクスポートできます。
検索の修正 	検索条件を変更できます。
New Search 	新しい検索を定義することができます。

デフォルトでユーザがリスト検索で実行できる操作を次に示します。

- 検索結果のプリント
- 結果リストからの電子メールの起動
- 結果リストからの［Detail］ポートレットの起動

15.1.1 結果リストの表示形式について

アイデンティティボールトから返されるデータの表示方法を定義できます。データは、次のページタイプの中の 1 つまたは複数の方法で編成できます。

- 通常識別ページには、連絡先情報が含まれます。次に例を示します。

- 通常位置ページには、位置情報が含まれます。次に例を示します。

- 通常組織ページには、組織階層情報が含まれます。次に例を示します。

ポートレットの詳細環境設定を使用すると、他の結果リスト形式を定義できます。たとえば、アイデンティティボールトスキーマに従業員のスキルに関する情報が含まれている場合、結果リストにこの情報を表示するよう設定できます。

ポートレットの設定内容に応じて、エンドユーザが実行できる操作を次に示します。

- 検索するアイデンティティボールトブジェクトのタイプを選択する (ユーザ、グループなど)
- 検索する条件を指定する (「名が指定した文字で開始」、「指定した文字を含む姓」など)
- 検索結果を \'95\'5c 示する \'95\'5c 示形式を選択する
- \'83\'5cート順を変更する

15.2 ［Search List］ポートレットの設定

リスト検索ポートレットを環境設定するには、表 15-3 の手順に従ってください。

表 15-3 *リスト検索ポートレットの環境設定手順*

手順	タスク	説明
1	定義 : • ユーザに検索を許可するエンティティおよび属性です。 • 結果リストの表示方法 .	Identity Manager ユーザアプリケーションのインストール時に同時にインストールされる事前定義されたディレクトリ検索アクションをそのまま使用できます。アクションは、変更することも、新たに独自に作成することもできます。 詳細については、275 ページの 「［Search List］の環境設定」を参照してください。
2	検索のためのエンティティおよび属性のセットが、ディレクトリ抽象化層で定義されていることを確認します。	詳細については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。

手順	タスク	説明
3	ポートレットにユーザがアクセスする方法を指定します。	既存または新しいページから、ユーザがこのポートレットを起動できるようにしますか？ ページの詳細については、139 ページの第 6 章「ページの管理」を参照してください。
4	ポートレットの初期設定を指定します。	［Search List］ポートレットの環境設定で定義する内容は、次のとおりです。 ◆ 結果リストの各形式で \'95\'5c 示する属性 . ◆ 検索により生成される、結果リスト表示形式です。 ◆ 結果リスト形式のデフォルトの \'83\'5cート順 . 詳細については、275 ページのセクション 15.2.2「［Search List］の環境設定」を参照してください。
5	設定をテストします。	結果リストに、対象の属性が \'95\'5c 示されていることを検証します。
6	eDirectory™ の権利を設定し、パフォーマンス向上にインデックスが有効であれば、インデックスを構築します。	eDirectory の権利 : 検索を実行する : ◆ 検索を実行するユーザには、検索対象となるユーザまたはオブジェクトへのブラウズ権が必要です。 検索を保存する (管理者以外のユーザの場合): ◆ *検索を実行する部門または組織*のトラスティ ◆ *ユーザ*には、書き込み権、自己権、およびスーパバイザ権が必要です。 **パフォーマンスの向上 :** 検索のパフォーマンスを向上させるには、eDirectory の値インデックスを、検索で基になる属性に追加します。

さまざまな結果リスト形式の定義の詳細については、275 ページのセクション 15.2.2「［Search List］の環境設定」を参照してください。

15.2.1 ディレクトリ抽象化層の設定

検索条件ドロップダウンリストから選択するエンティティおよび属性、およびアイデンティティボールト検索から返されるデータは、ディレクトリ抽象化層で定義されている必要があります。表 15-4 に、リスト検索で使用するエンティティおよび属性について設定の必要があるプロパティを示します。

表 15-4 *リスト検索のエンティティと属性*

定義タイプ	設定	ディレクトリ抽象化層の値
entity	表示	Selected(true)

定義タイプ	設定	ディレクトリ抽象化層の値
attribute	有効	Selected(true).
	検索	Selected(true). 利用可能な検索条件のリストに表示する属性は、search=true でなければなりません。「false」の場合、この属性の検索を定義できないか、この属性を結果リスト形式に含めることができません。
	hide	Unselected(false). 結果リストに入れる属性は、hide=false でなければなりません。

ディレクトリ抽象化層のその他の設定 . ディレクトリ抽象化層のデータタイプ、形式タイプ、フィルタ、および検索範囲も、リスト検索ポートレットに影響を与えます。データタイプおよび形式タイプは外観に影響を与え、フィルタおよび検索範囲は、データが返される方法に影響を与えます。

詳細については、『*Identity Manager ユーザアプリケーション: 設計ガイド*』を参照してください。

15.2.2 ［Search List］の環境設定

次の 2 種類の初期設定を定義できます。

- 275 ページの 「検索の環境設定」
- 277 ページの 「結果リストの形式の環境設定」

検索の環境設定

検索の環境設定は、1 つの環境設定ページに含まれています。

検索の初期設定は、277 ページの 表 15-5 に定義されています。

表 15-5 *リスト検索ポートレットの初期設定*

初期設定	操作

Default Mode	ユーザが最初にアクセスしたときのポートレットの表示方法を指定します。次の値があります。
	
	基本検索:1 つの検索条件を入力できます。例 :
	`First Name starts with A`
	
	高度な検索:1 つまたは複数の検索ブロックに、複数の検索条件を定義できます。検索条件または検索ブロックで、論理演算子 and や or を使用できます。たとえば、次のような検索を作成できます。
	`(First Name starts with A or First Name starts with B) and (Region = Northeast or Region = Southeast)`
	
	OR
	`(First Name starts with A and Last Name starts with B) or (First Name starts with B and Last Name starts with A)`
	
	マイ保存済み検索: 現在ログインしているユーザによって保存された検索のリストを表示します。検索は、ユーザの srvprvQueryList 属性に保存されます。
	
	注 : ランタイム時に、検索の実行または編集を行うか、ポートレットの下部にある \'83\'7b タンをクリックすると、これらのモードのいずれかにアクセスします。

Pagination	一度に \'95\'5c 示できる最大行数を指定します。
Results Limit	検索によって返される最大一致件数を指定します。0 に設定している場合は、ディレクトリ抽象化層の設定に従います。
Search and List complex preference	次の条件で絞り込む場合にクリックします。 ◆ 検索するエンティティ ◆ 結果セットタイプ ◆ ページに含める属性および \'95\'5c 示する順序

結果リストの形式の環境設定

複合初期設定ページで、検索に含めるエンティティ、および結果リスト形式を定義できます。デフォルトの環境設定ページは、このようなものです。

複合初期設定を、279 ページの 表 15-6 に示します。

表 15-6 *リスト検索ポートレット : 複合初期設定*

初期設定	操作
エンティティ定義	検索に有効なオブジェクト (view=true) にはそれぞれ、対応する［エンティティ定義］ブロックがあります。これらの環境設定は、次の目的に使用します。 ◆ 検索に含めるオブジェクトを定義します。 ◆ 結果リストの形式の定義を変更します (\'95\'5c 示する属性の追加および削除、およびデフォルトの \'83\'5c一ト順など)。 ◆ 検索に含めないオブジェクトを削除するには、［エンティティ定義］行の *［削除］* をクリックします。すると、［Entity Definition］ブロック全体が削除されます。 オブジェクトを後で再び \'95\'5c 示するには、［*Add Entity Definition*］(ページ下部) をクリックし、ウィザードの選択パネルの指示に従います。 **ヒント :** あるオブジェクトがこのリストに表示されず、ディレクト抽象化層のリストには表示されている場合、そのエンティティオブジェクトの「*view*」修飾を確認します。false に設定されている場合、識別ポートレットはそのエンティティを使用できません。
Show email as Icon	*True* を設定し、結果リストに電子メール属性が指定されている場合、それがアイコンで表示されます。*False* を設定すると、電子メール属性が完全な電子メールアドレスで表示されます。電子メール属性は (テキストの場合もアイコンの場合も)、クリック可能な *mailto:* リンクです。
Results List Types (default)	現在のエンティティについて、結果リストのデフォルト形式を指定します。デフォルトは、現在のユーザが別の形式を選択しない場合にのみ使用されます。
Results List display format block	\'95\'5c 示形式 (［Identity］、［Location］、［Organizational］の各ページなど) を指定し、タイプに含める属性のセットを含みます。 ［Results List Type］を削除する ◆ *結果リストタイプ*の横にある *［削除］* をクリックします。 これにより、ページタイプおよびすべての関連属性が検索から削除されます。 結果セットページを追加する ◆ *［展開］* ボタンをクリックし、結果セット形式を選択リストから選択します。

初期設定	操作
属性	特定の \'95\'5c 示形式で \'95\'5c 示する属性のセットを指定します。 属性の追加または削除を実行する ◆ *[Modify Attributes (属性の変更)]* ボタンをクリックします。 ◆ 属性を追加をするには、Available 属性のリストから) 対象の属性を選択します。 ◆ 矢印をクリックして、属性を［選択済み］リストに移動します。属性を結果リストから削除するには、逆の手順を実行します。 ◆ 属性リストを並べ替えるには、［Selected］リストの右にある上下の矢印をクリックします。 ◆ ［*送信*］をクリックします。 **属性およびデータタイプ** : 属性のデータタイプは、表示方法に影響を与えます。たとえば、ローカルリストまたはグローバルリストのサブタイプとして属性が定義されている場合、指定できる値は、基本検索または高度な検索の条件画面のドロップダウンリストボックスに表示されます。タイプが DN である場合、基本検索または高度な検索の条件画面でユーザが値を選択できるように、［finder and history (検索および履歴)］ボタンが表示され、DN は、結果リストの形式でユーザにわかりやすいように表示されます。データのタイプおよびサブタイプは、有効な比較のみが作成されるよう、\'95\'5c 示する比較演算子も制限します。 詳細については、16 ページのセクション 1.2.2「ディレクトリ抽象化層」を参照してください。
Results List display format block Sort	この属性に基づく結果リストのソート順を指定します。デフォルトの \'83\'5c一ト順は、［Result Set Type］が現在のユーザセッションの \'95\'5c 示形式でない場合にのみ有効です。 **複数値属性と単一値属性** : 結果リストに表示されるレコード数は、ソート属性が単一値か複数値かによって異なります。複数値属性をソートすると、通常、一致件数の合計は同じでもレコード数が多くなります。これは、複数値属性の値がそれぞれ、単独で 1 行に \'95\'5c 示されるためです。

初期設定パネルの設定の完了

有効なエントリを送信したことを確認するには、*[送信]* をクリックします。エントリが有効でない場合、初期設定ページの上部にエラーメッセージが表示されます。エラーが解決できたら、*[リストビューに戻る]*、*[設定の保存]* の順にクリックします。

15.3 匿名アクセス用のリスト検索の環境設定

匿名アクセス用にリスト検索ポートレットを設定する

1. *[管理]* > *[ポートレット管理]* を選択します。
2. リスト検索ポートレットの新しいインスタンスを登録して名前を指定します (例 : 公開用検索)。
3. 新しいインスタンスを選択し、*[設定]* に移動します。

4 *［認証の要求］* に False を設定して、*［設定を保存］* をクリックします。

5 *［初期設定］* に移動して、次の作業を行います。

- *［デフォルト検索］* を *［基本］* または *［高度］* に変更します (匿名ユーザは保存済み検索モードを利用できません)。
- パブリックアクセス用に設定されている詳細ポートレットインスタンスの指定を検討します (［認証の要求］に False が設定されている)。デフォルトの DetailPortlet を使用する場合、結果リストリンクの詳細を表示するには、ユーザがログインする必要があります。
- *［カスタム初期設定の表示 / 編集］* に移動して、ゲストユーザに表示させないエンティティや属性を削除します。

匿名リスト検索用の共有ページを作成する

1 *［管理］* > *［ページ管理］* を選択します。

2 新しいページを作成し、それをゲストページカテゴリ (およびログインユーザ用の他のカテゴリ) に追加します。

3 *［許可の割り当て］* をクリックします。*［表示許可を管理者のみに設定］* の選択を解除します。

4 ページを保存します。

リスト検索ポートレットインスタンスに DNLookup 属性が必要な場合は、ParamListPortlet の *［認証の要求］* を False に変更してください。

プロビジョニングワークフローの環境設定と管理

V

これらの節では、プロビジョニング要求、プロビジョニングワークフロー、およびプロビジョニングチームの環境設定と管理について説明します。

16 ワークフローを開始するためのユーザアプリケーションドライバの環境設定

この節では、ユーザアプリケーションドライバの概要と、アイデンティティボールト内のイベントに基づいたワークフローの自動実行の設定方法について説明していきます。

16.1 ユーザアプリケーションドライバについて

ユーザアプリケーションドライバは、プロビジョニングワークフローを開始する役割と、アイデンティティボールトの変更 (たとえば、Designer for Identity Manager を使用してディレクトリ抽象化層を変更した場合) をユーザアプリケーションに通知する役割を果たします。このドライバでは購読者チャネルだけが使用されます。このドライバは、アイデンティティボールトからユーザアプリケーション (アプリケーションサーバで実行) へのメッセージを処理します。ユーザアプリケーションで発生し、アイデンティティボールトに返されるイベントもありますが、こうしたイベントはユーザアプリケーションドライバの発行者チャネルを使用しません。

アプリケーションサーバが起動されると、ドライバはアプリケーションサーバとのセッションを確立します。ドライバは、アプリケーションサーバで実行されているユーザアプリケーションにメッセージを送信します (「仮想ディレクトリ定義の新しいセットを取得」など)。

ドライバの \'83\'5cースコンポーネントには次の内容が含まれます。

- ComposerDriverShim.jar – Composer ドライバシムです。Windows では lib ディレクトリ (/Novell/NDS/lib) に、Linux では classes ディレクトリ (/usr/lib/dirxml/classes) にインストールされます。
- srvprvUAD.jar – アプリケーションドライバシムです。Windows では lib ディレクトリ (/Novell/NDS/lib) に、Linux では classes ディレクトリ (/usr/lib/dirxml/classes) にインストールされます。
- UserApplicationDriver.xml - 新しいドライバを設定するための環境設定データが含まれたファイルです。このファイルは DirXML.Drivers ディレクトリにインストールされます (Windows の場合 \Tomcat\webapps\nps\DirXML.Drivers、Linux の場合 /usr/lib/dirxml/rules/DirXML.Drivers)。

ユーザアプリケーションドライバコンポーネントは、Identity Manager のインストール時にインストールされます。Identity Manager ユーザアプリケーションを実行する前に、新しいドライバセットや既存のドライバセットにユーザアプリケーションドライバを追加し、ドライバをアクティブにする必要があります。

ユーザの作業環境によっては、ユーザアプリケーションドライバの設定がほとんど必要ない場合もありますし、ドライバポリシーに複雑な業務ルールセットを実装するのが望まし

い場合もあります。ユーザアプリケーションドライバは Identity Manager の他のドライバと同じく、柔軟なデータ同期メカニズムを提供しています。

16.2 ワークフローの自動起動の設定

プロビジョニングモジュールがインストールされている場合、ユーザがリソースを要求してプロビジョニング要求を開始したときに、ワークフローが自動的に起動されます。また、Identity Manager のユーザアプリケーションドライバはアイデンティティボールト内のイベントをリッスンし、イベントに応答して適切なプロビジョニングワークフローを起動します (設定されている場合)。たとえば、アイデンティティボールトに新しいユーザが追加されるとプロビジョニングワークフローが自動起動されるよう、ユーザアプリケーションドライバを設定できます。ユーザアプリケーションドライバがワークフローを自動起動するよう設定するには、Identity Manager のポリシーとルールを使用します。

16.2.1 ポリシーについて

ユーザアプリケーションドライバでも、Identity Manager の他のドライバと同じ方法でフィルタとポリシーを使用できます。アイデンティティボールトでイベントが発生すると、そのイベントを説明する XML ドキュメントが Identity Manager によって作成されます。XML ドキュメントは、チャネルを通して接続システムに渡されます (この場合、接続システムはユーザですアプリケーションドライバに関連付けられたフィルタやポリシーで、イベントに応答する方法を定義できます。また、その応答処理中に接続システムが使用できる形式に XML ドキュメントを変換する方法も定義できます。Identity Manager はいくつかのカテゴリのポリシーを提供しています (イベント変換、コ \'83\'7d ンド変換、スキー \'83\'7d\'83\'7dッピング、出力変換など)。これらのポリシーを決められた手順で適用することにより、XML ドキュメントを変換できます。

ここでは、アイデンティティボールトのイベントに基づいてワークフローを起動する例を示します。どのポリシーを使用してもワークフローを起動できますが、この節の例では最も簡単で便利な方法を示しています。

ユーザアプリケーションドライバを作成すると、ドライバが使用するためのイベント変換ポリシーが作成されます。イベント変換ポリシーは、残りの加入者チャネルポリシーで処理される XML ドキュメントを作成する役割を果たします。

注 : ユーザアプリケーションドライバの作成時に作成されたイベント変換ポリシーは変更しないでください。このポリシーの DN は Manage.Modify.Subscriber で始まります。このポリシーを変更するとワークフロープロセスが失敗することがあります。

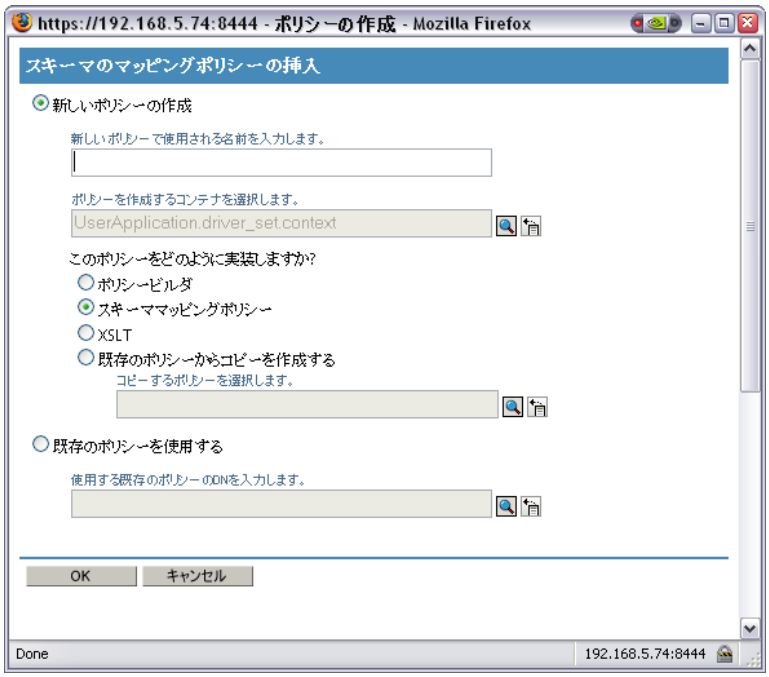
空のスキーママッピングポリシーも作成されます。このポリシーは、アイデンティティボールト内のイベントに基づいてワークフローを起動する際の開始点として使用できます。

16.2.2 Policy Builder の使用

アイデンティティボールトイベントに基づいてワークフローを自動的に開始する一番簡単な方法は、ポリシービルダを使用することです。ポリシービルダには、ワークフローを自

動開始する設定プロセスを簡素化するワークフローの開始アクションが用意されています。

1 iManager で、［*Identity Manager*］*役割を展開し、*［*Identity Manager の概要*］をクリックします。
2 ドライバセットを指定します。
3 ポリシーを管理するドライバをクリックします。*［Identity Manager ドライバの概要］*が表示されます。
4 編集するポリシーをクリックします。
5 *［挿入］*をクリックして、ポリシービルダを表示します。

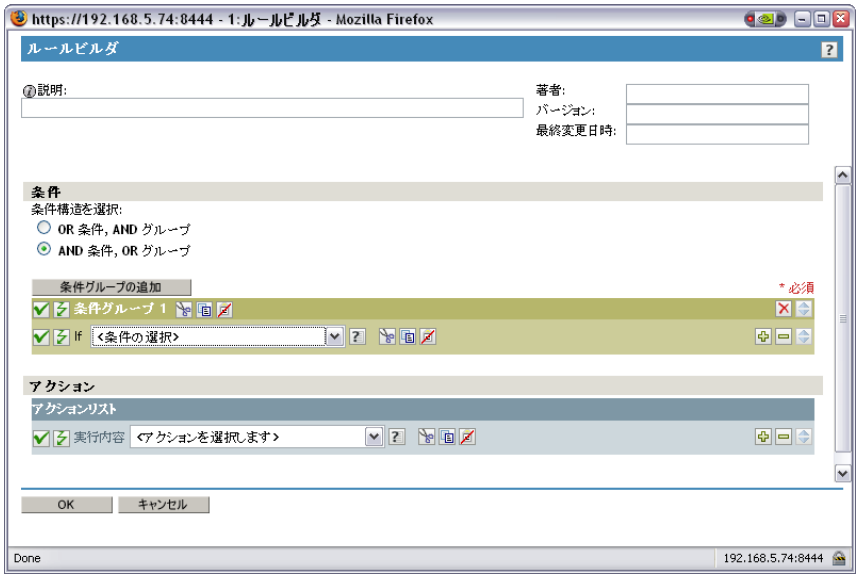


6 *［新しいポリシーの作成］*をクリックします。
7 ポリシー名を入力します。
8 *［ポリシービルダ］*をクリックします。
9 *［OK］*をクリックします。

iManager の画面に、定義されているポリシールールが表示されます。

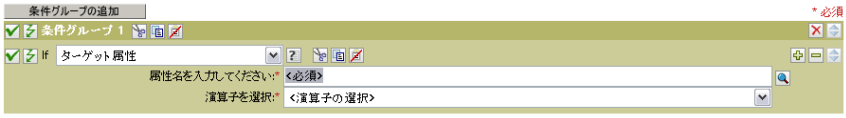
10 *［新しいルールの追加］*をクリックします。

iManager に、*［ルールビルダ］*が表示されます。



11 ルールの *［説明］*を入力します。

12 *［条件グループ 1］*の *［If］*条件で *［操作属性］*を選択します。

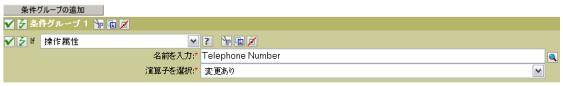


13 ワークフローの開始に使用するアイデンティティボールト属性を指定するには、*［名前を入力］*フィールドの *［属性の参照］*ボタンを使用します。

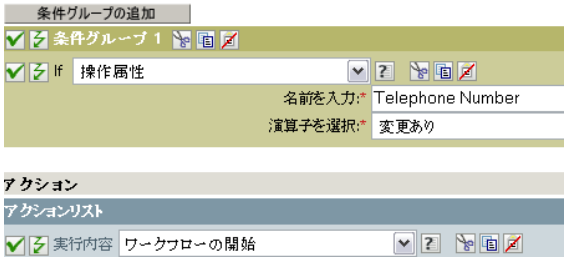
たとえば、電話番号の変更時にワークフローを開始する場合は、*［電話番号］*属性を選択します。

14 指定した属性をテストするために使用する演算子を選択する場合は、*[演算子を選択]* リストを使用します。

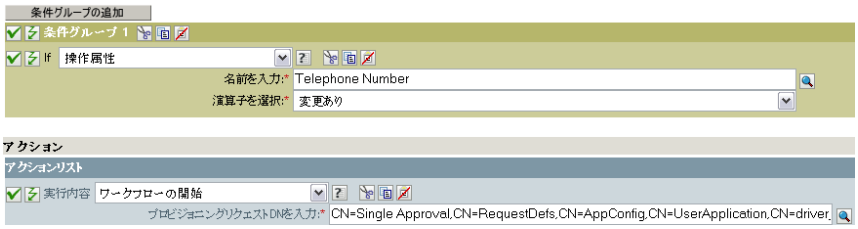
たとえば、電話番号の変更時にワークフローを開始する場合は、*[変更あり]* 属性を選択します。



15 *[アクション]* リストから、*[ワークフローの開始]* を選択します。

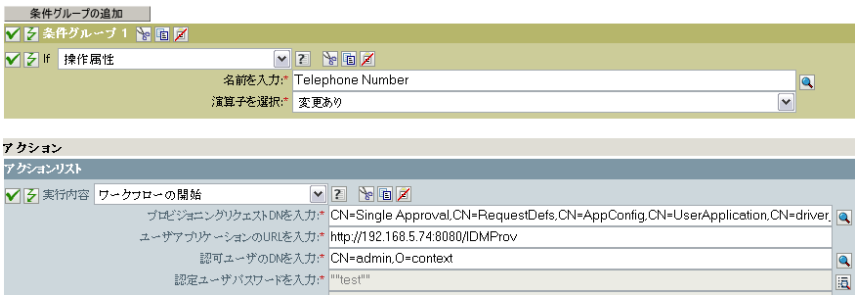


16 *If* 条件が真の場合に実行するプロビジョニング要求定義を選択するには、*[プロビジョニング要求 DN を入力]* フィールドのオブジェクトセレクタを使用します。



[ユーザアプリケーションの URL を入力] フィールドと *[認可ユーザの DN を入力]* フィールドには、自動的に値が設定されます。

17 *[認定ユーザパスワードを入力]* フィールドに、ユーザアプリケーション管理者のパスワードを入力します。

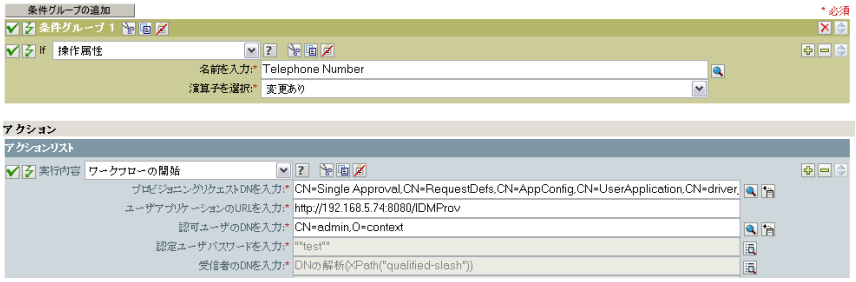


平文でパスワードを入力することはセキュリティ上の危険を伴うため、名前付きパスワードの使用をお勧めします。詳細は、『Policies in iManager in Identity Manager 3.5 (http://www.novell.com/documentation/idm35/index.html?page=/documentation/idm35/policy_imanager/data/bookinfo.html)』(Identity Manager 3.5 の iManager のポリシー) ガイドの名前付きパスワードに関する項目を参照してください。

18 *[受信者の DN を入力]* フィールドに、ワークフローの受信者の DN を LDAP 形式で指定します。

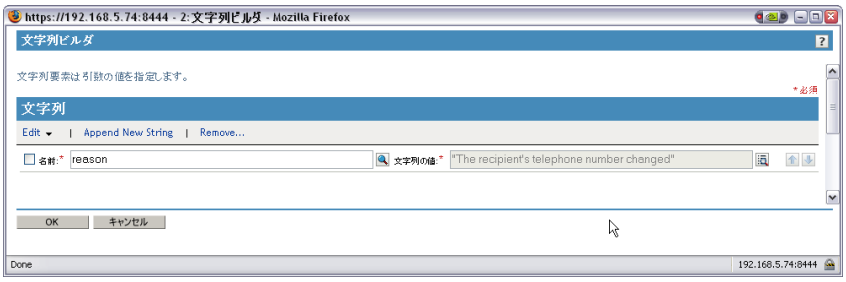
受信者 DN の式では、DN が RFC2253 の形式 (cn=user,ou=organizational unit,o=organization) に準拠していることを評価する必要があります。たとえば、*[受信者の DN を入力]* フィールドの *[引数ビルダ]* ボタンをクリックして、ワークフローに受信者の DN を渡す次の式を作成できます。

```
Parse DN("qualified-slash","ldap",XPath("@qualified-src-dn"))
```



19 *[追加の引数を入力]* フィールドに、ワークフローの引数を指定します。

ワークフローが必要とする*理由*属性を指定するには、このフィールドを使用する必要があります。*[追加の引数を入力]* フィールドの *[文字列ビルダ]* ボタンをクリックして、*理由*属性を指定して、その属性の値を作成できます (例 :「受信者の電話番号が変更された」)。



20 *[OK]* をクリックして、ルールビルダを終了します。

21 *[OK]* をクリックして、ポリシービルダを終了します。

22 *[OK]* をクリックして、[ポリシー] 画面を終了します。

23 ワークフローが必要とする属性をすべてフィルタに追加したことを確認してください。

この手順に記載している例の場合は、フィルタに *[電話番号]* と *[CN]* を追加する必要があります。フィルタへのオブジェクトの追加方法については、『Policies in iManager in Identity Manager 3.5 (http://www.novell.com/documentation/idm35/index.html?page=/documentation/idm35/policy_imanager/data/bookinfo.html)』(Identity Manager 3.5 の iManager のポリシー) ガイド中のフィルタによるオブジェクトフローの制御に関する項目を参照してください。

16.2.3 スキーママッピングポリシーエディタの使用

スキーママッピングポリシーエディタでは、ワークフローを自動開始するための手段として、ワークフローのランタイムデータをアイデンティティボールト属性にマッピングできます。作業を開始するために、ユーザアプリケーションドライバには、編集可能な空のポリシーが用意されています。ワークフローランタイムデータは、299 ページの第 17 章「プロビジョニング要求定義の設定」で説明しているワークフロー定義テンプレートから取得できます。

ワークフローの作成時、アイデンティティボールトには次のグローバル属性が作成されます。

- < *ワークフロー名* > _StartWorkflow. この属性がワークフローを開始します。
- < *ワークフロー名* > _recipient. この属性は、ワークフローが必要とするランタイムデータをアイデンティティボールトから受け取ります。
- < *ワークフロー名* > _reason. この属性は、ワークフローが必要とするランタイムデータをアイデンティティボールトから受け取ります。

他の 2 つの属性は常に存在し、ワークフローが必要とするランタイムデータをアイデンティティボールトから受け取ります。

- AllWorkflows:reason
- AllWorkflows:recipient

アイデンティティボールト内のイベントに基づいたワークフローの開始を設定する前に、次の情報があることを確認してください。

- ワークフローを開始する契機となるアイデンティティボールト属性名
- 開始するワークフロー名ワークフローにはすべて、< ワークフロー名 >_StartApprovalFlow という名前の特別な属性が含まれています。適切な eDirectory 属性をワークフローの < ワークフロー名 >_StartApprovalFlow 属性にマッピングすることにより、アイデンティティボールト内のイベントに基づいてワークフローを自動起動するよう設定できます。

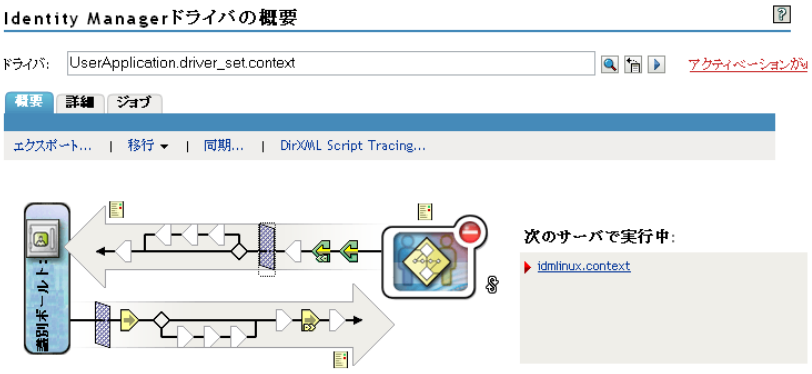
アイデンティティボールト内のイベントに基づいてワークフローが起動されるように設定する

1 iManager で、iManager ナビゲーションツリーにある［Identity Manager］の下の［*Identity Manager Overview*］*リンクをクリックします。*

［Identity Manager の概要］ページが表示されます。このページでは、ドライバセットを選択するよう求めるメッセージが \'95\'5c 示されます。

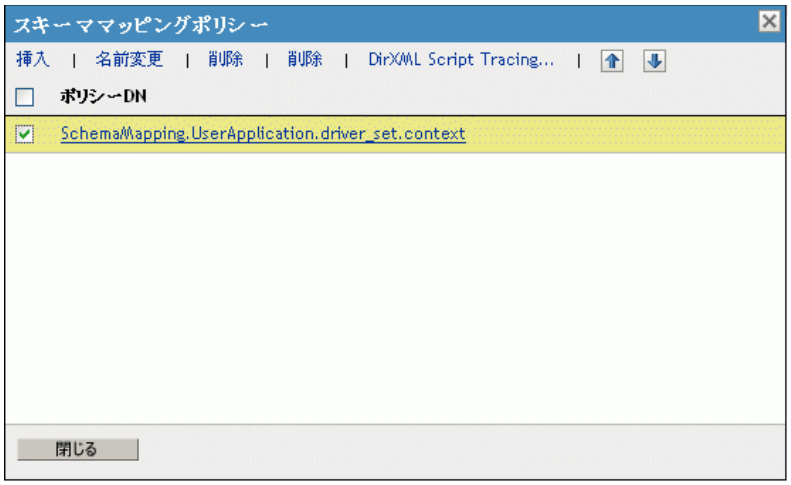
2 ［*ツリー全体を検索する*］、［*検索*］の順にクリックします。［Identity Manager の概要］ページに、現在選択されているドライバセットのドライバを表すグラフィックが表示されます。

3 ユーザアプリケーションドライバを表す大型のドライバアイコンをクリックします。

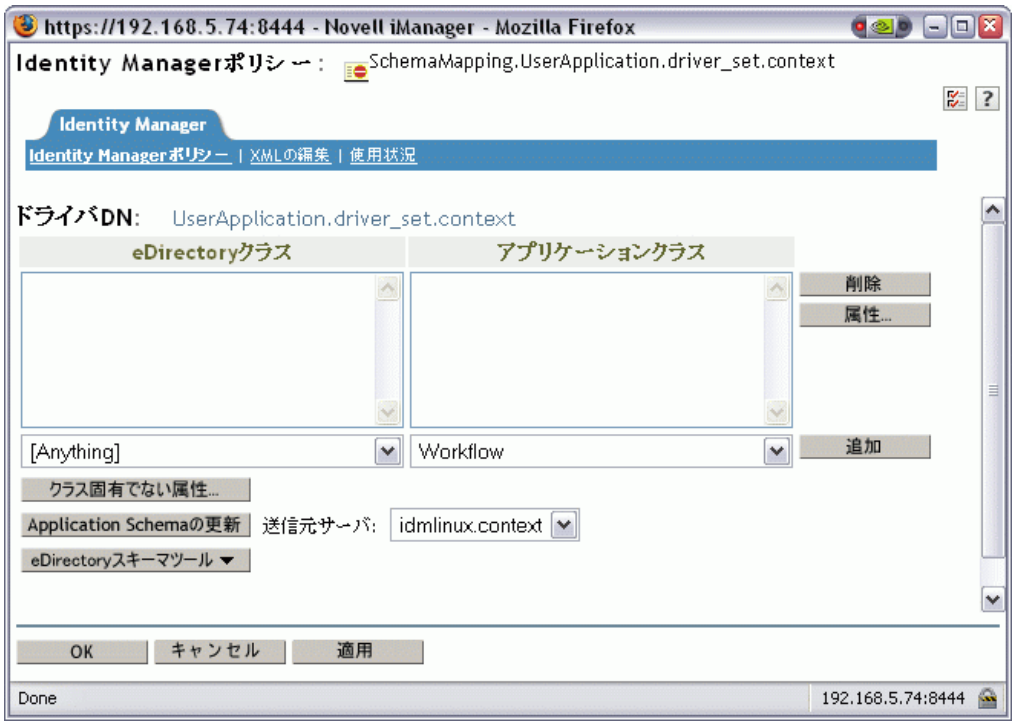


上の左向きの矢印は発行者チャネル (ユーザアプリケーションドライバでは使用されません) を表し、下の右向きの矢印は購読者チャネルを表します。グラフィック内のオブジェクトにマウスポインタを置くと、そのオブジェクトの説明が表示されます。

4 [*スキーママッピングポリシー*] アイコンをクリックします。[*スキーママッピングポリシー*] ダイアログボックスが表示されます。



5 [*編集*] をクリックします。[Identity Manager ポリシー] ダイアログボックスが表示されます。(このダイアログボックスは、アイデンティティボールトのクラスをアプリケーションクラスとマッピングしますが、この手順では eDirectory 属性をユーザアプリケーション属性とマッピングします)



6 *[アプリケーションスキーマのリフレッシュ]* をクリックします。スキーマを読み込むためにドライバを停止し、再起動するよう指示するメッセージが表示されます。スキーマのリフレッシュには約 60 秒かかります。この手順では、次の手順の準備として最新のワークフロー情報のセットが読み込まれます。この情報では識別 \'83\'7bールトから、起動されるワークフローへ移動する情報が指定されます。

7 *[OK]* をクリックして、スキーマをリフレッシュします。スキーマのリフレッシュが完了するとメッセージが表示されます。

8 *[OK]* をクリックして、スキーマのリフレッシュメッセージを閉じます。[Identity Manager Policy] ダイアログ \'83\'7bックスに戻ります。

9 *[クラスに固有でない属性]* をクリックします。Identity Manager スキーママッピングポリシーエディタが表示されます。

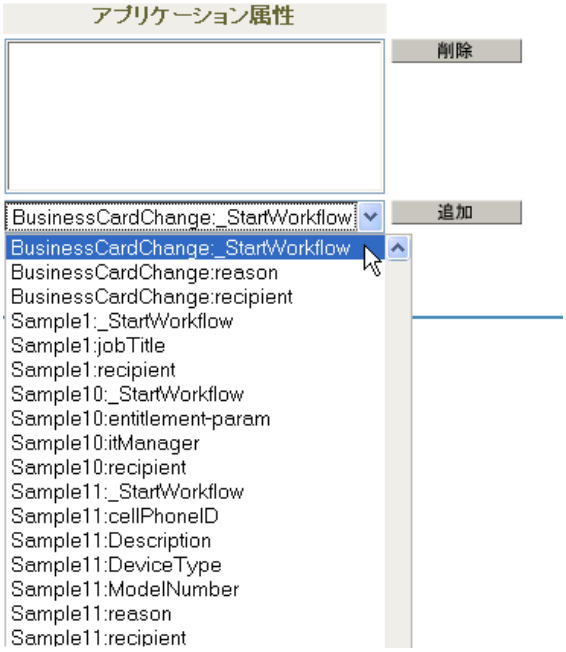
[eDirectory 属性] ドロップダウンリストには、eDirectory のすべての属性が含まれています。

[アプリケーション属性] ドロップダウンリストには、アクティブなすべてのワークフローの属性が含まれています。リスト内の属性には、AllWorkflows (属性がすべてのワークフローに適用されることを示します)、または特定のワークフロー名が先頭に付いています。同じ eDirectory 属性 (manager など) を、すべてのワークフローの manager 属性にマップする場合は、manager を Allworkflows:manager にマップします。異なる eDirectory 属性 (HRmanager など) を特定のワークフローで使用するには、eDirectory 属性を特定のワークフロー属性 (BusinessCardChange:manager など) にマップします。

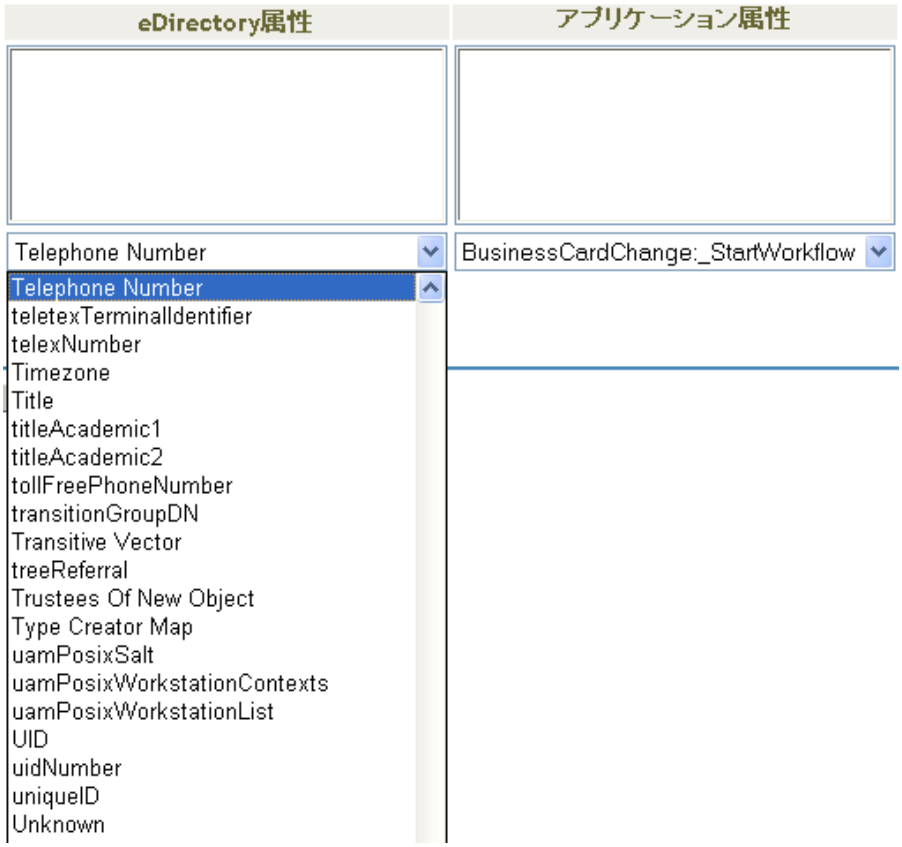
\'83\'7dップされた属性は、[*eDirectory Attributes*] 列と [Application Attributes] 列に並んで \'95\'5c 示されます。

次の手順では、ワークフローの起動に使用する eDirectory 属性をそのワークフローの _StartWorkflow 属性にマップします。ワークフローで他の eDirectory 属性も使用される可能性がある場合は、その属性もマップしてください。たとえば、eDirectory の Address 属性がワークフローのトリガである場合、ワークフローでは City や State などの属性も必要になります。代わりに、これらの属性をポリシーでマップすることもできます。

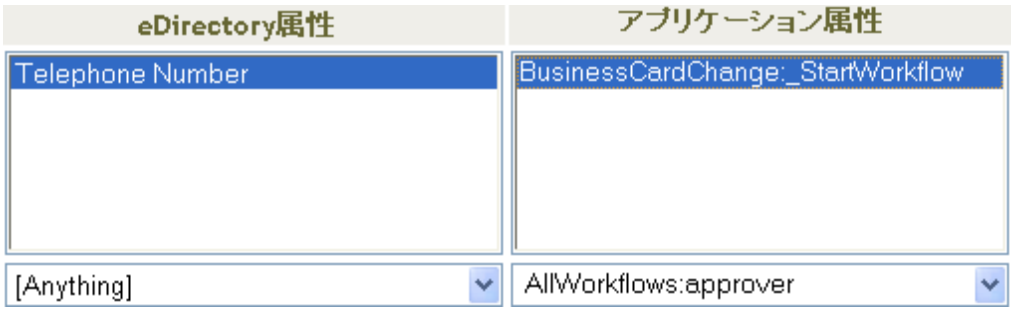
10 *[アプリケーション属性]* リストで、設定するワークフローの _StartWorkflow 属性を選択します。次の例では、BusinessCardChange ワークフローの _StartWorkflow 属性が表示されています (BusinessCardChange_StartWorkflow)。



11 ［*eDirectory 属性*］リストで eDirectory 属性を選び、その属性が変更された場合にワークフローを起動するようにします。次の例では、Telephone 属性が選択されています。この場合、従業員の電話番号が変更されると、BusinessCardChange ワークフローが起動します。



12 ［*追加*］をクリックします。eDirectory 属性がアプリケーション属性にマップされます。



13 ステップ 10 ～ステップ 12 を繰り返して、eDirectory 属性をワークフローの _reason 属性と _recipient 属性にマップします。

14 ワークフローで必要となる eDirectory 属性がまだある場合、ステップ 10 ～ステップ 12 を繰り返して、マップが必要な属性をすべてマップします。

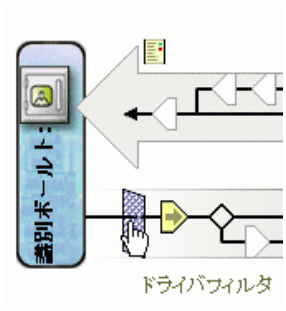
アプリケーションの _StartApprovalFlow 属性にマップされた eDirectory 属性で変化が起きると、ワークフローが自動起動されます。ただし、eDirectory 属性がドライバフィルタに含まれている場合は、eDirectory 属性はスキーママッピングポリシーに到達するだけです。次の手順では、eDirectory 属性をドライバフィルタに追加します。

15 ［*OK*］をクリックして、［Schema Mapping Policy Editor］を閉じます。

16 ［*OK*］をクリックして、［Identity Manager Policy］ダイアログ \'83\'7bックスを閉じます。

17 ［*Close*］をクリックして、［*Schema Mapping Policies*］ダイアログ \'83\'7bックスを閉じます。

18 ［*ドライバフィルタ*］アイコンをクリックします。



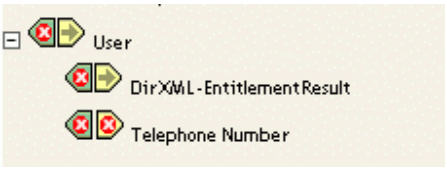
フィルタウィンドウが表示されます。

イベントフィルタでは、Identity Manager エンジンがイベントを処理するオブジェクトクラスや属性を指定します。左側にある読み込み専用の *［フィルタ］* リストでは、クラスの属性が表示されます。右側の *［Class Name］リストでは、ターゲットオブジェクトに関連付けられたオプションが \'95\'5c 示されます。*

19 フィルタに追加する属性が属しているクラスの名前をクリックします (例 :User)。

20 *［属性の追加］* をクリックします。属性の一覧が表示されます。

21 属性を選択して、*［OK］* をクリックします。*［Filter］リストに属性が追加されます。*



22 属性名をクリックします。右側のパネルに、その属性の同期オプションが \'95\'5c 示されます。

23 ［*Subscribe*］で、［Synchronize］をクリックします。

24 フィルタの他の属性を指定します。属性値への変更をレポート、同期するには、その属性の［*同期*］を選択します。属性値への変更がレポートおよび同期されないようにするには、［*Ignore*］を選択します。

25 ［*OK*］をクリックします。変更を有効にするためにドライバを再起動するかどうか尋ねるメッセージが表示されます。

26 ［*OK*］をクリックします。［Identity Manager Driver Overview］ページに戻ります。

プロビジョニング要求定義の設定 17

この節では、プロビジョニング要求定義の設定について説明します。主なトピックは次のとおりです。

17.1 プロビジョニング要求の環境設定プラグインについて

プロビジョニング要求定義を設定するには、iManager で Provisioning Request Configuration プラグインを使用する必要があります。このプラグインにより、プロビジョニング要求定義をプロビジョニングされたリソースに関連付け、関連ワークフローのランタイム特性を指定し、定義を有効にできます。このリリースでは、プロビジョニング対象リ \'83\'5cースは、Identity Manager エンタイトルメントに \'83\'7dップされます。

プロビジョニング要求の環境設定プラグインは、iManager の［Identity Manager］カテゴリ内にあります。このプラグインでは、［プロビジョニング環境設定］の役割に［プロビジョニング要求］タスクが含まれています。［プロビジョニング要求］タスクは、表 17-1 で説明している複数のパネルで構成されています。

表 17-1 *［プロビジョニング要求］タスク : パネル*

パネル	説明
Provisioning Driver Selection	Identity Manager ユーザアプリケーションのドライバを選択できます。ドライバには、一連の事前展開済みのプロビジョニング要求定義が含まれているため、プロビジョニング要求の設定を開始する前にドライバを選択する必要があります。
Provisioning Request Configuration	このツールでは、次の操作ができます。 ◆ 使用できるプロビジョニング要求定義をブラウズし、設定する定義を選択する ◆ 既存の定義に基づき、新しいプロビジョニング要求定義を作成する ◆ プロビジョニング要求定義のプロパティを設定する ◆ プロビジョニング要求定義を、プロビジョニング対象リ \'83\'5cースに割り当てる ◆ 関連ワークフロー内の各アクティビティについて、名宛人およびタイムアウトの設定を編集する 新しいプロビジョニング要求の作成または既存の要求の編集を選択した場合、プラグインにより［Provisioning Request Configuration］ウィザードが実行されます。

17.2 インストールされているテンプレートでの作業

Identity Manager の Designer では、プロビジョニング要求定義を最初から作成できます。代わりに、製品に同梱されているプロビジョニング要求テンプレートの定義後に、プロビジョニング要求をモデル化して、プロビジョニング要求を定義することもできます。テンプレートを使用するには、インストールされているテンプレートに基づいて新しいオブジェクトを定義してから、これらのオブジェクトを組織のニーズに合うようにカスタマイズします。

インストールされているテンプレートを使用すると、実行する要求に必要な承認ステップの数を指定できます。プロビジョニング要求は、次のステップを必要とするように設定できます。

- 承認なし
- 1 つの承認ステップ
- 2 つの承認ステップ
- 3 つの承認ステップ
- 4 つの承認ステップ
- 5 つの承認ステップ

シーケンシャル処理またはパラレル処理のどちらをサポートするか、および処理中にワークフローのタイムアウトが発生した場合に要求を承認するか拒否するかも指定できます。

Identity Manager には、表 17-2 に記載されているテンプレートが同梱されています。

表 17-2 *プロビジョニング要求用テンプレート*

テンプレート	説明
Self Provision Approval	承認なしにプロビジョニング要求を実行できるようにします。
One Step Approval (Timeout Approves)	プロビジョニング要求の実行に、1 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。
Two Step Sequential Approval (Timeout Approves)	プロビジョニング要求の実行に、2 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはシーケンシャル処理をサポートします。
Three Step Sequential Approval (Timeout Approves)	プロビジョニング要求の実行に、3 つの承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはシーケンシャル処理をサポートします。

テンプレート	説明
Four Step Sequential Approval (Timeout Approves)	プロビジョニング要求の実行に、4 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはシーケンシャル処理をサポートします。
Five Step Sequential Approval (Timeout Approves)	プロビジョニング要求の実行に、5 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはシーケンシャル処理をサポートします。
One Step Approval (Timeout Denies)	プロビジョニング要求の実行に、1 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはシーケンシャル処理をサポートします。
Two Step Sequential Approval (Timeout Denies)	プロビジョニング要求の実行に、2 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはシーケンシャル処理をサポートします。
Three Step Sequential Approval (Timeout Denies)	プロビジョニング要求の実行に、3 つの承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはシーケンシャル処理をサポートします。
Four Step Sequential Approval (Timeout Denies)	プロビジョニング要求の実行に、4 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはシーケンシャル処理をサポートします。
Five Step Sequential Approval (Timeout Denies)	プロビジョニング要求の実行に、5 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはシーケンシャル処理をサポートします。
Two Step Parallel Approval (Timeout Approves)	プロビジョニング要求の実行に、2 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはパラレル処理をサポートします。

テンプレート	説明
Three Step Parallel Approval (Timeout Approves)	プロビジョニング要求の実行に、3 つの承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはパラレル処理をサポートします。
Four Step Parallel Approval (Timeout Approves)	プロビジョニング要求の実行に、4 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはパラレル処理をサポートします。
Five Step Parallel Approval (Timeout Approves)	プロビジョニング要求の実行に、5 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、アクティビティは要求を承認し、ワーク項目は次のアクティビティに送られます。 このテンプレートはパラレル処理をサポートします。
Two Step Parallel Approval (Timeout Denies)	プロビジョニング要求の実行に、2 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはパラレル処理をサポートします。
Three Step Parallel Approval (Timeout Denies)	プロビジョニング要求の実行に、3 つの承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはパラレル処理をサポートします。
Four Step Parallel Approval (Timeout Denies)	プロビジョニング要求の実行に、4 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはパラレル処理をサポートします。
Five Step Parallel Approval (Timeout Denies)	プロビジョニング要求の実行に、5 段階の承認を必要とします。アクティビティのタイムアウトが発生した場合、ワークフローは要求を拒否します。 このテンプレートはパラレル処理をサポートします。

ワークフローおよびプロビジョニングされたリソース 新しいプロビジョニング要求定義を作成する場合、それをプロビジョニングされたリソースにバインドします。要求定義に関連するプロビジョニング対象リ \'83\'5cースは変更できますが、ワークフローまたはそのトポロジは変更できません。

プロビジョニング要求のカテゴリ プロビジョニング要求のテンプレートもそれぞれ、カテゴリに関連付けられています。エンドユーザはカテゴリを使って、プロビジョニング要求を整理できます。すべてのプロビジョニング要求テンプレートのデフォルトのカテゴリは、*Entitlements* です。カテゴリキー、すなわち srvprvCategoryKey 属性の値は *entitlements*(小文字) です。

ディレクトリ抽象化層エディタを使用すると、独自のカテゴリを作成できます。新しいカテゴリを作成する際には、カテゴリキー (srvprvCategoryKey の値) が小文字になるように

注意してください。これは、Identity Manager ユーザアプリケーションでカテゴリが適切に機能するために必要です。

プロビジョニングカテゴリの作成方法の詳細は、『*Identity Manager ユーザアプリケーション: 設計ガイド*』を参照してください。

17.3 プロビジョニング要求定義の設定

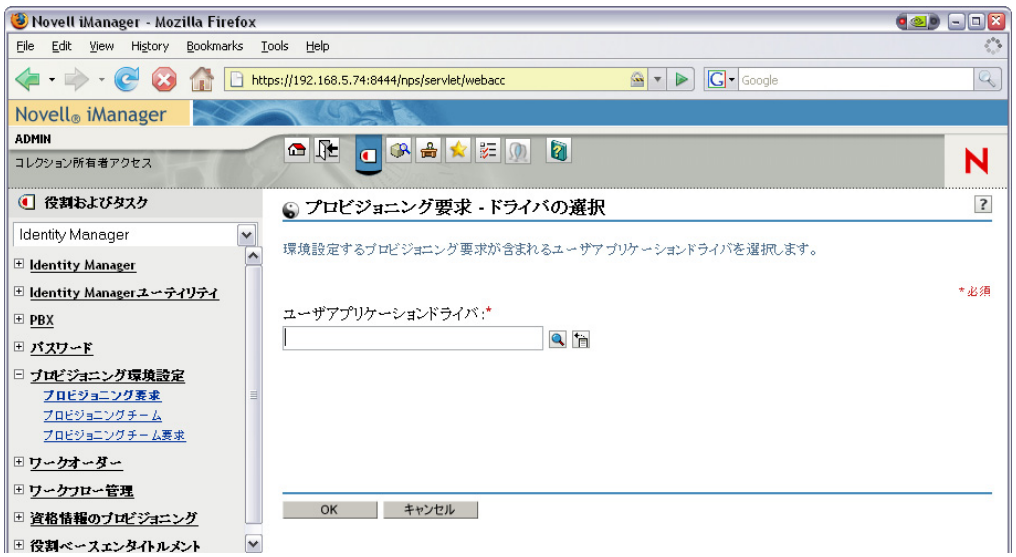
プロビジョニング要求定義を設定する前に、定義が含まれる Identity Manager ユーザアプリケーションドライバを選択する必要があります。ドライバを選択したら、新しいプロビジョニング要求定義を作成したり、既存の定義を編集することができます。また、プロビジョニング要求定義を削除したり、要求定義のステータスを変更したり、要求定義の権利を定義したりすることもできます。

17.3.1 ドライバの選択

Identity Manager ユーザアプリケーションドライバを選択する

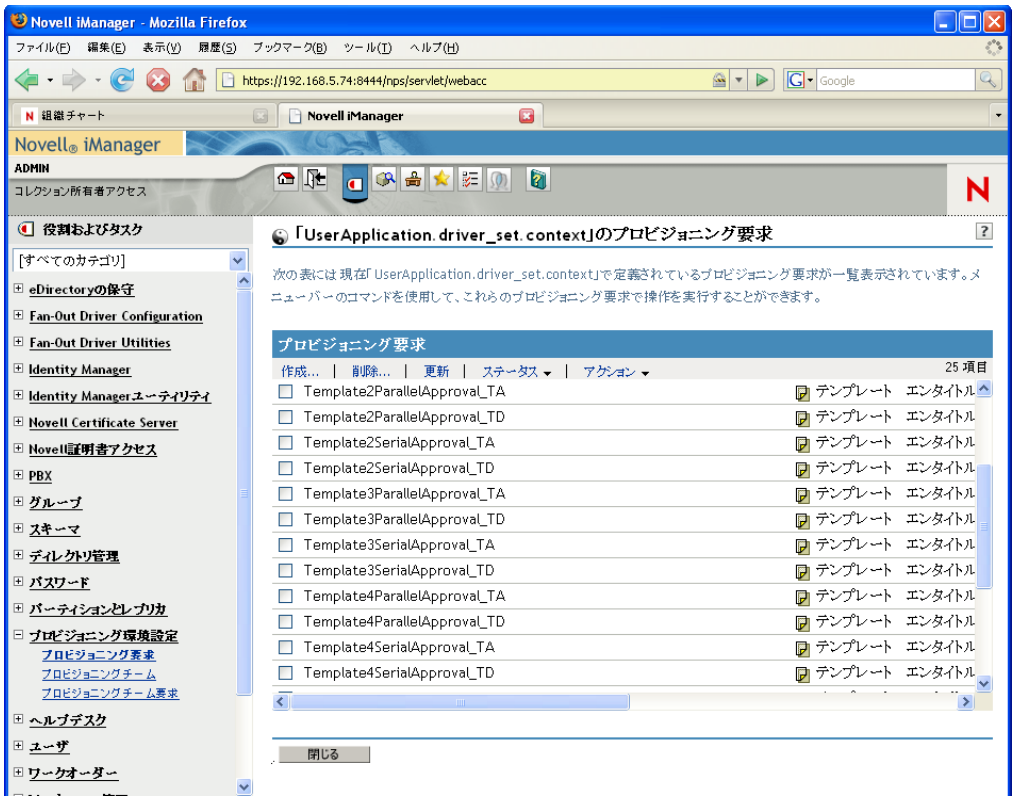
1. iManager で、*[Identity Manager]* カテゴリを選択します。
2. *[Provisioning Request Configuration]* 役割を開きます。
3. *[プロビジョニング要求]* タスクをクリックします。

   iManager に［ユーザアプリケーションドライバ］パネルが表示されます。



4. *[ユーザアプリケーションドライバ]* フィールドでドライバ名を指定し、*[OK]* をクリックします。

   ［プロビジョニング要求の環境設定］パネルが表示されます。［Provisioning Request Configuration］パネルには、使用できるプロビジョニング要求定義のリストが \'95\'5c示されます。



インストールされているテンプレートは、*[テンプレート]* というステータスとともに黒字で表示されます。テンプレートの要求定義には、ハイパーテキストリンクは \'95\'5c 示されません。これらは読み込み専用であるためです。

注 : 要求定義がローカライズ済みテキストを使用するよう設定されている場合、これらの定義の名前および説明には、現在のロケースに応じたテキストが \'95\'5c 示されます。

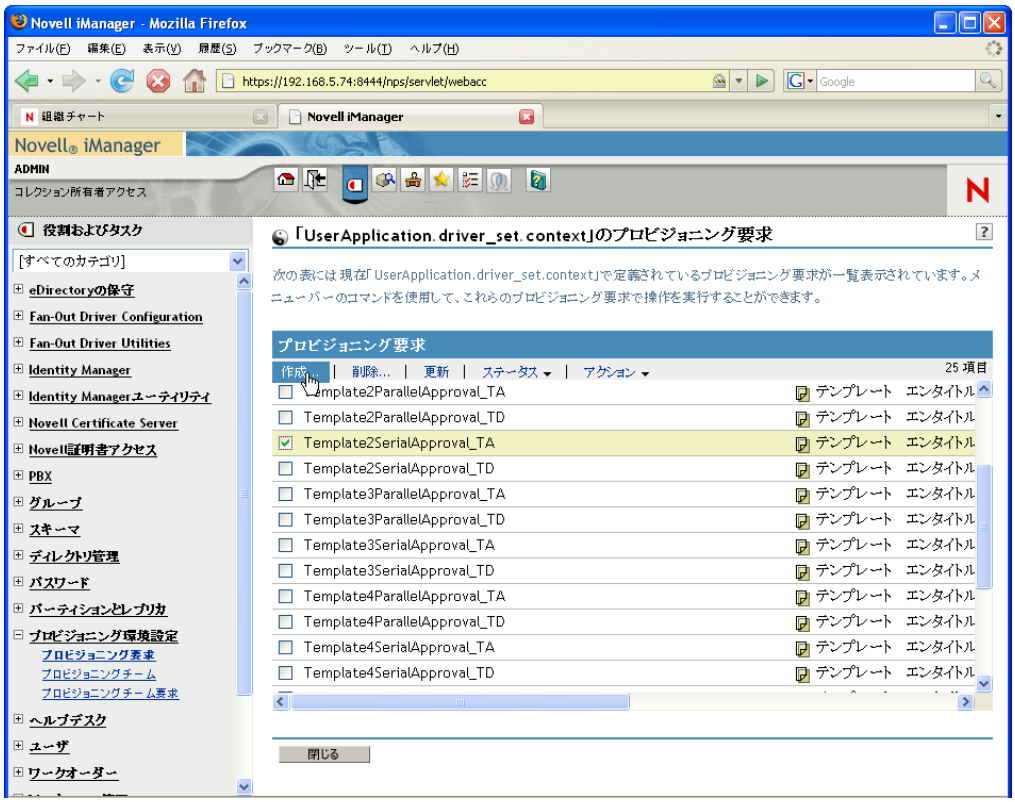
ドライバの変更 いったんドライバを選択すると、別のドライバを選択しない限り、iManager セッション中は選択したドライバが有効となります。新しいドライバを選択するには、*[アクション]* コマンドをクリックし、*[アクション]* メニューから *[ユーザアプリケーションドライバの選択]* を選択します。

17.3.2 プロビジョニング要求の作成または編集

プロビジョニング要求を作成する

1 ［プロビジョニング要求の環境設定］パネルで、テンプレートとして使用するプロビジョニング要求の名前をクリックします。

2 ［Provisioning Request Configuration］パネルの［*Create From*］コ \'83\'7d ンドをクリックします。



［Configure New Provisioning Request］ウィザードの最初のページが \'95\'5c 示されます。

3 ［*Name*］フィールドに、*新しいオブジェクトの共通の名前を入力します。*

4 アプリケーションでサポートする各言語について、*［プロビジョニング要求のローカライズされた文字列］*の*［表示名］*および*［説明］*の各フィールドにローカライズされたテキストを入力します。このテキストは、ユーザアプリケーションにおけるプロビジョニング要求の識別に使用されます。

5 新しい言語をリストに追加するには、*［追加］*をクリックしてから目的の言語を選択します。

デフォルトでは、新しく作成されたプロビジョニング要求は英語のみをサポートします。

6 ［*次へ*］をクリックします。

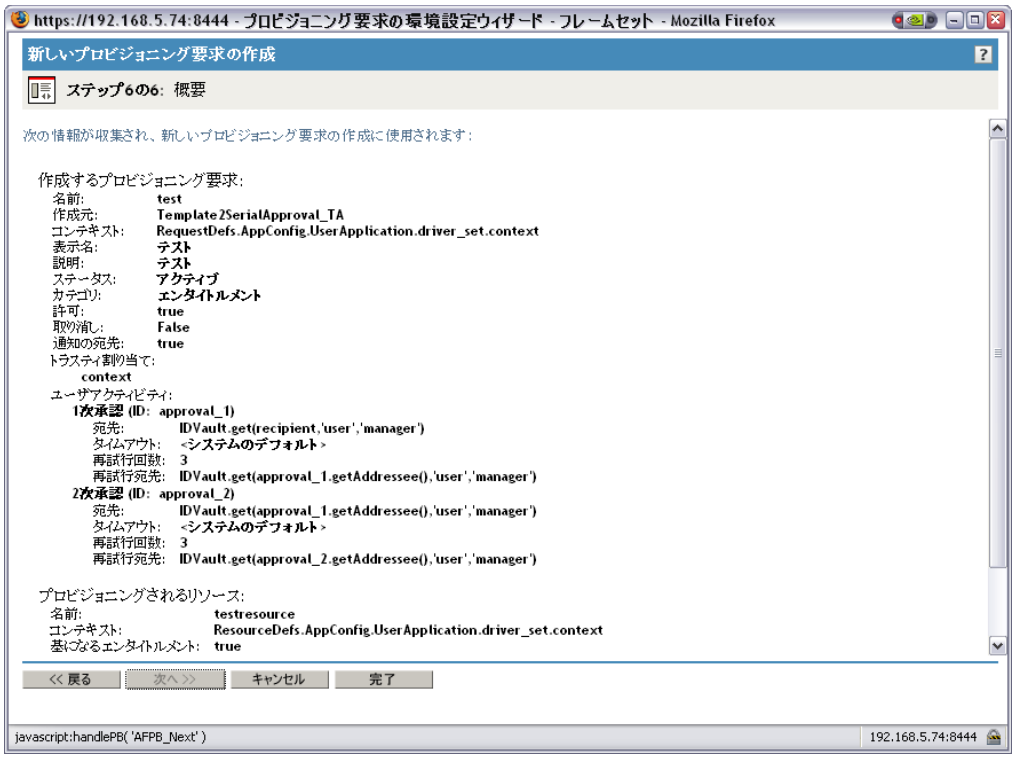
7 の説明に従い、要求定義のプロビジョニング対象リ\'83\'5cースを指定します。308 ページの 「プロビジョニングされたリソースの指定」

8 の説明に従い、要求定義に関連するワークフローのアクティビティを設定します。312 ページの 「ワークフローアクティビティの設定」

9 の説明に従い、要求定義のアクセス権を指定します。322 ページの 「プロビジョニング要求のアクセス権の指定」

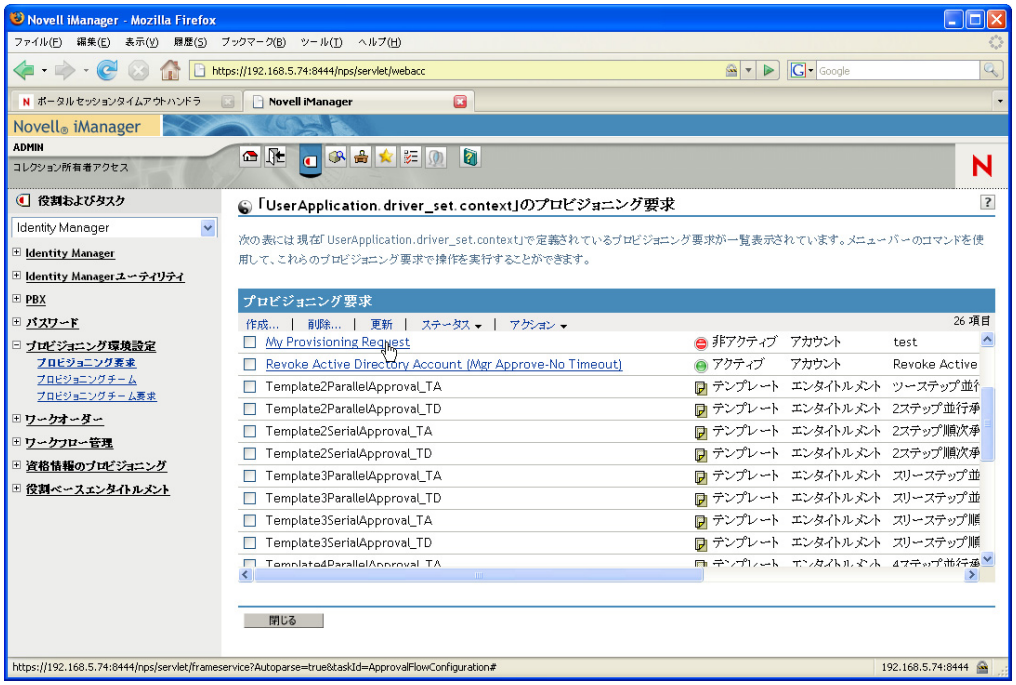
10 の説明に従い、要求定義の初期ステータスを指定します。323 ページの 「プロビジョニング要求の初期ステータスの指定」

11 設定内容を確認し、*［完了］*をクリックします。



既存のプロビジョニング要求を編集する

1 [プロビジョニング要求の環境設定] パネルで、プロビジョニング要求の名前をクリックします。



テンプレートのプロビジョニング要求を編集することはできません。ステータスが［Template］となっている要求定義には、ハイパーテキストリンクは \'95\'5c 示されません。これらは読み込み専用であるためです。

多数の要求定義が存在する場合は、*［名前］*、*［説明］* など特定の列でソートしなければならない場合があります。列の見出しをクリックすると、その列を基準にソートできます。

2 アプリケーションでサポートする各言語について、*［プロビジョニング要求のローカライズされた文字列］* の下に一覧表示される言語の横にあるチェックボックスをオンにし、*［表示名］* と *［説明］* の各フィールドにローカライズされたテキストを入力します。このテキストは、ユーザアプリケーションにおけるプロビジョニング要求の識別に使用されます。

3 新しい言語をリストに追加するには、*［追加］* をクリックしてから目的の言語を選択します。

デフォルトでは、新しく作成されたプロビジョニング要求は英語のみをサポートします。

4 ［*次へ*］をクリックします。

5 の説明に従い、要求定義のプロビジョニング対象リ \'83\'5cースを指定します。308 ページの 「プロビジョニングされたリソースの指定」

6 の説明に従い、要求定義に関連するワークフローのアクティビティを設定します。312 ページの 「ワークフローアクティビティの設定」

7 の説明に従い、要求定義のアクセス権を指定します。322 ページの 「プロビジョニング要求のアクセス権の指定」

8 の説明に従い、要求定義の初期ステータスを指定します。323 ページの 「プロビジョニング要求の初期ステータスの指定」

9 設定内容を確認し、*［完了］* をクリックします。

プロビジョニングされたリソースの指定

この節では、エンタイトルメントに基づくプロビジョニングされたリソースの指定について詳しく説明します。エンタイトルメントの概念、またはエンタイトルメントの作成および使用については説明しません。

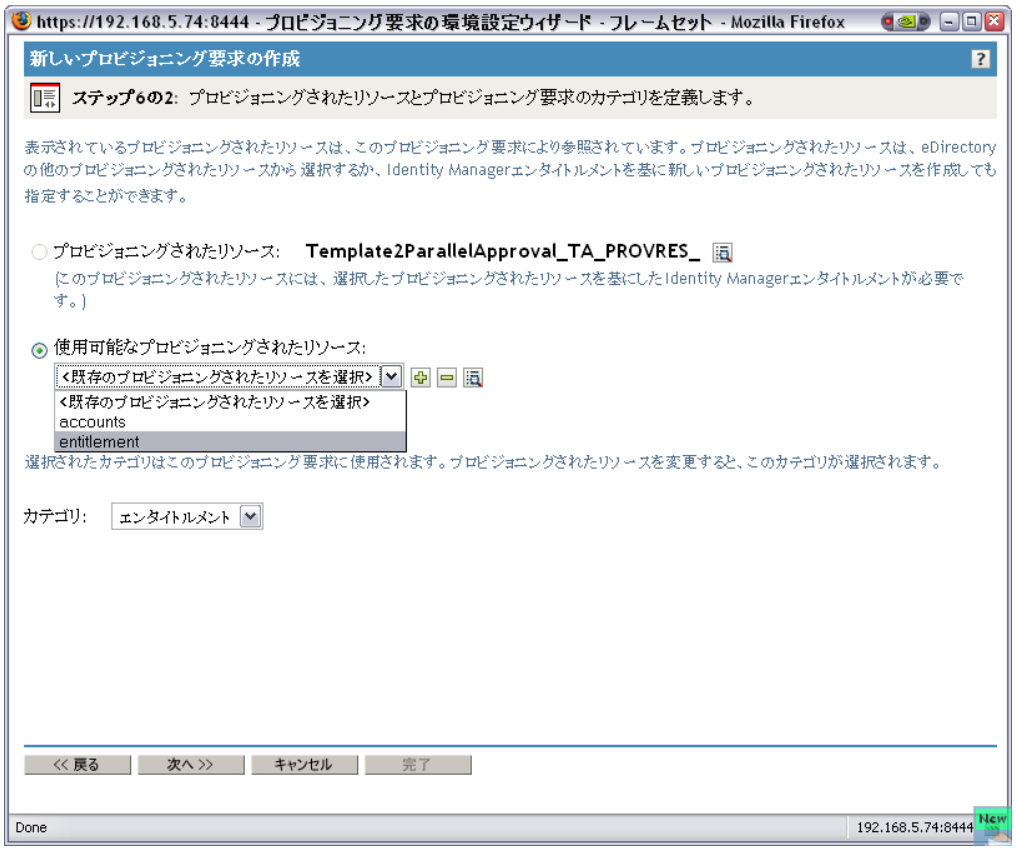
エンタイトルメントの詳細については、『*Novell Identity Manager: 管理ガイド*』を参照してください。

プロビジョニング対象リ \'83\'5cースを指定する

1 要求定義に現在関連付けられているターゲットを使用する場合には、*［プロビジョニングされたリソース］* を選択します。

有効なリソースを参照するリクエスト定義を編集する場合、デフォルトでは *［プロビジョニングされたリソース］* が選択されています。新しいプロビジョニング要求を定義している場合には、このオプションは選択されていません。

2 現在選択しているドライバ内で、以前定義された別のリソースに要求定義を関連付ける場合は、*［使用可能なプロビジョニングされたリソース］* を選択し、ドロップダウンリストからターゲットを選択します。



エンタイトルメントではないリ\'83\'5cースに要求定義が関連付けられている場合、リ\'83\'5cースを変更することはできません。

3 *[カテゴリ]* ドロップダウンリストから、プロビジョニングされたリソース定義のカテゴリを選択します。

デフォルトのカテゴリは、現在選択しているプロビジョニングされたリソースのカテゴリになります。プロビジョニングされたリソースを変更すると、リソースのカテゴリに合致するように、リクエスト定義のカテゴリも変わります。リクエスト定義に別のカテゴリを割り当てる場合は、*[カテゴリ]* ドロップダウンリストからカテゴリを選択します。

4 エンタイトルメントに基づいて新しいリソースを作成する場合は、*[+]* アイコンをクリックします。

既存のリソースを編集する場合は、ペンの形をしたアイコンをクリックします。

リソースの特徴を定義する

4a ［Name (CN)］*フィールドにリ\'83\'5cースの名前を入力します。*

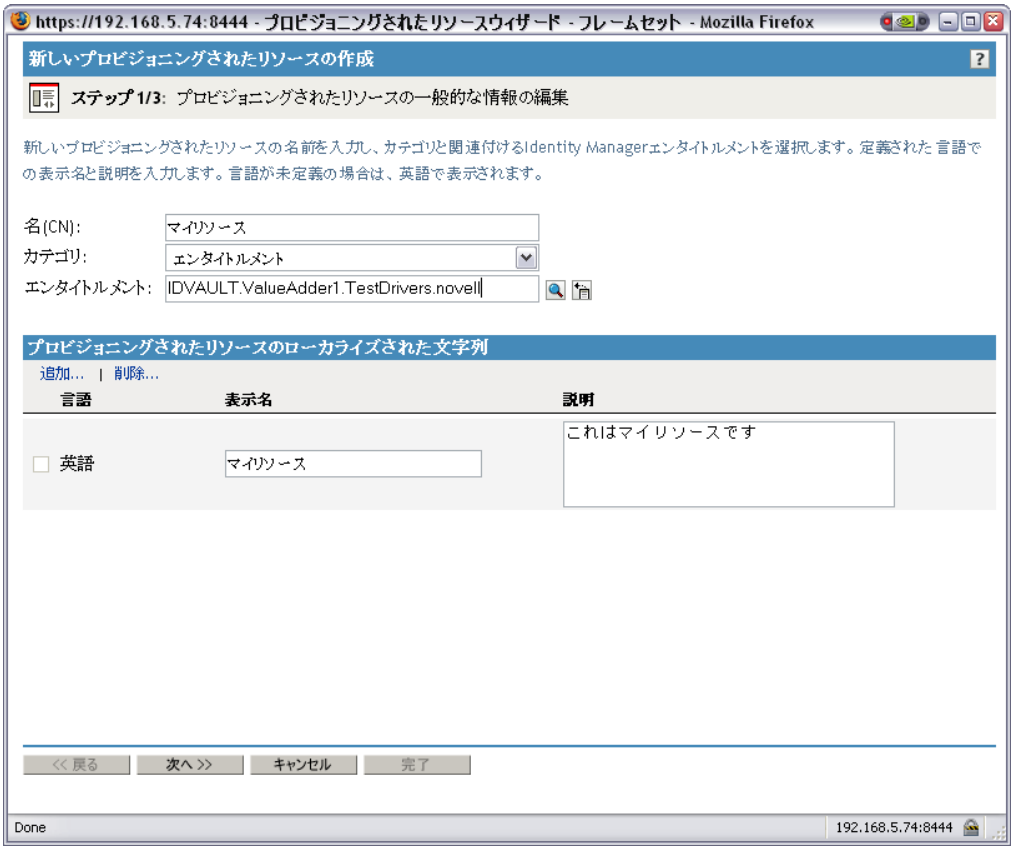
4b *［カテゴリ］* ドロップダウンリストから、リソースのカテゴリを選択します。

4c ［*Entitlement*］*フィールドで、エンタイトルメントを指定します。*

4d アプリケーションでサポートする各言語について、*[プロビジョニングされたリソースのローカライズされた文字列]* に一覧表示されている言語の横にあるチェックボックスをオンにし、*[表示名]* および *[説明]* の各フィールドにローカライズされたテキストを入力します。このテキストは、ユーザアプリケーションにおけるプロビジョニングリソースの識別に使用されます。

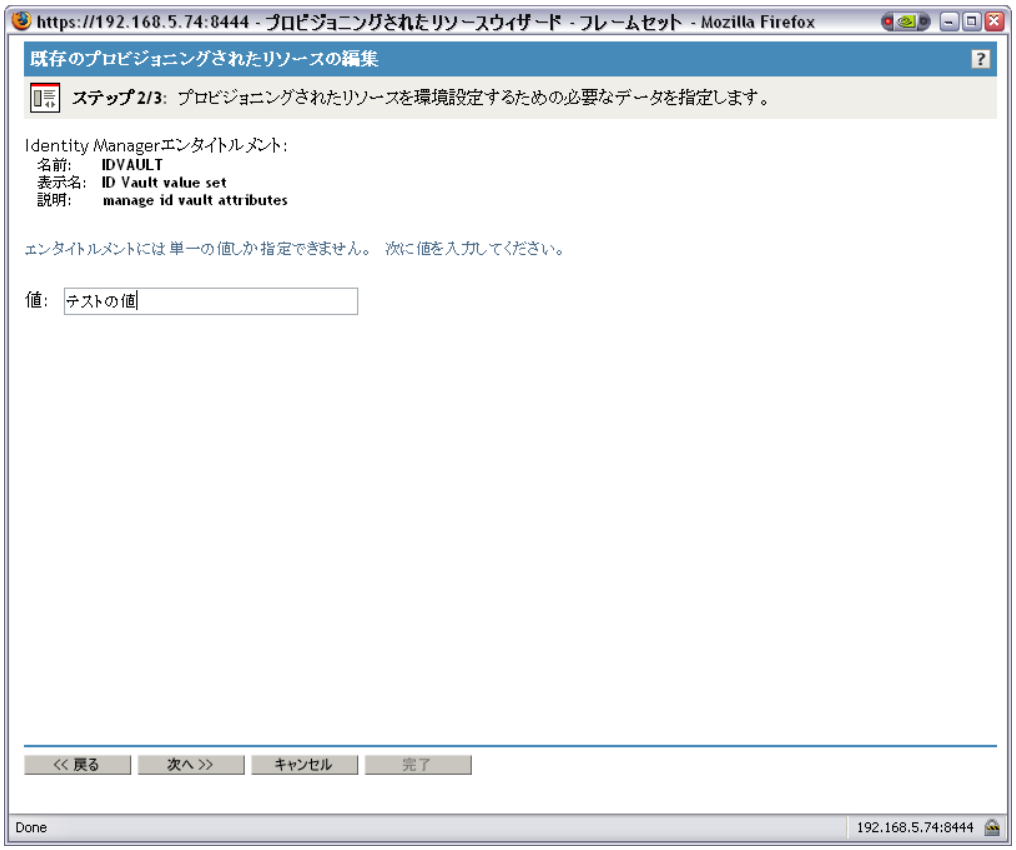
4e 新しい言語をリストに追加するには、*[追加]* をクリックしてから目的の言語を選択します。

デフォルトでは、新しく作成されたプロビジョニングリ \'83\'5cースは英語のみをサポートします。



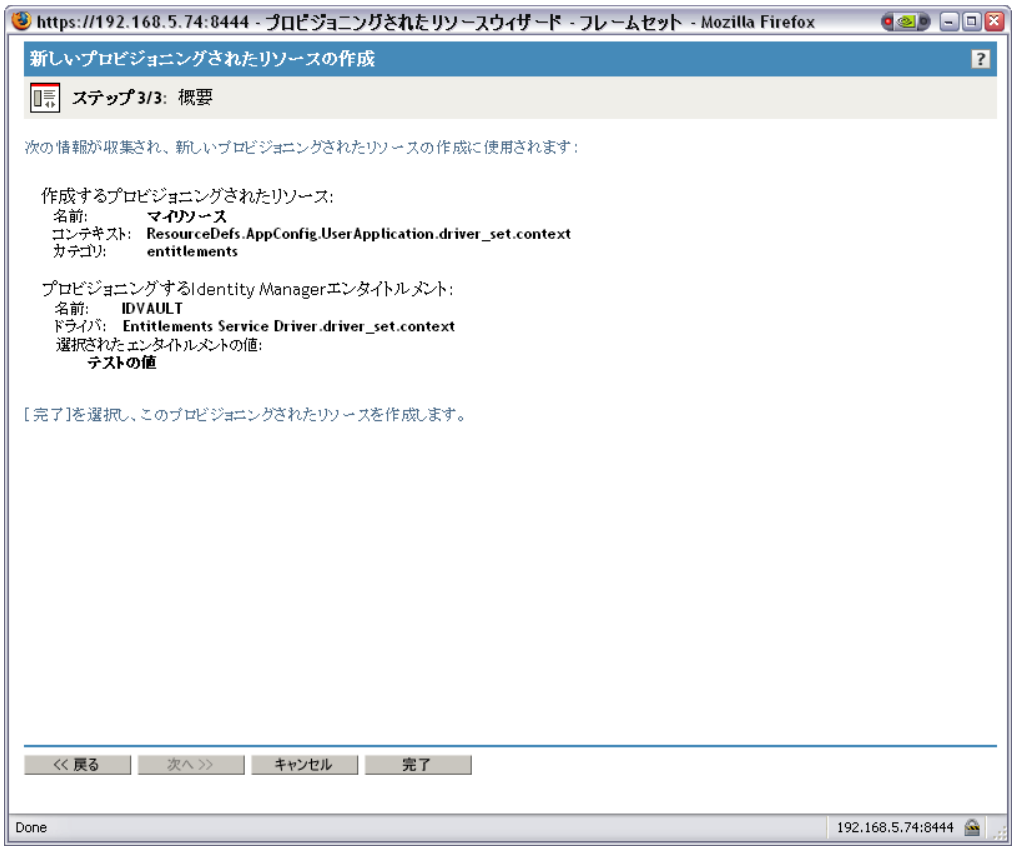
5 [*次へ*] をクリックします。

プロビジョニングされたリソースウィザードに、エンタイトルメントに必要なパラメータを入力するためのページが表示されます。



6 エンタイトルメントにパラメータが必要ない場合は、［Next］*をクリックします。*

［Create New Provisioned Resource］ウィザードに［Summary］ページが \'95\'5c 示され、定義するリ \'83\'5cースについての情報が \'95\'5c 示されます。

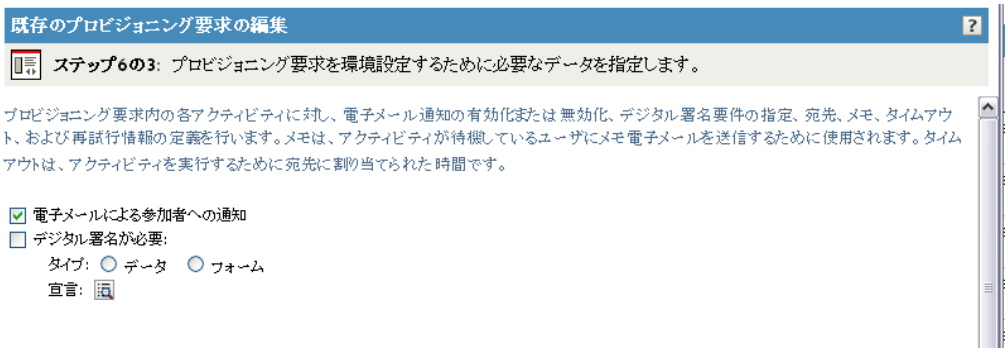


7 [*完了*] をクリックします。

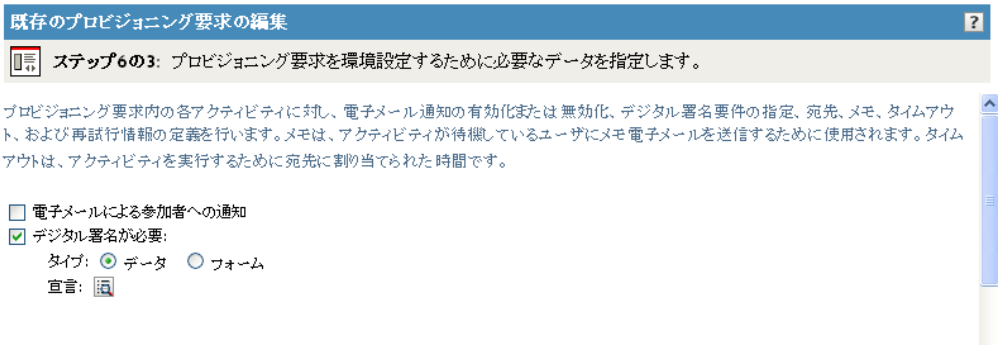
ワークフローアクティビティの設定

関連ワークフローのアクティビティを設定する

1 [*電子メールで参加者に通知*] チェックボックスを選択、または選択解除してワークフローの電子メール通知を有効にするかどうかを指定します。



2 [*デジタル署名が必要*] チェックボックスを選択、または選択解除して、プロビジョニング要求を開始するためにデジタル署名が必要かどうかを指定します。

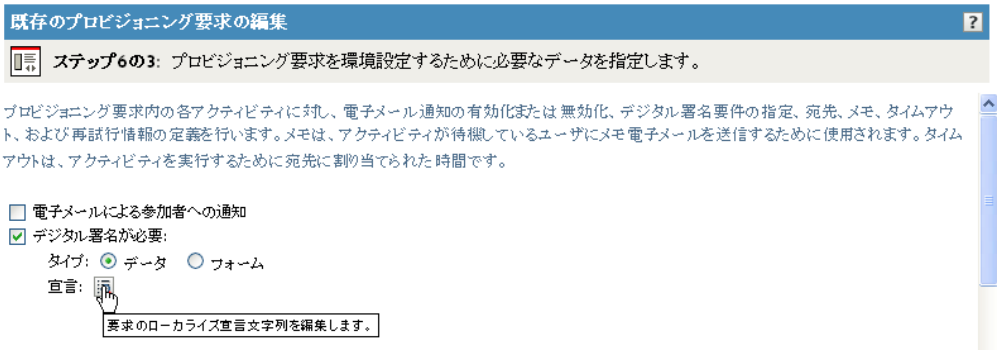


2a *[デジタル署名が必要]* チェックボックスを選択した場合は、デジタル署名がデータを使用するのか、またはフォームを使用するのかを指定します。

- *[データ]* は、XML 署名がユーザ合意として使用されることを示します。［データ］を選択した場合、XML データが監査ログに記録されます。
- *[フォーム]* は、生成するデジタル署名宣言を含む PDF ドキュメントを示します。このドキュメントがユーザ合意として使用されます。ユーザは、要求の送信や承認前に、生成された PDF ドキュメントをプレビューできます。［フォーム］を選択した場合、PDF ドキュメント (XML 内にカプセル化) が監査ログに記録されます。

警告 : デジタル署名したドキュメントを保持するには、Novell Audit(または Sentinel) を使用する必要があります。デジタル署名ドキュメントはユーザアプリケーションデータベースには保管されません。ログデータベースに保管されます。これらのドキュメントを保管するには、ログを有効にする必要があります。

2b *[デジタル署名が必要]* チェックボックスを選択した場合は、デジタル署名確認文字列も指定する必要があります。そのためには、*[宣言]* アイコンをクリックします。



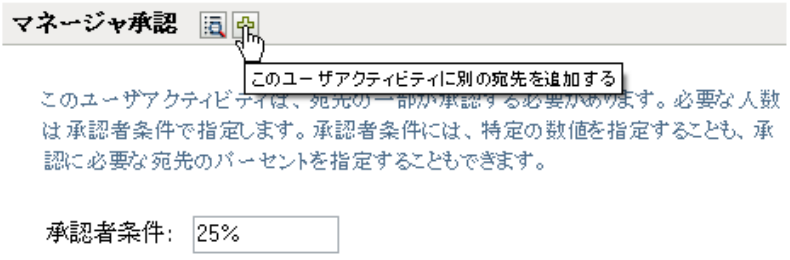
署名確認文字列を入力して、*[OK]* をクリックします。

3 (オプション) アクティビティの名前の横にあるアイコンをクリックして、各ワークフローアクティビティに対して表示ラベルを変更します (この例では *[マネージャ承認]*)。

[表示ラベル] フィールドに表示ラベルを入力し、*[OK]* をクリックします。

デフォルトの表示ラベル (［1 次承認］、［2 次承認］ など) は、承認がシーケンシャルに処理されることを示しています。パラレルフローの場合、シーケンシャル処理を暗示しないラベルを指定する必要があることもあります。たとえば、One of Three Parallel Approvals、Two of Three Parallel Approvals などのラベルを割り当てる場合です。

4 (オプション) 定数または複数アドレスをサポートする各ワークフローアクティビティに対して、アクティビティ名の隣にある *［このユーザアクティビティに別の宛先を追加する］* アイコンをクリックして、他のアドレスを追加します。

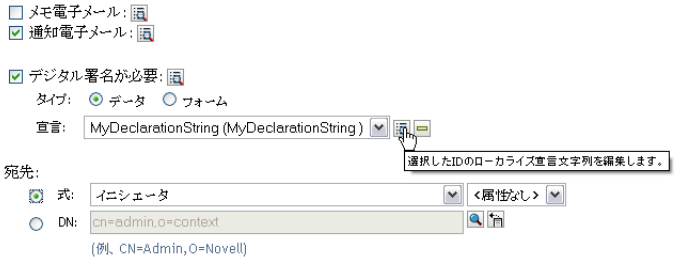


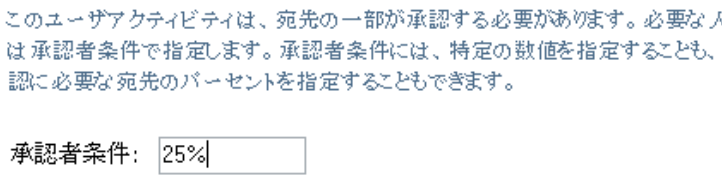
このボタンをクリックすると、ページに新しい［アドレス］セクションが表示されます。このセクションのコントロールを使って、宛先の式または DN を定義することができます (この手順の次のステップで説明)。宛先を削除するには、宛先の隣にあるマイナス記号をクリックします。

5 ワークフローアクティビティそれぞれについて、次の情報を入力します。

フィールド	説明
メモ電子メール	このアクティビティのメモ電子メールメッセージを送信するかどうかを指定します。 メモ通知を設定するには、*[このアクティビティのメモ電子メールを編集します]* をクリックします。次の項目を設定します。 ◆ *[開始]* には、最初のメモを送信する時期を指定します。開始の値は、アクティビティに関連する最初の割り当て時刻からのオフセットになります。 ◆ *[間隔]* には、最初のメモを送信した後の、メモの送信間隔を指定します。 ◆ *[電子メールテンプレート]* には、メモ電子メールメッセージのテンプレートとして使用する、言語に依存しない名前を指定します。テンプレート名を指定すると、通知エンジンによりランタイム時に使用する言語固有のテンプレートが判断されます。 言語に依存しないテンプレートには、任意の名前を指定できます。メモ電子メールメッセージのデフォルトテンプレート名を次に示します。 Provisioning Reminder 言語固有版の各テンプレートには、言語コードに対応したサフィックスを付ける必要があります (たとえば、フランス語の場合は _fr、スペイン語の場合は _ex)。

フィールド	説明
通知電子メール	このアクティビティの通知電子メールメッセージを送信するかどうかを指定します。 通知電子メールメッセージを設定するには、*[このアクティビティの通知電子メールを編集します]* アイコンをクリックします。次の項目を設定します。 ◆ *[電子メールテンプレート]* には、通知電子メールメッセージのテンプレートとして使用する、言語に依存しない名前を指定します。テンプレート名を指定すると、通知エンジンによりランタイム時に使用する言語固有のテンプレートが判断されます。 言語に依存しないテンプレートには、任意の名前を指定できます。通知電子メールメッセージのデフォルトテンプレート名を次に示します。 Provisioning Notification 言語固有版の各テンプレートには、言語コードに対応したサフィックスを付ける必要があります (たとえば、フランス語の場合は _fr、スペイン語の場合は _ex)。 ◆ *[置換パラメータマップ]* には、電子メールテンプレートで使用される置換パラメータの代替値を 1 つまたは複数指定します。既存の値を編集するには、置換パラメータをクリックしてから、ECMAScript 式または固定値を指定します。新しい代替値を追加するには、[追加] をクリックしてターゲットパラメータを選択し、次に式または固定値を指定します。

フィールド	説明
デジタル署名が必要	要求の承認にデジタル署名が必要かどうかを指定します。各承認ステップには複数の送信リンクがある場合もあるため、各リンクに対してデジタル署名が必要かどうかを指定する必要があります。 このチェックボックスを選択すると、デジタル署名を必要とするリンクの選択を要求するメッセージが表示されます。リンクを選択して、*[閉じる]* をクリックします。  *[デジタル署名が必要]* チェックボックスを選択した場合は、デジタル署名がデータを使用するのか、またはフォームを使用するのかを指定します。 • *[データ]* は、XML 署名がユーザ合意として使用されることを示します。［データ］を選択した場合、XML データが監査ログに記録されます。 • *[フォーム]* は、生成するデジタル署名宣言を含む PDF ドキュメントを示します。このドキュメントがユーザ合意として使用されます。ユーザは、要求の送信や承認前に、生成された PDF ドキュメントをプレビューできます。［フォーム］を選択した場合、PDF ドキュメント (XML 内にカプセル化) が監査ログに記録されます。 *[デジタル署名が必要]* **チェックボックスを選択した場合は、デジタル署名確認文字列も指定する必要があります。まず、［宣言］リストボックスの** *[ID を作成]* を選択して、文字列の識別子を作成します。次に ID を選択し、*[宣言]* アイコンをクリックします。  署名確認文字列を入力して、*[OK]* をクリックします。

フィールド	説明
承認者条件	定数の承認者条件を指定します。 アクティビティの承認者タイプが定数をサポートするように設定されている場合、承認者条件にアクティビティを承認するために必要な承認者数を設定できます。条件は数値定数、または承認者の割合 (パーセント) で指定できます。 たとえば、全体の 25% を条件とする場合は、承認者条件に 25% を指定します (下を参照)。  また、2 人の承認者からの承認を必要とする場合は、承認者条件に 2 を指定します。 **注 :** アクティビティの定数サポートを有効にした場合、アクティビティの再試行設定は指定できません。そのため、[再試行エスカレーションメモ電子メール]、[再試行回数]、[再試行間隔]、および [再試行宛先] フィールドは表示されません。
Addressee Expression	アクティビティの宛先を指定するダイナミックな式を指定します。名宛人は、式の値が求められる方法に基づいて、ランタイム時に特定されます。 名宛人を \'95\'5c す式の最初の用語は、次の値のいずれかになります。 ◆ Initiator ◆ 受信者 ◆ *アクティビティ名*の宛先 ワークフロー内の各アクティビティに対して、[式] ドロップダウンリストに個別の*アクティビティ名*の宛先が表示されます (現在設定中のアクティビティを除く)。activity-name *は、アクティビティについて指定した \'95\'5c 示ラベルです。\'95\'5c 示ラベルを指定しなかった場合は、デフォルト名となります。* 名宛人を \'95\'5c す式の 2 番目の用語は、次の値のいずれかになります。 ◆ \'83\'7d ネージャ ◆ <No attribute> Manager 属性は、抽象化層のユーザエンティティで定義済みの属性であるため、自動的に使用できます。他の属性 (Manager 以外) は、次の条件を満たす場合に、選択できます。 ◆ 抽象化層の User エンティティで定義されている ◆ 単一値である ◆ DN データタイプを持つ

フィールド	説明
Addressee DN	ユーザ、グループ、またはタスクグループの識別名を指定します。 **注 :** ユーザアプリケーションの［My Team Tasks (マイチームのタスク)］アクションで、タスクグループマネージャがタスクグループでタスクを検索できるようにする場合は、そのタスクグループを宛先として指定する必要があります。
Timeout	動作が処理を完了するために割り当てられる時間を指定します。タイムアウト間隔は、動作に対して許可した時間の合計であり、各再試行に対して許可した時間ではありません。 各動作の［タイムアウト］の設定は、［再試行回数］および［再試行間隔］の値よりも優先されます。したがって、1 つ以上の再試行を実行する前に動作の［タイムアウト］の設定に達した場合、その動作は、再試行を実行することなく処理を停止します。たとえば、タイムアウトを 10 分に設定し、3 回の再試行を再試行間隔 5 分で定義した場合、動作は一部の再試行を実行することなく、10 分後には終了します。この例の場合、2 回目の再試行はキャンセルされます。動作が終了すると、ワークフローエンジンは、Designer の最終タイムアウトアクションで定義されているリンクに従います。 値の単位は、ミリ秒、秒、分、時間、または日です。
再試行エスカレーションメモ電子メール	アクションが必要なアクティビティの現在の宛先に対して、メモ電子メールメッセージを送信するかどうかを指定します。このオプションを有効にするには、チェックボックスを選択してください。 このアクティビティの再試行エスカレーションメモ電子メールの設定を変更するには、*［このアクティビティの再試行メモ電子メールを編集します］* アイコンをクリックして、エスカレーションメモ通知を設定します。次の項目を設定します。 ◆ *［開始］* には、最初のメモを送信する時期を指定します。値は、再試行割り当て時刻からのオフセットになります。 ◆ *［間隔］* には、最初のメモを送信した後の、メモの送信間隔を指定します。 ◆ *［電子メールテンプレート］* には、メモ電子メールメッセージのテンプレートとして使用する、言語に依存しない名前を指定します。テンプレート名を指定すると、通知エンジンによりランタイム時に使用する言語固有のテンプレートが判断されます。 言語に依存しないテンプレートには、任意の名前を指定できます。リマインダ電子メールメッセージのデフォルトテンプレート名を次に示します。 Provisioning Reminder 言語固有版の各テンプレートには、言語コードに対応したサフィックスを付ける必要があります (たとえば、フランス語の場合は _fr、スペイン語の場合は _ex)。

フィールド	説明
Retry Attempts	再試行間隔に指定された時間が経過した場合に、アクティビティを再試行する回数を指定します。 アクティビティの再試行間隔に指定された時間が経過すると、ワークフローはアクティビティの完了を再試行します (アクティビティの再試行回数による)。各再試行時に、ワークフロープロセスはアクティビティを別のユーザにエスカレートすることができます。この場合、このアクティビティは新しい宛先 (たとえばユーザのマネージャ) に再度割り当てられ、新しい宛先ユーザがこのアクティビティの作業を実行できるようにします。最後の再試行が実行されると、ワークフローの設定内容に従い、そのアクティビティは承認済みまたは拒否としてマークされます。 各動作の［タイムアウト］の設定は、［再試行回数］および［再試行間隔］の値よりも優先されます。したがって、1 つ以上の再試行を実行する前に動作の［タイムアウト］の設定に達した場合、その動作は、再試行を実行することなく処理を停止します。たとえば、タイムアウトを 10 分に設定し、3 回の再試行を再試行間隔 5 分で定義した場合、動作は一部の再試行を実行することなく、10 分後には終了します。この例の場合、2 回目の再試行はキャンセルされます。動作が終了すると、ワークフローエンジンは、Designer の最終タイムアウトアクションで定義されているリンクに従います。
再試行間隔	宛先がタスクを完了するために割り当てられる時間を指定します。再試行間隔に指定された時間が経過した場合、必要に応じて新しい宛先にアクティビティを割り当て直したり、元の宛先で再試行することができます。再割り当てを設定するには、宛先の式を使用します。

フィールド	説明
Retry Addressee Expression	タイムアウト制限に到達した場合に、タスクを取得するユーザを特定するためのダイナミックな式を指定します。 再試行名宛人は、式の値が求められる方法に基づいて、ランタイム時に特定されます。 名宛人を \'95\'5c す式の最初の用語は、次の値のいずれかになります。 ◆ Initiator ◆ 受信者 ◆ *アクティビティ名*の宛先 ワークフロー内の各アクティビティに対して、［式］ドロップダウンリストに個別の*アクティビティ名*の宛先が表示されます (現在設定中のアクティビティを含む)。activity-name *は、アクティビティについて指定した \'95\'5c 示ラベルです。\'95\'5c 示ラベルを指定しなかった場合は、デフォルト名となります。* 宛先の式の 2 番目の部分は、データ抽象化層の定義内容によって異なります。たとえば、次の値が表示されます。 ◆ \'83\'7d ネージャ ◆ *グループ* ◆ ダイレクトレポート ◆ <No attribute> ［マネージャ］を選択した場合、各再試行は、組織内の上位レベルにある新しいマネージャにエスカレートされます。このため、再試行回数は組織に適した数値に設定してください。どのような場合でも、再試行回数は、現在の名宛人の上にある管理階層のレベルの数を超えてはなりません。
Retry Addressee DN	再試行制限に到達した場合に、タスクを取得するユーザまたはグループの識別名を指定します。

6 アクティビティの設定が完了したら、ページをスクロールして、フローの他のアクティビティを確認することが必要になる場合もあります。

7 ［*次*へ］をクリックします。

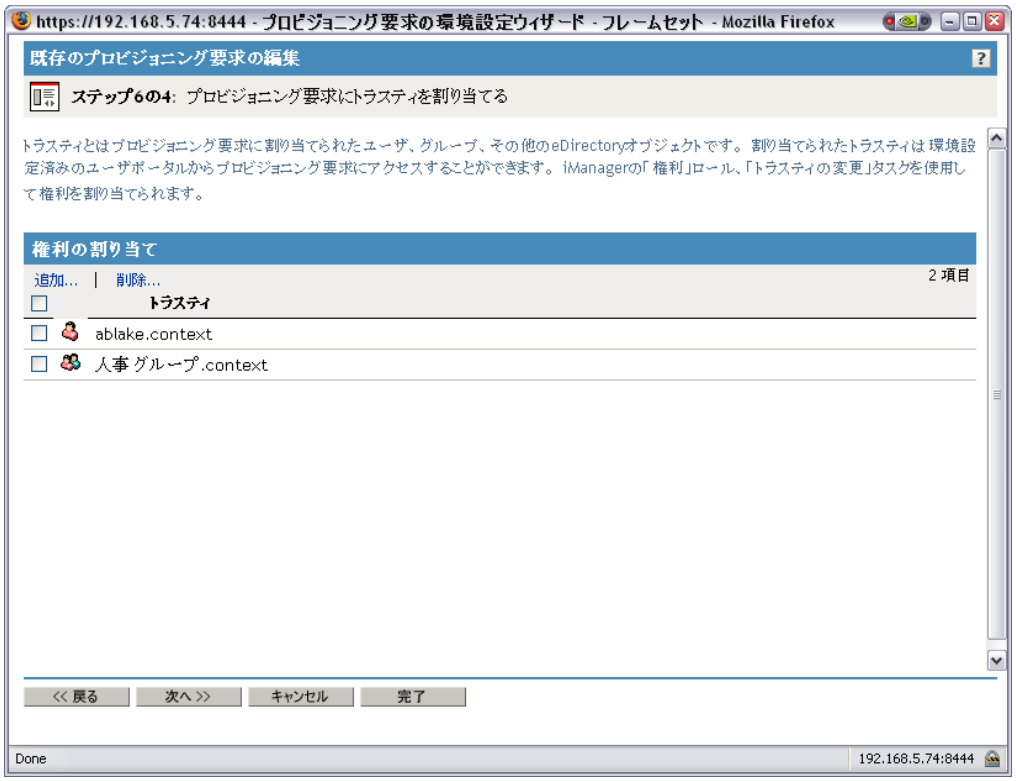
注 : 設定できるアクティビティ数は、この定義を作成する基盤として使用されたプロビジョニング要求定義によって異なります。エンタイトルメントパラメータの数およびタイプは、リクエストに関連付けられているプロビジョニングされたリソースによって異なります。

プロビジョニング要求のアクセス権の指定

プロビジョニング要求のアクセス権を指定する

1 リクエスト定義のトラスティのリストに、ユーザ、グループ、または別の eDirectory™ オブジェクトを追加するには、*［追加］*をクリックしてオブジェクトを選択します。

オブジェクトを追加すると、それがトラスティのリストに表示されます。



2 ユーザ、グループまたは他のオブジェクトを削除するには、［トラスティ］リストから項目を選択し、*［削除］*をクリックします。

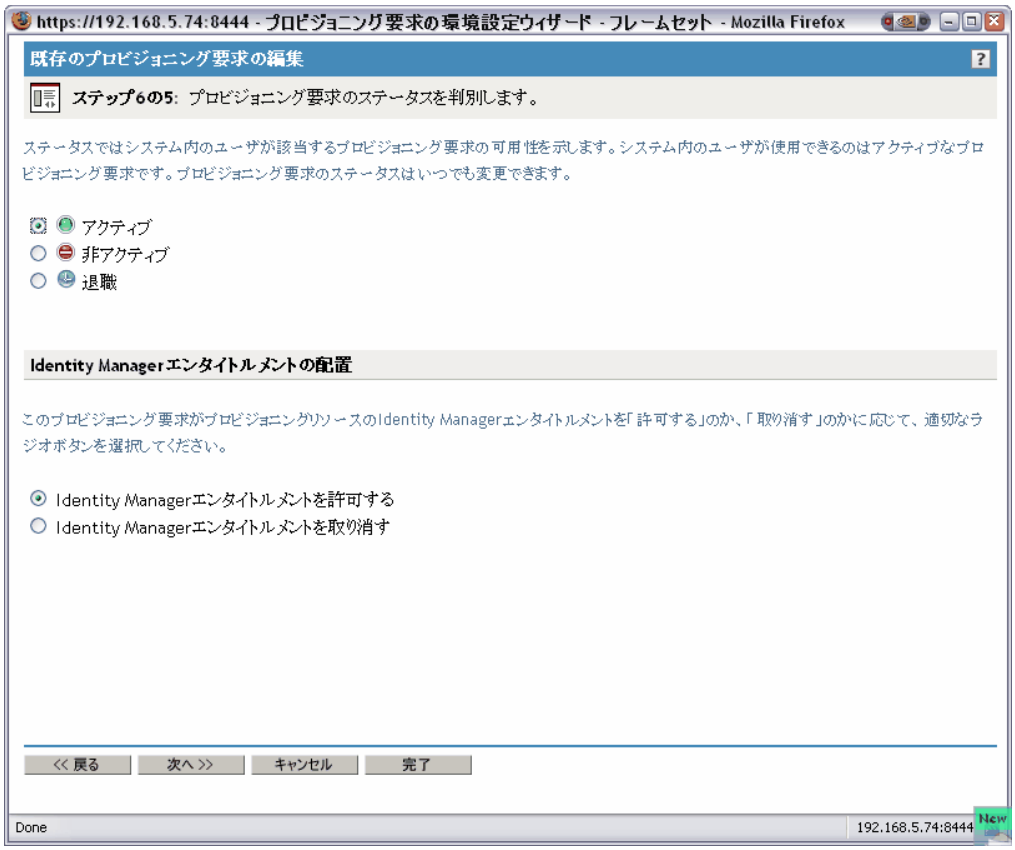
3 ［*次へ*］をクリックします。

プロビジョニング要求の初期ステータスの指定

プロビジョニング要求の初期ステータスを設定する

1 対象のステータスのボタンをクリックします。

ステータス	説明
Active	使用できます。
Inactive	一時的に使用不可になっています。これがデフォルトの設定です。
Retired	無効になっています。



2 正しいアクションに対応するボタン (*付与*または*取り消し*) をクリックします。

3 ［*次へ*］をクリックします。

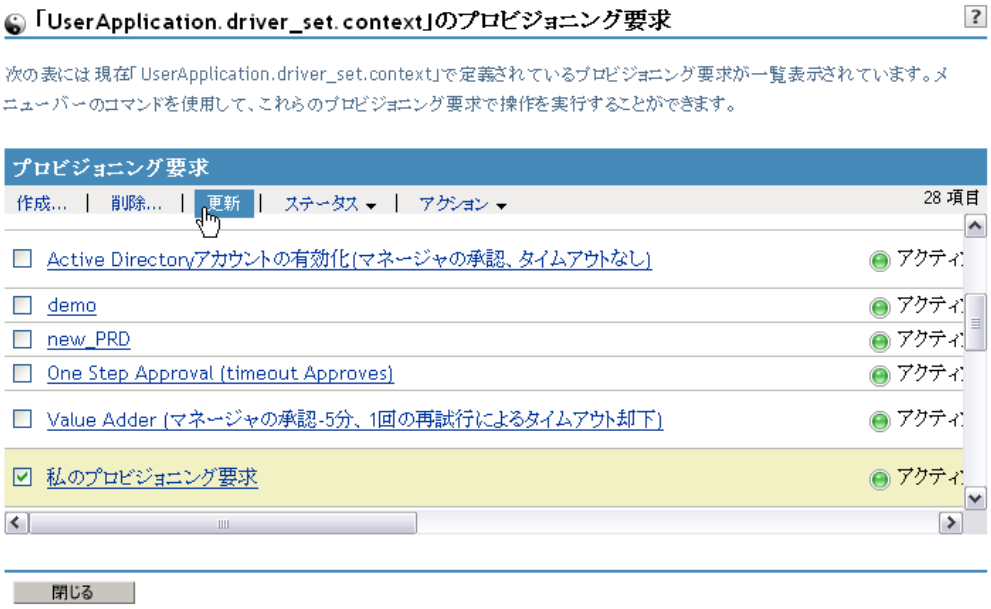
17.3.3 プロビジョニング要求の削除

プロビジョニング要求を削除する

1 名前の横にあるチェックボックスを選択し、削除するプロビジョニング要求を選択します。

テンプレートとなっているプロビジョニング要求を削除することはできません。

2 ［Provisioning Request Configuration］パネルの［*Delete*］コ *\'83\'7d* ンドをクリックします。



17.3.4 既存のプロビジョニング要求のステータスの変更

既存のプロビジョニング要求のステータスを変更する

1 名前の横にあるチェック \'83\'7bックスをオンにし、ステータスを変更するプロビジョニング要求を選択します。

2 ［Provisioning Request Configuration］パネルの［*Change Status*］コ \'83\'7d ンドをクリックします。

3 *［ステータス］*メニューのステータスをクリックします。

ステータス	説明
Active	使用できます。

ステータス	説明
Inactive	一時的に使用不可になっています。
Retired	無効になっています。

4 正しいアクションに対応するボタン (*付与*または*取り消し*) をクリックします。

5 ［*完了*］をクリックします。

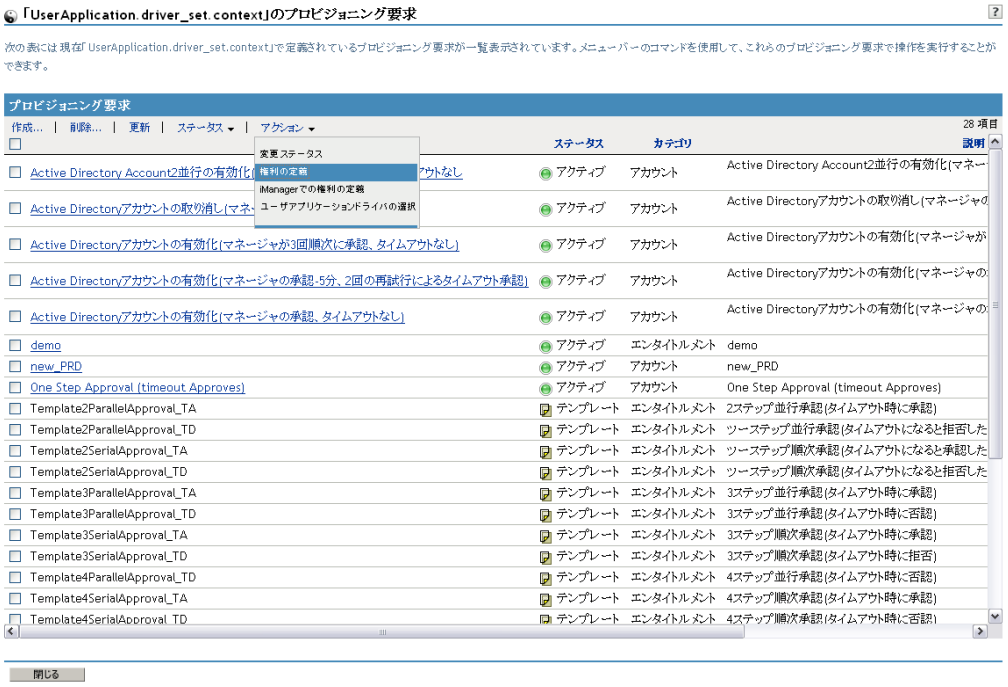
17.3.5 既存のプロビジョニング要求の権利の定義

既存のプロビジョニング要求の権利を定義する

1 名前の横にあるチェック \'83\'7bックスをオンにし、権利を定義するプロビジョニング要求を選択します。

2 ［Provisioning Request Configuration］パネルの［*Actions*］コ \'83\'7d ンドをクリックします。

3 ［*アクション*］メニューの［*権利の定義*］コマンドをクリックします。



4 の手順に従ってください。322 ページの 「プロビジョニング要求のアクセス権の指定」

iManager でプロビジョニング要求の権利を定義する

1 名前の横にあるチェック \'83\'7bックスをオンにし、権利を定義するプロビジョニング要求を選択します。

2 ［Provisioning Request Configuration］パネルの［*Actions*］コ \'83\'7d ンドをクリックします。

3 ［*アクション*］メニューの［*iManager での権利の定義*］コマンドをクリックします。

18 プロビジョニングワークフローの管理

この節では、ランタイム時のプロビジョニングワークフローの管理について説明します。プロビジョニングワークフローの電子メール通知の設定についても説明します。

主なトピックは次のとおりです。

18.1 ワークフロー管理プラグインについて

iManager でワークフロー管理プラグインを使用すると、ブラウザベースのインタフェースを使用して、ワークフロープロセスのステータスを表示したり、ワークフロー内のアクティビティを再割り当てしたり、応答がなく再開始できないワークフローを終了したりすることができます。

ワークフロー管理プラグインは、iManager の［Identity Manager］カテゴリ内にあります。このプラグインでは、*［ワークフロー管理］*役割に *［ワークフロー］*タスクが含まれています。

［ワークフロー管理］役割には、*［電子メールテンプレート］*および *［電子メールサーバオプション］*の各タスクが含まれています。これらのタスクは、［Passwords］役割にリストされている他のタスクへのショートカットとなります。

ワークフロータスクは、表 18-1 に記載されているパネルから成り立っています。

表 18-1 *ワークフロータスク : パネル*

パネル	説明
Workflows	プロビジョニングワークフローを管理するプライマリユーザインタフェースを提供します。このインタフェースには、現在処理中のワークフローが一覧表示され、これらのワークフローに対してさまざまなアクションを実行できます。 ［ワークフロー］タスクを開始すると、［ワークフロー］パネルにより、Identity Manager ユーザアプリケーションドライバを選択するよう要求されます。ドライバは、ワークフローサーバを指しています。サーバにログインしてワークフロー管理を開始する前に、ドライバを選択する必要があります。 ドライバを選択すると、管理するワークフローを選択するための検索条件を指定できます。
Workflow Detail	特定のワークフローについての詳細についての読み込み専用ユーザインタフェースを、\'95\'5c 示目的で提供します。

18.2 ワークフローの管理

この節では、ワークフロー管理プラグインを使用したプロビジョニングワークフローの管理手順について説明します。

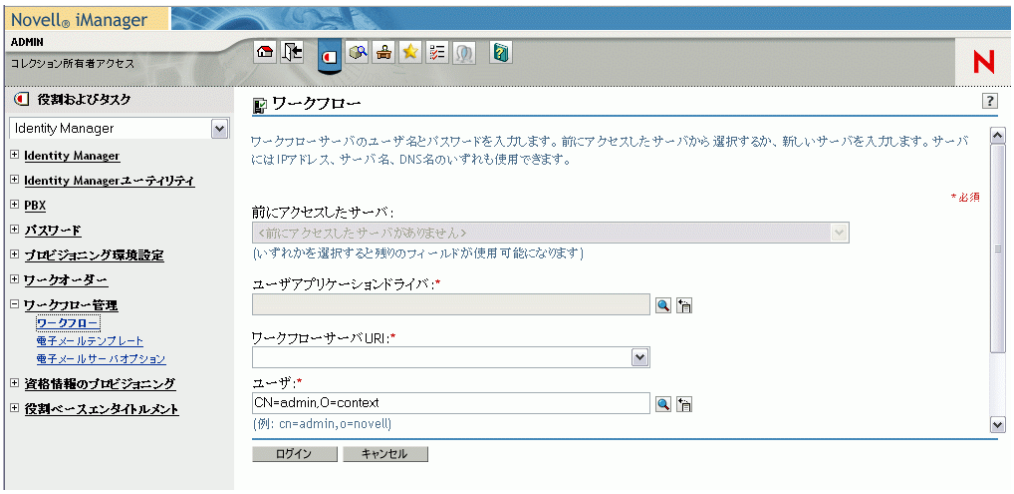
18.2.1 ワークフローサーバへの接続

ワークフロー管理を開始する前に、ワークフローサーバに接続する必要があります。ユーザアプリケーションドライバが 1 つのワークフローサーバに関連付けられている場合は、使用するドライバの名前を指定するだけで済みます。ドライバが複数のワークフローサーバに関連付けられている場合は、ターゲットとなるワークフローサーバを選択する必要があります。

ワークフローサーバに接続する

1. iManager で、［Identity Manager］カテゴリを選択します。
2. ［Workflow Administration］*役割を開きます。*
3. *［ワークフロー］*タスクをクリックします。

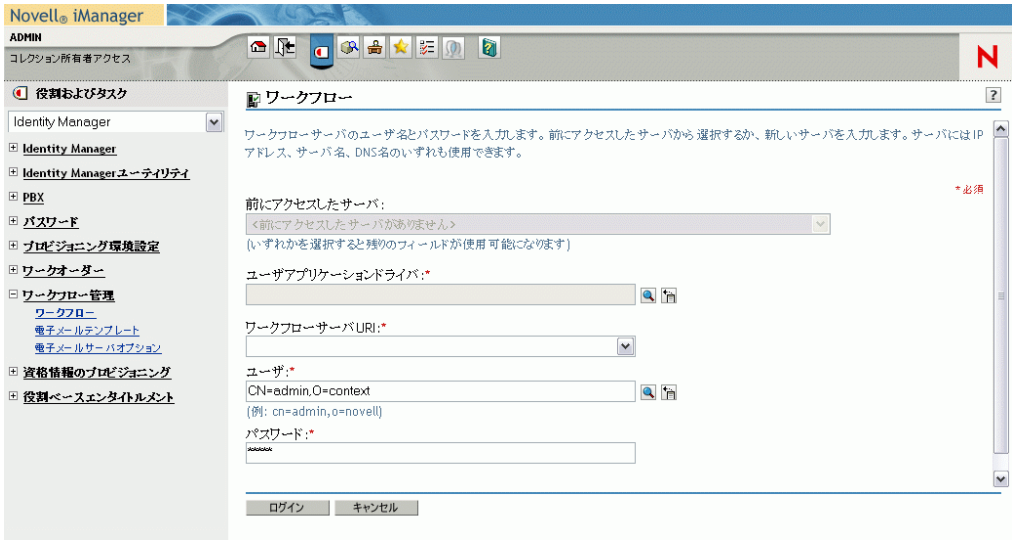
   ［ワークフロー］パネルが表示されます。



4. 以前にターゲットワークフローサーバにアクセスしたことがある場合は、*［前にアクセスしたサーバ］*ドロップダウンリストからサーバを選択できます。

   パネル内の残りのフィールドは、iManager が入力します。
5. ワークフローサーバにアクセスしたことがない場合は、*［ユーザアプリケーションドライバ］*フィールドでドライバ名を指定してから *［OK］*をクリックします。

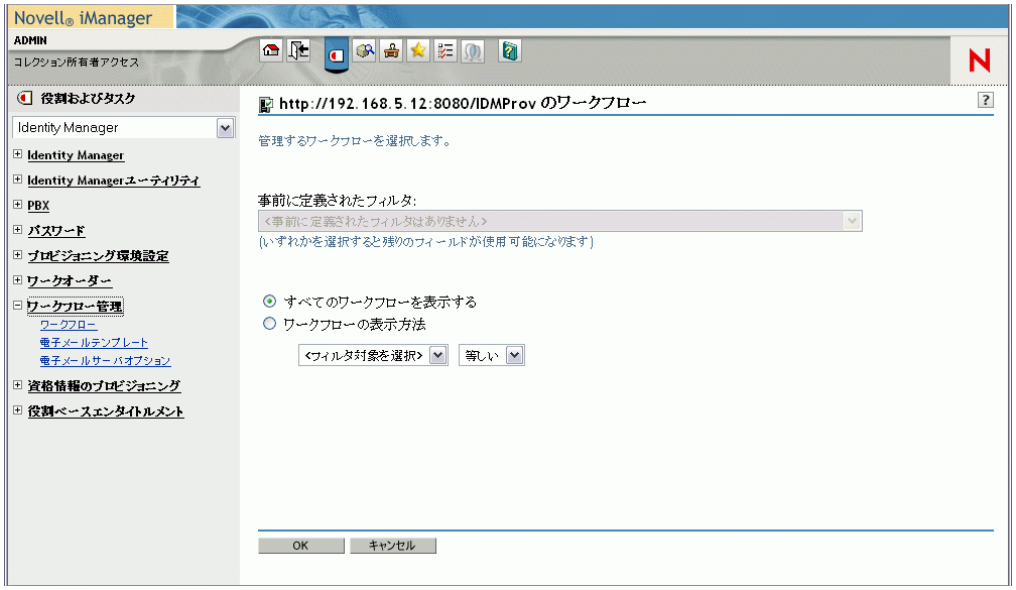
[ワークフローサーバ URI] および *[ユーザ]* フィールドは、iManager が入力します。



6 *[パスワード]* フィールドに、ユーザのパスワードを入力します。

7 *[Login]* をクリックします。

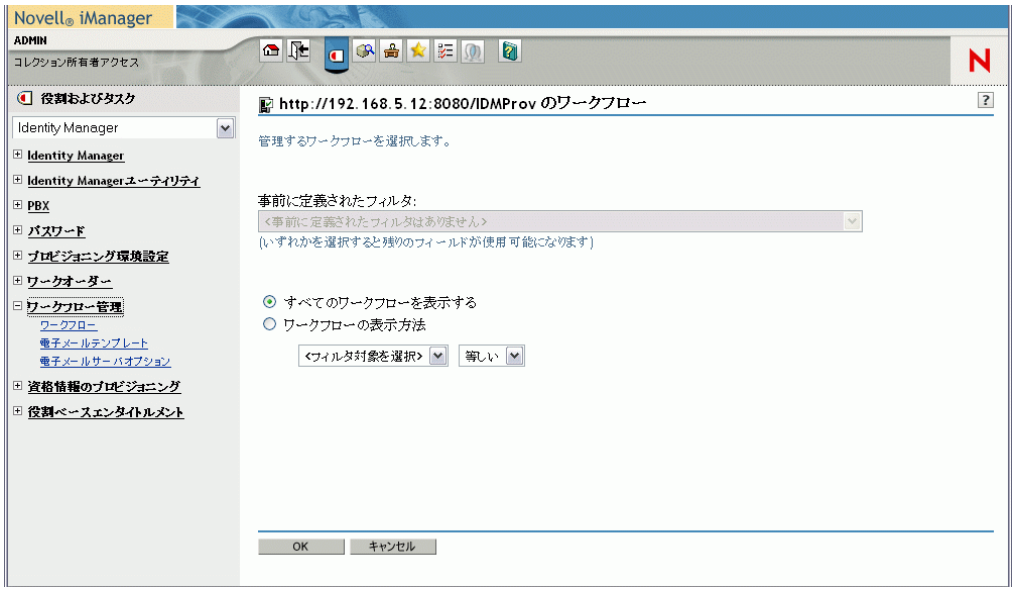
Workflow Administration プラグインに、ワークフローを検索するためのフィルタを指定するページが \'95\'5c 示されます。



18.2.2 検索条件に合致するワークフローの検索

ターゲットとなるワークフローサーバで多数のワークフロープロセスが実行されている場合、iManager でワークフローのリストのフィルタを行う必要があることがあります。フィルタを実行するには、検索条件を指定します。

1 *[ワークフローの表示方法]* を選択します。



デフォルトでは、*[すべてのワークフローを表示する]* が選択されています。サーバ上のワークフローの完全なリストを \'95\'5c 示する場合は、このデフォルトを変更しないでください。

2 条件を指定する属性を選択します。

属性	説明
Creation time	ワークフローが開始される時刻です。
Initiator	要求者のユーザ名です。
受信者	受信者のユーザ名です。
Process Status	ワークフロープロセス全体のステータスです (Completed、Running、または Terminated)。
Approval status	承認プロセスのステータスです (Approved、Denied、または Retracted)。
Entitlement status	プロビジョニング要求により開始されるエンタイトルメントのステータスです (Error、Fatal、Success、Unknown、または Warning)。

3 演算子を選択します :

演算子	コメント
Equals	すべての属性をサポートします。
Before	Creation time 属性のみをサポートします。
After	Creation time 属性のみをサポートします。
Between	Creation time 属性のみをサポートします。

4 属性および演算子の下のフィールドに値を指定します。

*［作成時刻］*については、*日付コントロールと時刻コントロール*を使用して値を選択します。*［イニシエータ］*および*［受信者］*については、*［オブジェクトの履歴］*または*［オブジェクトセレクタ］*を使用して値を指定します。他のすべての属性については、ドロップダウンリストから値を選択します。

5 *［OK］*をクリックします。

［ワークフロー］パネルで選択したワークフローが表示されます。

ターゲットサーバおよびフィルタの変更 ワークフローサーバを選択すると、新しいサーバを選択しない限り、iManager セッションの間中、選択したサーバが有効となります。新しいサーバを選択するには、*［アクション］*コマンドをクリックし、*［アクション］*メニューから*［サーバの選択］*を選択します。

別の検索条件を指定するには、［*Actions*］メニューから［Define Filter］を選択します。

18.2.3 アクティブなワークフローの表示の制御

［ワークフロー］パネルには、指定した検索条件に合致するワークフローが一覧表示されます。このリストのフィルタに加え、表示方法を制御することもできます。たとえば、リストを更新する頻度、および特定の列によってリストを \'83\'5cートする頻度を指定できます。

ワークフローのリストの更新

ワークフローサーバの動作が活発な場合、アクティブワークフローのリストは頻繁に変更されます。このような場合、サーバで実行されるアクティブワークフローのリストを更新する必要があります。

1 ［Workflows］パネルの［Refresh*］コ \'83\'7d ンドをクリックします。*

2 *［更新］*メニューから次のオプションのいずれかを選択して、更新間隔を指定します。

- 更新しない
- 今すぐ更新する
- 10 秒
- 30 秒
- 60 秒
- 5 分

3 *［OK］*をクリックします。

ワークフローのリストのソート

多数のリクエスト定義が存在する場合は、*［名前］*、*［説明］*など特定の列でソートしなければならない場合があります。

1 ソートする列の見出しをクリックします。

18.2.4 ワークフローインスタンスの終了

ワークフローインスタンスの処理を続行しない場合は、ワークフローを終了できます。

1 ワークフロー名の横にあるチェック \'83\'7bックスをオンにすることにより、［Workflows］パネルのワークフローを選択します。

2 ［Workflows］パネルの［Terminate］コ *\'83\'7d* ンドをクリックします。

18.2.5 ワークフローインスタンスの詳細の表示

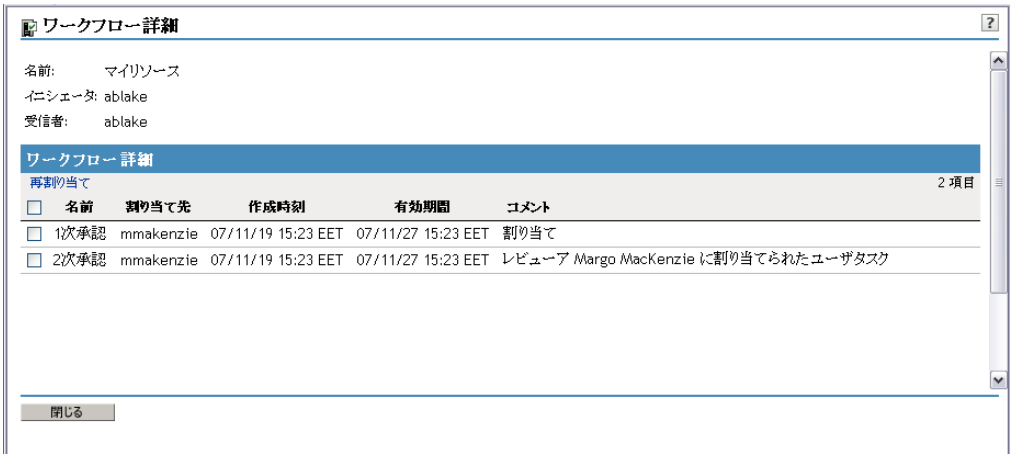
特定のサーバ上で実行中のワークフローのセットを表示したら、ワークフローインスタンスを選択して、実行中のプロセスについての詳細を表示することができます。

注 : ワークフローインスタンスがシリアル処理の設計パターンを使用している場合、1 つのアクティビティが現在のアクティビティとして表示されます。これは、その作業アイテムを一度に実行できる 1 人のユーザに限定されるためです。一方、ワークフローがパラレル処理およびブランチ処理に対応している場合、ワークフローインスタンスとして複数の現在のアクティビティが存在することがあります。

特定のワークフローインスタンスについての詳細を \'95\'5c 示する

1 ［Workflows］パネルのワークフローインスタンスの名前をクリックします。

［Workflow Detail］パネルが \'95\'5c 示されます。

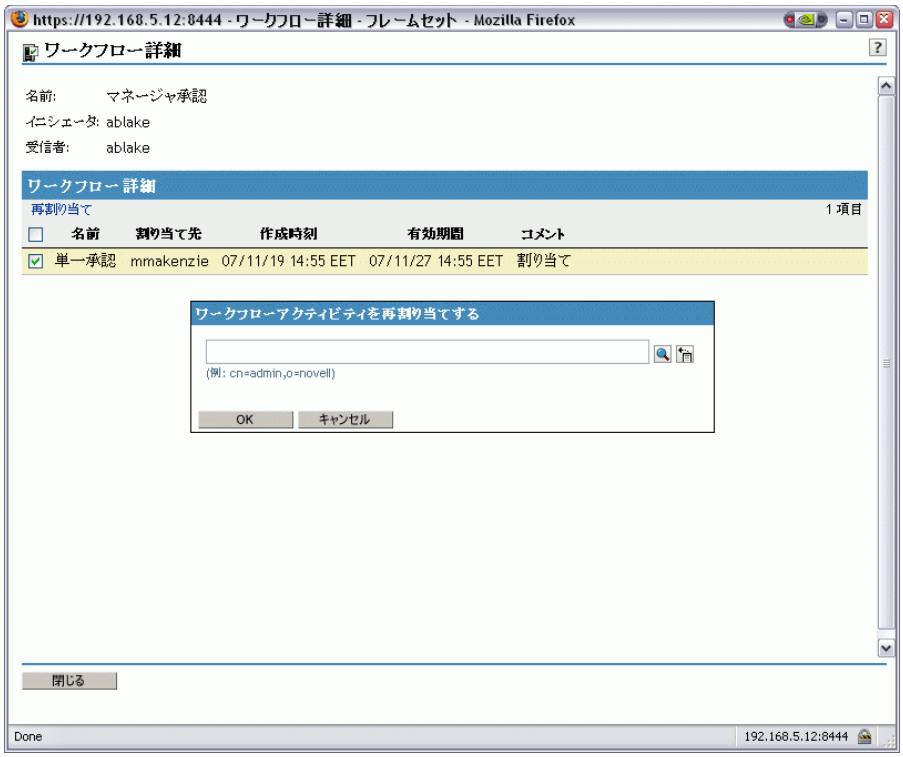


18.2.6 ワークフローインスタンスの再割り当て

ワークフローインスタンスが停止して、再開できない場合は、その作業アイテムを他のユーザ／グループに再割り当てすることができます。

1 ［Workflow Detail］パネルの名前の横にあるチェック \'83\'7bックスをオンにし、ワークフローに関連付けられている現在のアクティビティを選択します。

2 ［Workflow Detail］パネルの［Reassign］コ *\'83\'7d* ンドをクリックします。



3 作業アイテムの再割り当てを行うユーザまたはグループを選択します。

18.2.7 クラスタ内のワークフロープロセスの管理

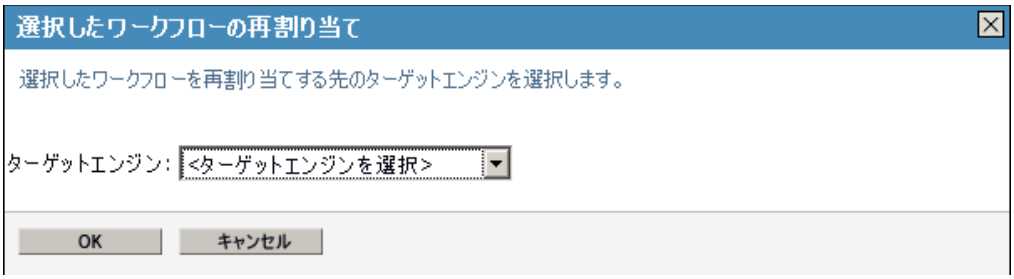
ワークフロー画面では、あるワークフローエンジンから別のワークフローエンジンに、プロセスを再割り当てすることができます。たとえば、ワークフローエンジンが回復したときに失敗したワークフローエンジンにプロセスを再割り当てしたり、クラスタからエンジンが削除されたときに他のエンジンにプロセスを再配布する場合などにこの機能を使用します。

ソースエンジンの状態は、SHUTDOWN または TIMEDOUT でなければなりません。エンジンに再割り当てされたプロセスを再開するには、ターゲットエンジンを再起動する必要があります。

あるワークフローエンジンから別のワークフローエンジンへのプロセスの再割り当て

1 [ワークフロー] パネルで、再割り当てするワークフロー名の隣にあるチェックボックスを選択します。

2 *[アクション] > [再割り当て]* を選択します。



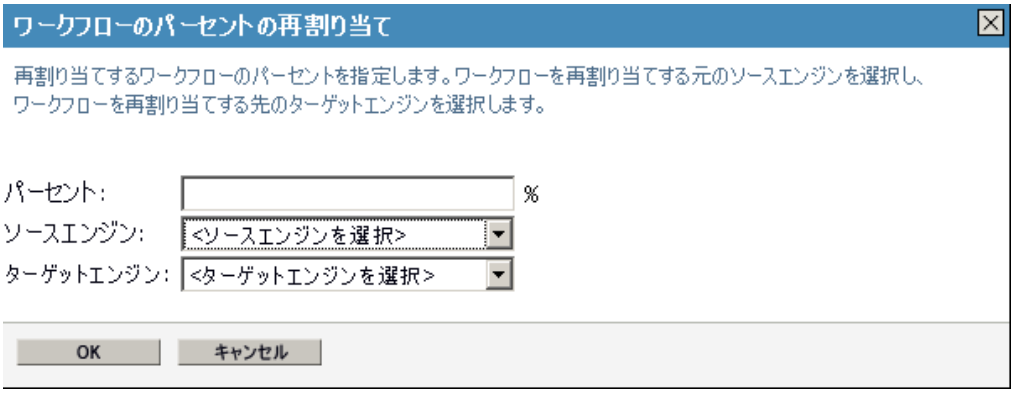
3 *[ターゲットエンジン]* リストから、プロセスを再割り当てするワークフローエンジンを選択します。

4 *[OK]* をクリックします。

あるワークフローから別のワークフローへのパーセント指定によるプロセスの再割り当て

1 [ワークフロー] パネルで、再割り当てするワークフロー名の隣にあるチェックボックスを選択します。

2 *[アクション]* > *[パーセントの再割り当て]* を選択します。



3 *[パーセント]* フィールドに、別のワークフローエンジンに再割り当てするワークフロープロセスのパーセントを入力します。

4 再割り当てするプロセスがあるワークフローエンジンを選択するには、*[ソースエンジン]* リストを使用します。

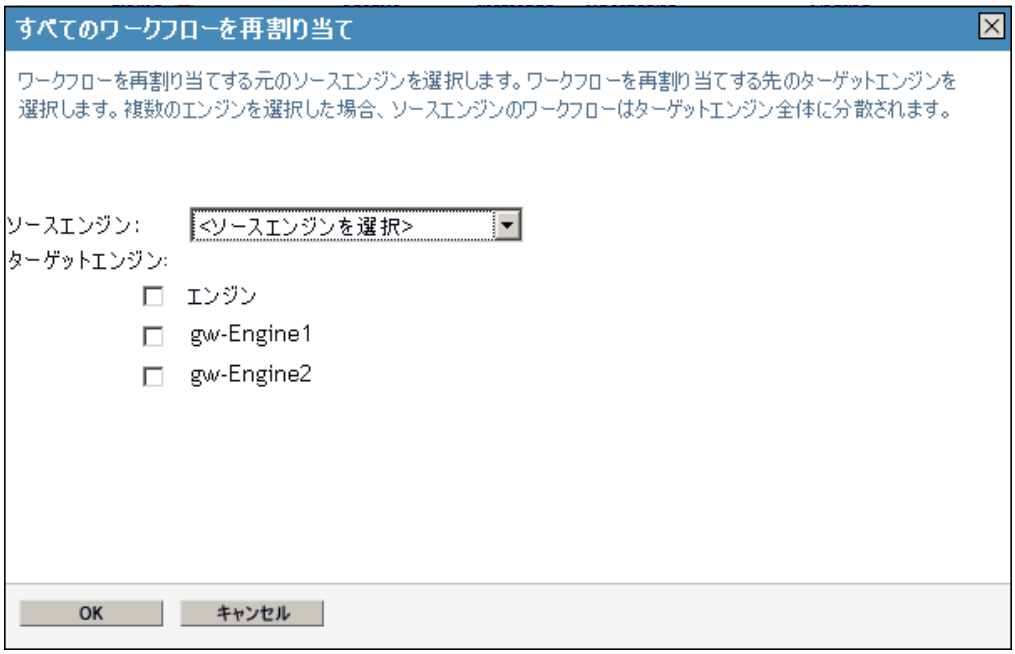
5 プロセスの再割り当て先ワークフローエンジンを選択するには、*[ターゲットエンジン]* リストを使用します。

6 *[OK]* をクリックします。

あるワークフローエンジンから別のワークフローエンジンへのすべてのプロセスの再割り当て

1 [ワークフロー] パネルで、再割り当てするワークフロー名の隣にあるチェックボックスを選択します。

2 *[アクション]* > *[すべて再割り当て]* を選択します。



3 再割り当てするプロセスがあるワークフローエンジンを選択するには、*［ソースエンジン］*リストを使用します。

4 プロセスを再割り当てするワークフローエンジン名の隣にあるチェックボックスを選択します。

複数のターゲットエンジンを選択した場合、ソースエンジンのプロセスは各ターゲットエンジンに均等に分配されます。

5 *［OK］*をクリックします。

18.3 電子メールサーバの設定

ワークフローシステムは、通常、実行中のさまざまなポイントで電子メール通知を送信します。たとえば、ワークフローアクティビティが新しい名宛人に割り当てられる場合、電子メールが送信されることがあります。

Identity Manager の電子メール通知機能を使用する前に、SMTP 電子メールサーバを設定する必要があります。設定するには、iManager の［*Workflow Administration*］*役割にある*［Email Server Options］タスクを使用します。

注 : このタスクは、［*Passwords*］*役割の*［Email Server Options］タスクへのショートカットとなります。

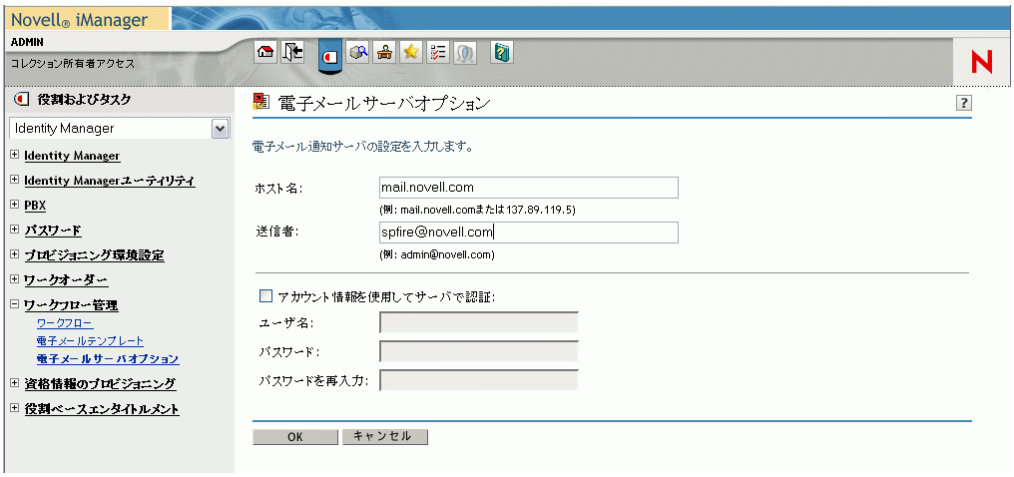
電子メールサーバを設定する

1 iManager で、［Identity Manager］カテゴリを選択します。

2 ［Workflow Administration］*役割を開きます。*

3 ［Email Server Options］*タスクをクリックします。*

［電子メールサーバオプション］パネルが表示されます。



4 ［*Host Name*］フィールドに、ホストサーバの名前 (または IP アドレス) を入力します。

5 ［From］フィールドに、送信者の電子メールアドレスを入力します。

受信者が電子メールを開くと、このテキストが電子メールの見出しの［*差出人*］フィールドに表示されます。メールサーバの設定に応じて、メールサーバに逆ルックアップや認証を行わせるには、このフィールド中の文字列を、システムの有効な送信者と一致させなければならないことがあります。たとえば、「パスワード管理者」などの説明的なテキストではなく、「helpdesk@company.com」と指定します。

6 サーバの電子メール送信前に認証を必要とする場合は、［*アカウント情報を使用してサーバで認証*］チェックボックスをオンにし、ユーザ名およびパスワードを入力します。

7 完了したら、［*OK*］をクリックします。

18.4 電子メールテンプレートに関する作業

Identity Manager には、ワークフローベースのプロビジョニング用に設計された電子メール通知テンプレートが用意されています。これらの電子メールテンプレートを次に示します。

- *新規プロビジョニング要求* (Provisioning Notification: プロビジョニング通知)
- *可用性設定の通知* (Availability: 可用性)
- *委任割り当ての通知* (Delegate: 委任)
- *プロビジョニング承認通知* (Provisioning Approval Completed Notification: プロビジョニング承認完了通知)
- *リマインダ - 承認を待っている要求* (Provisioning Reminder: プロビジョニングリマインダ)
- *代理人割り当ての通知* (Proxy: 代理人)

上記の各項目は、件名を最初に表示しています。テンプレート名はかっこで囲んで表示しています (iManager や Designer に表示)。

テンプレートの内容を変更したり、電子メールメッセージの書式を設定することができます。また、新しいテンプレートを作成することもできます。新しいテンプレートを作成する場合は、次の命名規則に従う必要があります。

- 言語に依存しないバージョンのプロビジョニング通知テンプレートには、任意の名前を指定できます。通知電子メールメッセージのデフォルトテンプレート名を次に示します。

  Provisioning Notification

- 言語に依存しないバージョンのプロビジョニングリマインダテンプレートには、任意の名前を指定できます。リマインダ電子メールメッセージのデフォルトテンプレート名を次に示します。

  Provisioning Reminder

- 委任テンプレートには、次の単語で始まる名前を指定する必要があります。

  委任

  言語に依存しない名前の後には、テンプレートの目的や内容を表す数文字を追加することができます。

- 代理テンプレートには、次の単語で始まる名前を指定する必要があります。

  代理

  言語に依存しない名前の後には、テンプレートの目的や内容を表す数文字を追加することができます。

- 可用性テンプレートには、次の単語で始まる名前を指定する必要があります。

  空き状況

  言語に依存しない名前の後には、テンプレートの目的や内容を表す数文字を追加することができます。

言語固有版の各テンプレートには、言語コードに対応したサフィックスを付ける必要があります (たとえば、フランス語の場合は _fr、スペイン語の場合は _ex)。

電子メールテンプレートを作成するには、iManager の［ワークフロー管理］役割の *［電子メールテンプレート］* を使用します。

注 : このタスクは、［*Passwords］役割の［*Edit Email Templates］タスクへのショートカットとなります。

また、Designer で電子メールテンプレートを作成、編集することもできます。

iManager または Designer でユーザアプリケーションドライバを作成する場合、標準の電子メール通知テンプレートセットから失われている電子メール通知テンプレートは置換されます。既存の電子メール通知テンプレートは更新されません。これは、カスタマイズされた電子メール通知テンプレートへの上書きを防止するためです。Designer を使って既存の電子メール通知テンプレートを手動で更新することができます (『*Identity Manager 3.5.1 ユーザアプリケーション : 設計ガイド*(http://www.novell.com/documentation/idm35/index.html)』の電子メール通知テンプレートに関する項目を参照)。電子メール通知テンプレートの詳細は、『*Novell Designer 2.1 for Identity Manager 3.5.1 管理ガイド*』の電子メール通知テンプレートの設定に関する項目を参照してください。

注 : プロビジョニング要求定義でローカライズされた電子メールテンプレートを使用する場合、通知の受信者の優先ロケールの設定は無視されます。たとえば、ローカライズされ

たスペイン語版の電子メール通知テンプレートを使ったプロビジョニング通知では、各ユーザの優先ロケールの設定に関係なく、スペイン語の電子メールのみが送信されます。

18.4.1 デフォルトのコンテンツおよび形式

この節では、製品のインストール後の電子メールテンプレートの内容について説明しています。また、電子メールテンプレートに使用できる置換タグについても説明していきます。

新規プロビジョニング要求

テンプレートは、電子メールメッセージをトリガしたプロビジョニング要求定義を識別します。承認を必要とするタスクに名宛人をリダイレクトする URL、そのユーザの保留中のタスについてクの完全なリストを \'95\'5c 示する URL も含まれています。

```
Hi,A new provisioning request has been submitted that requires your
approval.Request name: $requestTitle$Submitted by:
$initiatorFullName$Recipient: $recipientFullName$Please review the
details of this request at $PROTOCOL$://$HOST$:$PORT$/$TASK_DETAILS$
to take the appropriate action.You can review a list of all requests
pending your approval at $PROTOCOL$://$HOST$:$PORT$/
$TASKLIST_CONTEXT$.
```

表 18-2 *新規プロビジョニング要求 : 置換タグ*

タグ	説明
$userFirstName$	名宛人の名です。
$requestTitle$	プロビジョニング要求定義の \'95\'5c 示名です。
$initiatorFullName$	イニシエータの完全な名前です。
$recipientFullName$	受信者の完全な名前です。
$PROTOCOL$	電子メールメッセージに含まれる URL のプロトコルです。
$SECURE_PROTOCOL$	電子メールメッセージに含まれる URL のセキュアプロトコルです。
$HOST$	Identity Manager ユーザアプリケーションを実行する JBoss サーバのホスト。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$PORT$	Identity Manager ユーザアプリケーションのポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$SECURE_PORT$	Identity Manager ユーザアプリケーションのセキュアポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。

タグ	説明
$TASKLIST_CONTEXT$	名宛人について保留となっているすべての要求のリストを \'95\'5c 示するページです。
$TASK_DETAILS$	この電子メールメッセージが生成される要求の詳細を \'95\'5c 示するページです。

可用性設定の通知

このテンプレートは、可用性が更新されたユーザを識別します。これには、ユーザが利用不可能な状態となる期間の開始時刻と有効期限、およびユーザが利用不可能なリソースが含まれています。

```
Hi,$submitterFirstName$ $submitterLastName$ has updated availability
settings for  $userFirstName$ $userLastName$.  This user has
$operation$ an availability setting that applies to the following
resources:  $resources$  This setting indicates that $userFirstName$
$userLastName$ is unavailable to work on these resources during the
timeframe outlined below:  Start time: $startTime$  Expiration time:
$expirationTime$  When a user is unavailable, any delegates assigned
may handle resource requests for that user.You can review a list of
your availability settings at $PROTOCOL$://$HOST$:$PORT$/
$AVAILABILITY_CONTEXT$.
```

表 18-3 *可用性設定の通知 : 置換タグ*

タグ	説明
$submitterFirstName$	可用性の設定を更新したユーザの名前。
$PROTOCOL$	電子メールメッセージに含まれる URL のプロトコルです。
$PORT$	Identity Manager ユーザアプリケーションのポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$startTime$	このプロビジョニング要求のワークフローの開始時刻。
$resources$	宛先が利用不可能なリソース (プロビジョニング要求)。
$SECURE_PROTOCOL$	電子メールメッセージに含まれる URL のセキュアプロトコルです。
$expirationTime$	可用性が失効する時刻。
$submitterLastName$	可用性の設定を更新したユーザの名字。
$SECURE_PORT$	Identity Manager ユーザアプリケーションのセキュアポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。

タグ	説明
$userFirstName$	この可用性設定を適用するユーザの名前。
$userLastName$	この可用性設定を適用するユーザの名字。
$HOST$	Identity Manager ユーザアプリケーションを実行する JBoss サーバのホスト。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$ASSIGNMENT_LIST_CONTEXT$	プロビジョニングユーザアプリケーションの URL のコンテキスト／パス。

委任割り当ての通知

このテンプレートは、ユーザの承認が必要なプロビジョニング要求が送信された場合に、ユーザにその旨を通知します。これには、リクエスト名、リクエストの送信者、受信者のフルネームが含まれています。また、プロビジョニング要求を表示するためのリンクと、そのユーザの承認待ちのすべてのプロビジョニング要求を表示するためのリンクも含まれています。

```
Hi,A new provisioning request has been submitted that requires your
approval.Request name: $requestTitle$Submitted by:
$initiatorFullName$Recipient: $recipientFullName$Please review the
details of this request at $PROTOCOL$://$HOST$:$PORT$/$TASK_DETAILS$
to take the appropriate action.You can review a list of all requests
pending your approval at $PROTOCOL$://$HOST$:$PORT$/
$TASKLIST_CONTEXT$._SUBJECT
```

表 18-4 *委任割り当ての通知 : 置換タグ*

タグ	説明
$submitterFirstName$	委任を割り当てたユーザの名前。
$PROTOCOL$	電子メールメッセージに含まれる URL のプロトコルです。
$PORT$	Identity Manager ユーザアプリケーションのポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$resources$	委任が可能なリソース (プロビジョニング要求)。
$SECURE_PROTOCOL$	電子メールメッセージに含まれる URL のセキュアプロトコルです。
$fromUsers$	割り当てられた委任に、リソース要求の処理権限があるユーザ。
$relationship$	この委任割り当てで選択された、ディレクトリ抽象化層に定義されている関係。

タグ	説明
$expirationTime$	委任割り当てが失効する時刻。
$fromContainers$	割り当てられた委任にリソース要求の処理権限があるコンテナ。
$fromGroups$	割り当てられた委任にリソース要求の処理権限があるグループ。
$submitterLastName$	委任を割り当てたユーザの名字。
$SECURE_PORT$	Identity Manager ユーザアプリケーションのセキュアポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$userFirstName$	委任が割り当てられたユーザの名前。
$userLastName$	委任が割り当てられたユーザの名字。
$HOST$	Identity Manager ユーザアプリケーションを実行する JBoss サーバのホスト。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$ASSIGNMENT_LIST_CONTEXT$	プロビジョニングユーザアプリケーションの URL のコンテキスト／パス。

プロビジョニング承認通知

このテンプレートは、ユーザが送信したプロビジョニング要求の承認プロセスが完了した時に、その旨をユーザに通知します。

```
Hi,The approval process of your provisioning request has
completed.Request name: $requestTitle$Request id: $requestId$Submitted
by: $initiatorFullName$Submitted on: $requestSubmissionTime$Recipient:
$recipientFullName$Status: $requestStatus$
```

表 18-5 *プロビジョニング承認通知 : 置換タグ*

タグ	説明
$initiatorFullName$	イニシエータの完全な名前です。
$requestSubmissionTime$	要求が送信された時刻。
$requestTitle$	プロビジョニング要求定義の \'95\'5c 示名です。
$requestId	プロビジョニング要求の ID。
$recipientFullName$	受信者の完全な名前です。

リマインダ - 承認を待っている要求

このテンプレートは、ユーザの承認が必要な、キュー内にある承認待ちプロビジョニング要求をユーザに知らせます。これには、リクエスト名、リクエストの送信者、受信者が含まれています。また、プロビジョニング要求を表示するためのリンクと、そのユーザの承認待ちのすべてのプロビジョニング要求を表示するためのリンクも含まれています。

```
Hi,This is a reminder that a provisioning request is sitting in your
queue waiting on your approval.Request name: $requestTitle$Submitted
by: $initiatorFullName$Recipient: $recipientFullName$Please review the
details of this request at $PROTOCOL$://$HOST$:$PORT$/$TASK_DETAILS$
to take the appropriate action.You can review a list of all requests
pending your approval at $PROTOCOL$://$HOST$:$PORT$/
$TASKLIST_CONTEXT$.
```

表 18-6 *リマインダ - 承認を待っている要求 : 置換タグ*

タグ	説明
$TASKLIST_CONTEXT$	名宛人について保留となっているすべての要求のリストを \'95\'5c 示するページです。
$PROTOCOL$	電子メールメッセージに含まれる URL のプロトコルです。
$PORT$	Identity Manager ユーザアプリケーションのポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$SECURE_PROTOCOL$	電子メールメッセージに含まれる URL のセキュアプロトコルです。
$initiatorFullName$	イニシエータの完全な名前です。
$recipientFullName$	受信者の完全な名前です。
$TASK_DETAILS$	この電子メールメッセージが生成される要求の詳細を \'95\'5c 示するページです。
$SECURE_PORT$	Identity Manager ユーザアプリケーションのセキュアポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$userFirstName$	名宛人の名です。
$HOST$	Identity Manager ユーザアプリケーションを実行する JBoss サーバのホスト。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$requestTitle$	プロビジョニング要求定義の \'95\'5c 示名です。

代理人割り当ての通知

このテンプレートは、代理人が割り当てられた受信者にその旨を通知します。代理人として割り当てられたユーザと、ユーザが代理人として行動できる権限のあるユーザ、グループ、コンテナが指定されています。これには、受信者の代理人割り当てのリストを表示するためのリンクも含まれています。

```
Hi,A proxy assignment that authorizes a user to act as proxy forone or
more users, groups, or containers was $operation$ by:
$submitterFirstName$ $submitterLastName$.Unlike delegate assignments,
proxy assignments are independent of resource requests, and therefore
apply to all work and settings actions.The user selected as proxy
is:$userFirstName$  $userLastName$The assigned proxy is authorized to
handle all work for these users, groups, and containers:Users:
$fromUsers$Groups: $fromGroups$Containers: $fromContainers$This proxy
assignment expires at:$expirationTime$You can review a list of your
proxy assignments at $PROTOCOL$://$HOST$:$PORT$/$PROXY_CONTEXT$.
```

***表 18-7** 代理人割り当ての通知 : 置換タグ*

タグ	説明
$submitterFirstName$	代理人を割り当てたユーザの名前。
$PROTOCOL$	電子メールメッセージに含まれる URL のプロトコルです。
$PORT$	Identity Manager ユーザアプリケーションのポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$resources$	代理人が利用可能なリソース (プロビジョニング要求)。
$SECURE_PROTOCOL$	電子メールメッセージに含まれる URL のセキュアプロトコルです。
$fromUsers$	割り当てられた代理人にリソース要求の処理権限があるユーザ。
$expirationTime$	代理人割り当てが失効する時刻
$fromContainers$	割り当てられた代理人にリソース要求の処理権限があるコンテナ。
$fromGroups$	割り当てられた代理人にリソース要求の処理権限があるグループ。
$submitterLastName$	代理人を割り当てたユーザの名字。
$SECURE_PORT$	Identity Manager ユーザアプリケーションのセキュアポート。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$userFirstName$	代理人が割り当てられたユーザの名前。

タグ	説明
$userLastName$	代理人が割り当てられたユーザの名字。
$HOST$	Identity Manager ユーザアプリケーションを実行する JBoss サーバのホスト。このパラメータの値の設定については、345 ページのセクション 18.4.3「テンプレートのデフォルト値の変更」を参照してください。
$ASSIGNMENT_LIST_CONTEXT$	プロビジョニングユーザアプリケーションの URL のコンテキスト／パス。

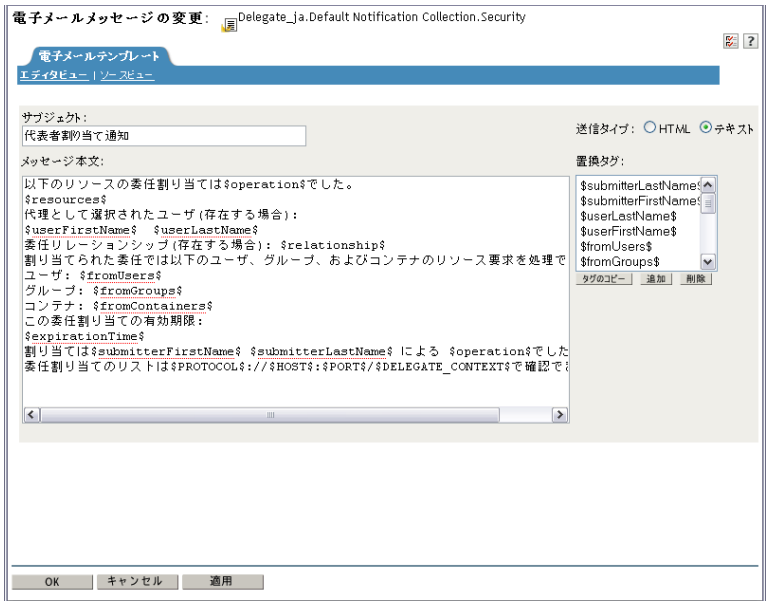
18.4.2 電子メールテンプレートの編集

用意されている電子メールテンプレートの内容や形式を変更することができます。電子メールテンプレートの作成方法については、『*Novell Identity Manager 管理ガイド*』の電子メール通知の環境設定に関する項目を参照してください。

テンプレートを編集する

1. iManager で、*［Identity Manager］*カテゴリを選択します。
2. ［Workflow Administration］*役割を開きます。*
3. *［電子メールテンプレート］*タスクをクリックします。
   *［電子メールテンプレートの編集］*パネルが表示されます。

4. 編集する電子メールテンプレート名をクリックします。
   *［電子メールメッセージの変更］*画面が表示されます。



5 [*Message Body*] \'83\'7b ックスで変更を行います。

6 必要に応じて、メッセージ本文にダイナミックテキストが含まれるように、*[置換タグ]* リストに表示されている 1 つまたは複数のタグをコピーします。

置換タグの詳細は、338 ページのセクション 18.4.1「デフォルトのコンテンツおよび形式」を参照してください。

7 完了したら、*[OK]* をクリックします。

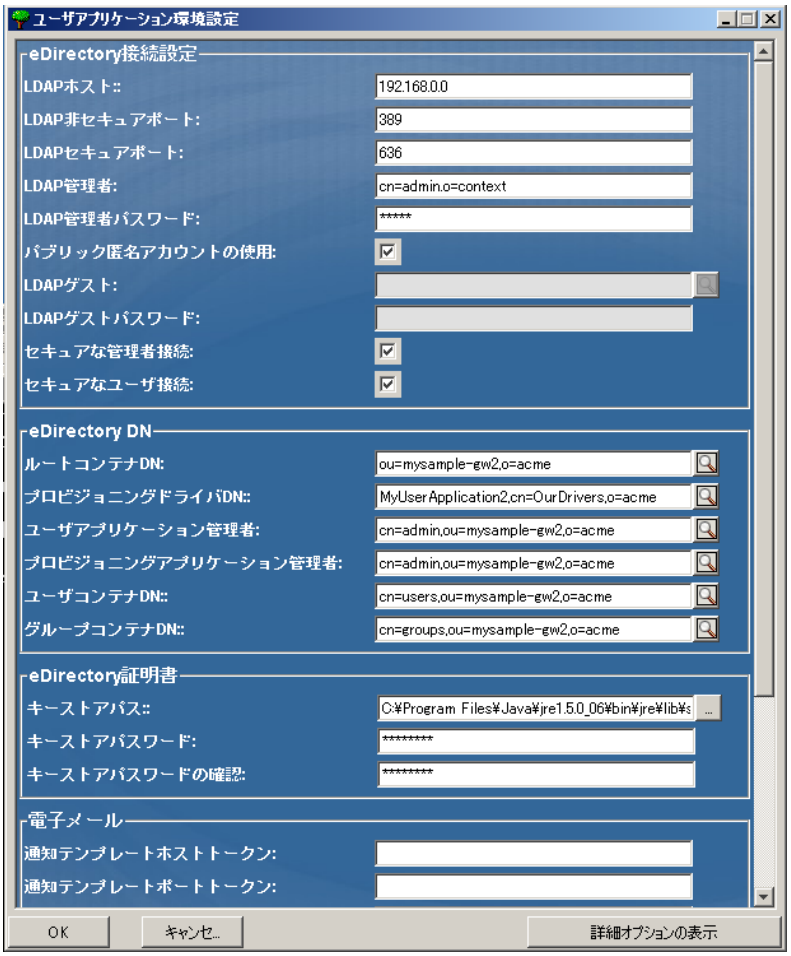
18.4.3 テンプレートのデフォルト値の変更

インストール時、電子メールテンプレートで使用するさまざまな置換タグに、デフォルト値を設定できます。インストール完了後、ユーザアプリケーションの環境設定ツールを使用して、これらの値を変更することもできます。

1 idm フォルダにある configupdate.sh スクリプトを実行します。

```
./configupdate.sh
```

Windows で、configupdate.bat を実行します。



2 必要に応じて、次のフィールドのいずれかを変更します。

フィールド	説明
Email Notify Host	承認フローで使用する電子メールテンプレートにある $HOST$ トークンの置き換えに使用します。空白のままの場合は、サーバにより計算されます
Email Notify Port	承認フローで使用する電子メールテンプレート内の $PORT$ トークンの置き換えに使用されます。
Email Notify Secure Port	承認フローで使用する電子メールテンプレート内の $SECURE_PORT$ トークンの置き換えに使用されます。

3 [OK] *をクリックして、変更を確認します。*

18.4.4 ローカライズされた電子メールテンプレートの追加

ローカライズされた電子メールテンプレートを追加する

1 iManager で、*［Identity Manager］カテゴリを選択します。*

2 ［Workflow Administration］*役割を開きます。*

3 *[電子メールテンプレート]* タスクをクリックします。

[電子メールテンプレートの編集] パネルが表示されます。

4 コピーしたい電子メールテンプレート (テンプレート名の中にロケールが含まれていないもの) を探します。

4a テンプレート名をメモしておきます (ステップ 5 で使用)。

4b そのテンプレートをクリックして開き、［件名］ボックス、［メッセージ本文］ボックス、および［置換タグ］ボックスの内容を確認します。

4c 作成したいテンプレートの中で使用する、件名、メッセージ本文、および置換タグをコピーします (件名とメッセージ本文は、作成時に翻訳する)。

4d ［キャンセル］をクリックします。? ●

5 *[作成]* ボタンをクリックし、ロケール拡張部の付いたテンプレート名を指定します。たとえば、ドイツ語版の Forgot Hint テンプレートを作成するには、「Forgot Hint_de」と入力します。末尾の「_de」はドイツ語を意味します。? ●

2 文字の言語コードと 2 文字の国コードを使用した場合、このテンプレートは正常に動作します。バリアント付きロケール (例 : en_US_TX) を使おうとした場合、バリアントと言語だけが指定されていると見なされます。電子メールテンプレートに名前を付ける場合は、ロケールバリアントは使用しないでください。

6 *[OK]* をクリックします。

7 テンプレート一覧で、新規に作成したテンプレート (例 : Forgot Hint_de) をクリックし、件名とメッセージ本文をローカライズ先言語で入力します。メッセージ本文内の、ドル記号 ($) で囲まれた置換タグは、そのまま維持してください。? ●

8 必要に応じて、メッセージ本文にダイナミックテキストが含まれるように、*[置換タグ]* リストに表示されている 1 つまたは複数のタグをコピーします。

置換タグの詳細は、338 ページのセクション 18.4.1「デフォルトのコンテンツおよび形式」を参照してください。

9 ［*適用*］をクリックします。

10 *[OK]* をクリックします。

注 : 電子メール送信先ユーザに対して優先ロケールが設定されている場合、そのロケールに合わせてローカライズされているコンテンツだけが送信されます。

プロビジョニングチームの環境設定 19

この節では、プロビジョニングチームを管理するための iManager プラグインの使用方法を説明します。主なトピックは次のとおりです。

19.1 プロビジョニングチームプラグインについて

Identity Manager ユーザインタフェースの *［要求と承認］* タブには、*［マイチームの作業］* というアクションカテゴリがあります。*［マイチームの作業］* アクションカテゴリにあるアクションを使うことにより、ワークフロー内のチームメンバーのタスクと要求を処理することができます。

プロビジョニングチームの環境設定を行うには、iManager でプロビジョニングチームおよびプロビジョニングチーム要求プラグインを使用する必要があります。プロビジョニングチームプラグインでは、チームの特徴を定義できます。プロビジョニングチーム要求プラグインでは、チームの要求権限を指定できます。

注 : 各チーム定義に対して、チーム要求オブジェクトを定義する必要があります。チーム要求オブジェクトのないプロビジョニングチームをユーザアプリケーションで使用することはできません。

プロビジョニングチームおよびプロビジョニングチーム要求プラグインは、iManager の［Identity Manager］カテゴリにあります。これらのプラグインは、［プロビジョニング環境設定］役割にあります。

19.1.1 チームの概要

*チーム*はユーザのグループを表し、このチームに関連付けられたプロビジョニング要求と承認タスクを管理できるユーザが定義されています。チーム定義は、以下で説明するように、チームマネージャ、チームメンバー、およびチームオプションのリストで構成されています。

- *チームマネージャ*とは、チームメンバーのための要求やタスクを管理できるユーザです。チームマネージャには、チームメンバーのために代理や委任を設定する権限を与えることもできます。チームマネージャにはユーザまたはグループがなることができます。
- *チームメンバー*とは、チームに参加することを許可されたユーザのことです。チームメンバーにはディレクトリ内のユーザ、グループ、またはコンテナがなることができます。または、ディレクトリ関係から生成することもできます。たとえば、メンバーのリストは組織内の「マネージャ - 従業員」の関係から生成できます。この場合のチームメンバーは、チームマネージャに報告義務のあるすべてのユーザを指します。

注 : プロビジョニングアプリケーション管理者は、連鎖関係をサポートするためのディレクトリ抽出層を設定できます。この場合、組織内のいくつかのレベルが 1 つのチームに含まれます。管理者は含めるレベルの数を設定できます。

- *チームオプション*とは、プロビジョニング要求の範囲を決定するもので、チームが個別のプロビジョニング要求、1 つ以上の要求カテゴリ、またはすべての要求を実行できるかどうかを指定します。チームオプションは、チームマネージャがチームメンバーのために代理を設定できるか、または委任の目的でチームメンバーの要請を設定できるかを決定します。

注 : ユーザアプリケーションは、1 レベルの代理人割り当てのみをサポートしています。代理人割り当てが複数レベルに反映されることはありません。

プロビジョニングアプリケーション管理者は、すべてのチーム管理機能を実行できます。

チーム定義自体は、iManager 内部で 1 人以上の管理マネージャによって管理されます。

チームとグループの違い チームがアイデンティティボールト中のグループを表す場合もありますが、チームとグループは別物です。アイデンティティボールトでグループを定義する場合、共通の特徴を持つ複数ユーザを指定します。ただし、ユーザアプリケーション内でグループにチームの機能が自動的に与えられることはありません。ユーザアプリケーションでチーム機能を活用するには、そのグループを示すチームを定義する必要があります。

19.1.2 チーム要求権限の概要

*チーム要求オブジェクト*は、チームのドメイン内に該当する要求や、チームマネージャに与えられた権限を指定します。要求権限は、プロビジョニング要求やタスクでチームマネージャが実行できるアクションを指定します。

チーム定義には、チーム要求オブジェクトとの *1 対多の関係*があります。つまり、各チームには最低 1 つのチーム要求オブジェクトを関連付ける必要があります。複数のチーム要求オブジェクトを持つことも可能です。各チーム要求オブジェクトは、1 つのチームとのみ関連付けられます。

チームマネージャに対して、次のタスクスコープオプションを設定することができます。

- チームメンバーが宛先のタスクを処理する
- チームメンバーが受信者のタスクを処理する

警告 : セキュリティ上の理由から、デフォルトでは受信者のタスクスコープオプションは無効になっています。チームマネージャに、要求の受信者がチームメンバーである場合のタスクの処理権限を与えると、いくつかのセキュリティ上の問題が発生することがあります。まず、マネージャは与えられているトラスティ権利に関係なく、ワークフローの実行中に表示される任意のフォームに含まれているデータを参照できてしまいます。また、許可オプション (後述) によっては、チームマネージャはタスクの引き受けや承認によって、またはタスクの再割り当てによって、承認プロセスを迂回できてしまいます。

前述の両方のタスクスコープオプションを無効にした場合、チームマネージャはアクティブな要求を表示したり、処理することはできません。そのため、通常チームマネージャは、最低 1 つのオプションを有効にすることを望みます。

チームマネージャに対して、次の許可オプションを設定することができます。

- チームメンバーに代わってプロビジョニング要求を開始する。
- チームメンバーに代わってプロビジョニング要求を撤回する。
- あるチームメンバーを、他のチームメンバーの要求の委任にする。
- 受信者または宛先(タスクスコープに基づく)のチームメンバーに対してタスクを引き受ける。
- 受信者または宛先(タスクスコープに基づく)のチームメンバーに対してタスクを再割り当てする。

注 : ユーザアプリケーションは、1 レベルの委任割り当てのみをサポートしています。委任割り当てが複数レベルに反映されることはありません。

プロビジョニング要求に定義されたトラスティ権利は、チームメンバーに代わって要求を開始するチームマネージャに適用されます。

19.1.3 チームを使ったダイレクトレポートの管理

適切に定義すれば、単一のチーム定義を使ってすべてのマネージャに対して、各自のダイレクトレポートのアクティビティを許可できます。各チーム個別にレポートのリレーションシップを定義する必要はありません。

組織内でダイレクトレポートをサポートするチームの基本的な要件を次に示します。

- チームのメンバーは、マネージャ - 従業員リレーションシップで定義されます。
- チームのマネージャは、マネージャのみを取得する検索フィルタを使って、サブコンテナを検索するダイナミックグループにより定義されます。

チームを定義したら、すべてのマネージャはナビゲーションメニューのチーム管理アクションを使用できるようになります。これにより、マネージャはダイレクトレポートが実行できるプロビジョニングアクティビティを制御することができます。

ダイレクトレポートを管理するためのチームの定義方法については、368 ページのセクション 19.4「ダイレクトレポートを管理するチームの作成」を参照してください。

注 : このテクニックは、以前のリリースの Identity Manager ユーザアプリケーションがサポートしていた組織チームの概念に代わるものです。

19.2 プロビジョニングチームの管理

プロビジョニングチームを設定する前に、定義が含まれる Identity Manager ユーザアプリケーションドライバを選択する必要があります。ドライバを選択したら、新しいチーム定義を作成したり、既存の定義を編集したり、既存の定義を削除したりすることができます。

[プロビジョニングチーム] および *[プロビジョニングチーム要求]* タスクは、*[プロビジョニング要求]* タスクと同じドライバを使用します。

19.2.1 ドライバの選択

*［プロビジョニング要求］、［プロビジョニングチーム］、または［プロビジョニングチーム要求］*タスクのユーザアプリケーションドライバを選択したら、この iManager セッション中はユーザアプリケーションドライバを再び選択する必要はありません。

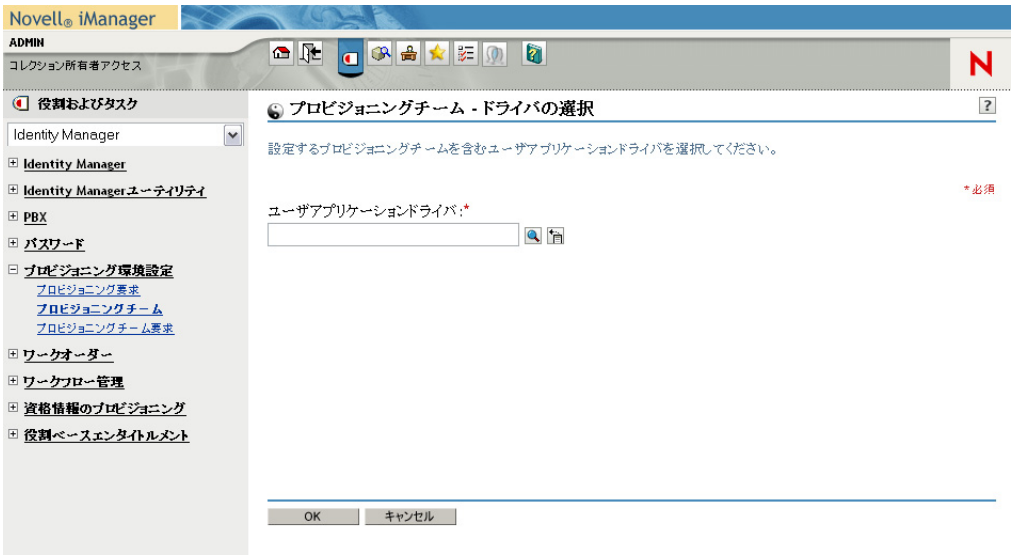
ユーザアプリケーションドライバを選択する

1 iManager で、*［Identity Manager］*カテゴリを選択します。

2 *［Provisioning Configuration］役割を開きます。*

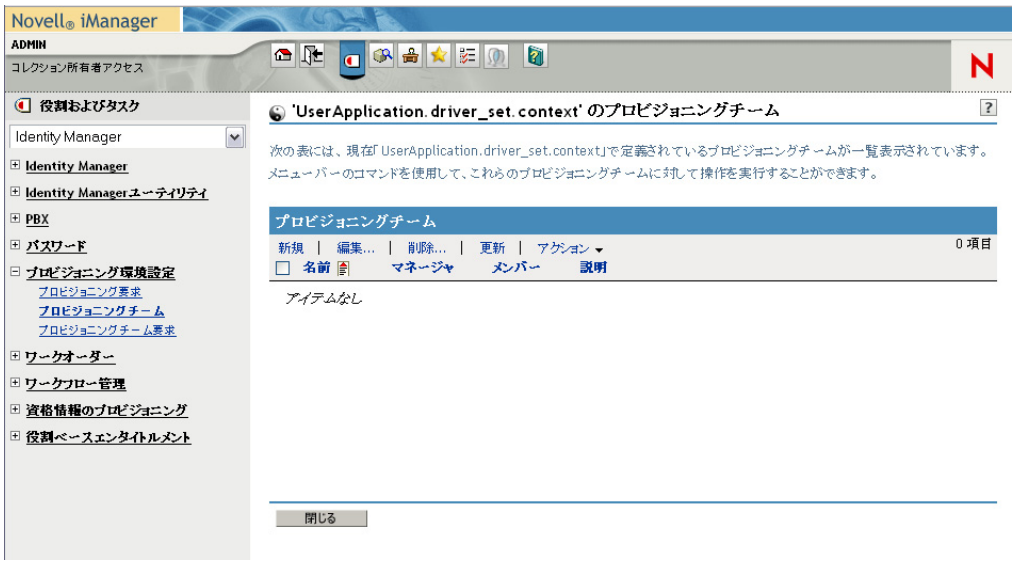
3 *［プロビジョニングチーム］*タスクをクリックします。

iManager に［ユーザアプリケーションドライバ］パネルが表示されます。



4 *［ユーザアプリケーションドライバ］*フィールドでドライバ名を指定し、*［OK］*をクリックします。

［プロビジョニングチーム］パネルが表示されます。［プロビジョニングチーム］パネルには、既存のチーム定義が一覧表示されます。



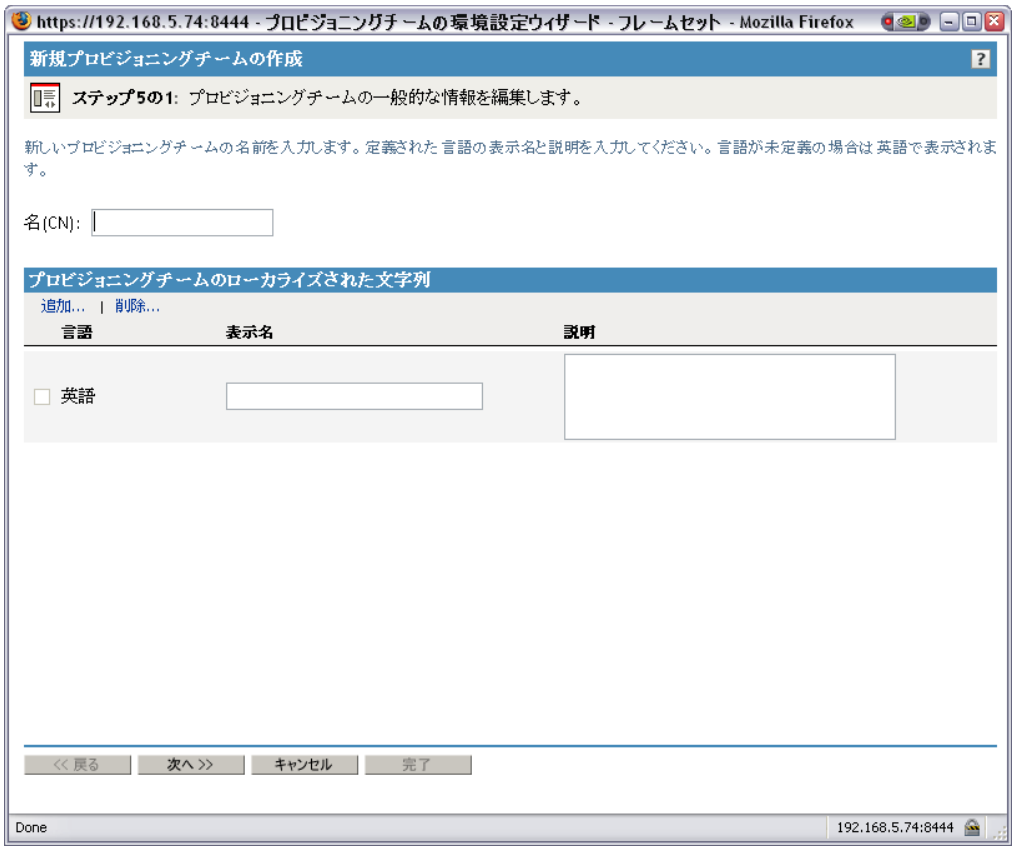
ドライバの変更 いったんドライバを選択すると、別のドライバを選択しない限り、iManager セッション中は選択したドライバが有効となります。新しいドライバを選択するには、［*Actions*］コ\'83\'7d ンドをクリックし、［Actions］メニューから［*Select User Application Driver*］を選択します。>

19.2.2 プロビジョニングチームの作成または編集

プロビジョニングチームを作成する

1 ［プロビジョニングチーム］パネルで、［*新規*］コマンドをクリックします。

新しいプロビジョニングチームの作成ウィザードの最初のページが表示されます。



2 *[名前 (CN)]* フィールドに、新しいオブジェクトの共通名を入力します。

3 アプリケーションでサポートする各言語について、*[プロビジョニングチームのローカライズされた文字列]* の *[表示名]* および *[説明]* の各フィールドにローカライズされたテキストを入力します。このテキストは、ユーザアプリケーションにおけるプロビジョニングチームの識別に使用されます。

4 新しい言語をリストに追加するには、*[追加]* をクリックしてから目的の言語を選択します。

デフォルトでは、新しく作成されたプロビジョニングチームは英語のみをサポートします。

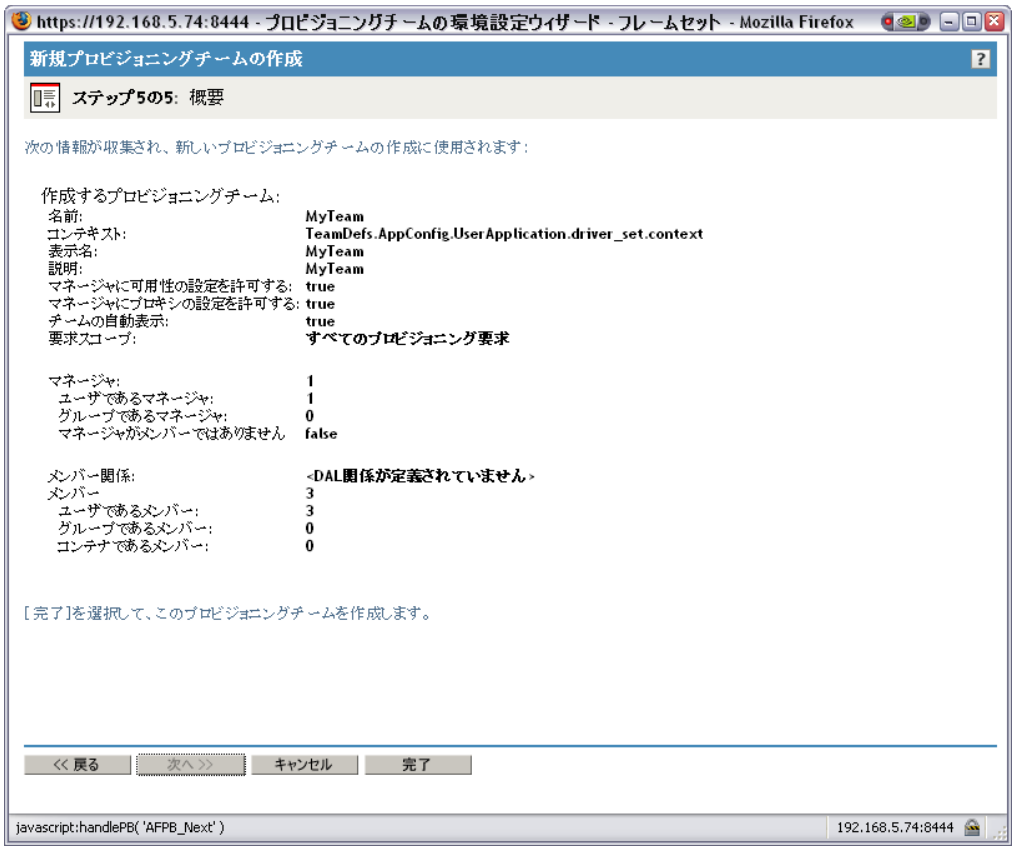
5 [*次へ*] をクリックします。

6 357 ページの 「チームマネージャの指定」の説明に従って、チームのマネージャを指定します。

7 357 ページの 「チームメンバーの指定」の説明に従って、チームのメンバーを指定します。

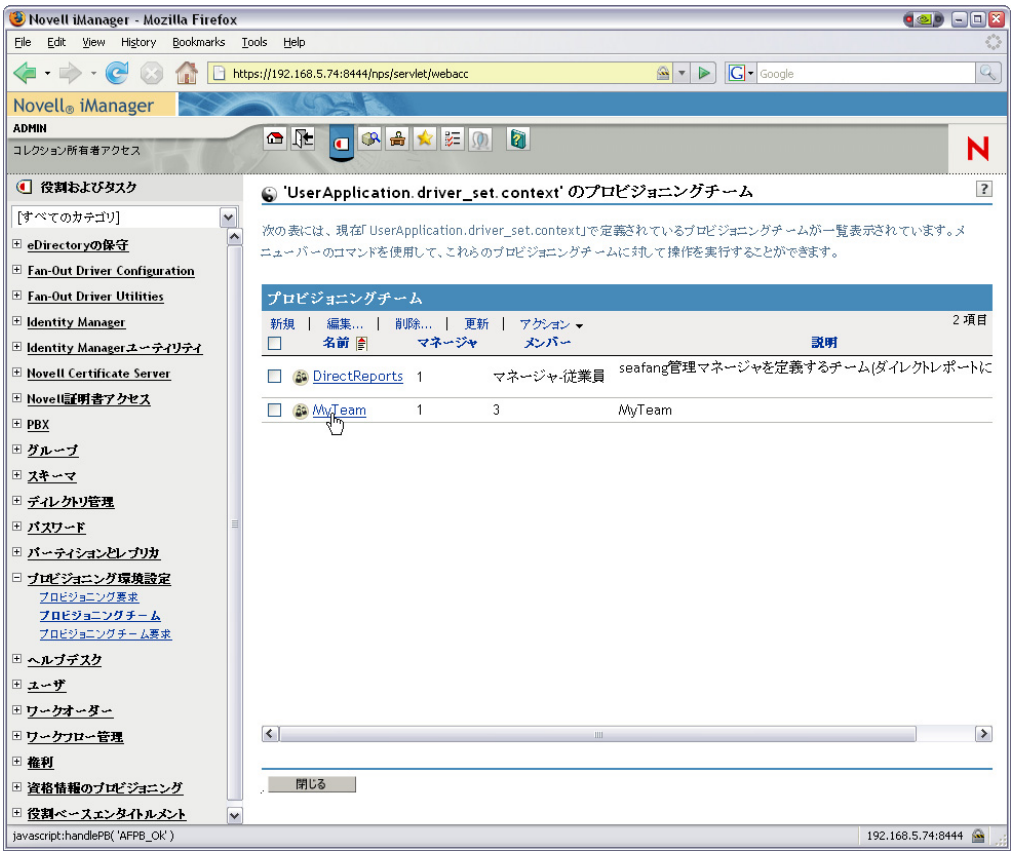
8 359 ページの 「チームオプションの指定」の説明に従って、チームオプションを指定します。

9 設定を確認し、*[完了]* をクリックします。



既存のプロビジョニングチームを編集する

1 [プロビジョニングチーム]パネルで、プロビジョニングチームの名前をクリックします。



多数のチーム定義が存在する場合は、*[名前]*、*[説明]* など特定の列でソートしなければならない場合があります。列の見出しをクリックすると、その列を基準にソートできます。

2 アプリケーションでサポートする各言語について、*[プロビジョニングチームのローカライズされた文字列]* の下に一覧表示される言語の横にあるチェックボックスをオンにし、*[表示名]* と *[説明]* の各フィールドにローカライズされたテキストを入力します。このテキストは、ユーザアプリケーションにおけるプロビジョニングチームの識別に使用されます。

3 新しい言語をリストに追加するには、*[Add] をクリックしてから対象の言語を選択します。*

デフォルトでは、新しく作成されたプロビジョニングチームは英語のみをサポートします。

4 [*次へ*] をクリックします。

5 357 ページの 「チームマネージャの指定」の説明に従って、チームのマネージャを指定します。

6 357 ページの 「チームメンバーの指定」の説明に従って、チームのメンバーを指定します。

7 359 ページの 「チームオプションの指定」の説明に従って、チームオプションを指定します。

8 設定内容を確認し、*[完了]* をクリックします。

ユーザアプリケーション内でこのチームを利用できるようにするには、チーム要求オブジェクトを定義する必要があることを知らせるメッセージが表示されます。

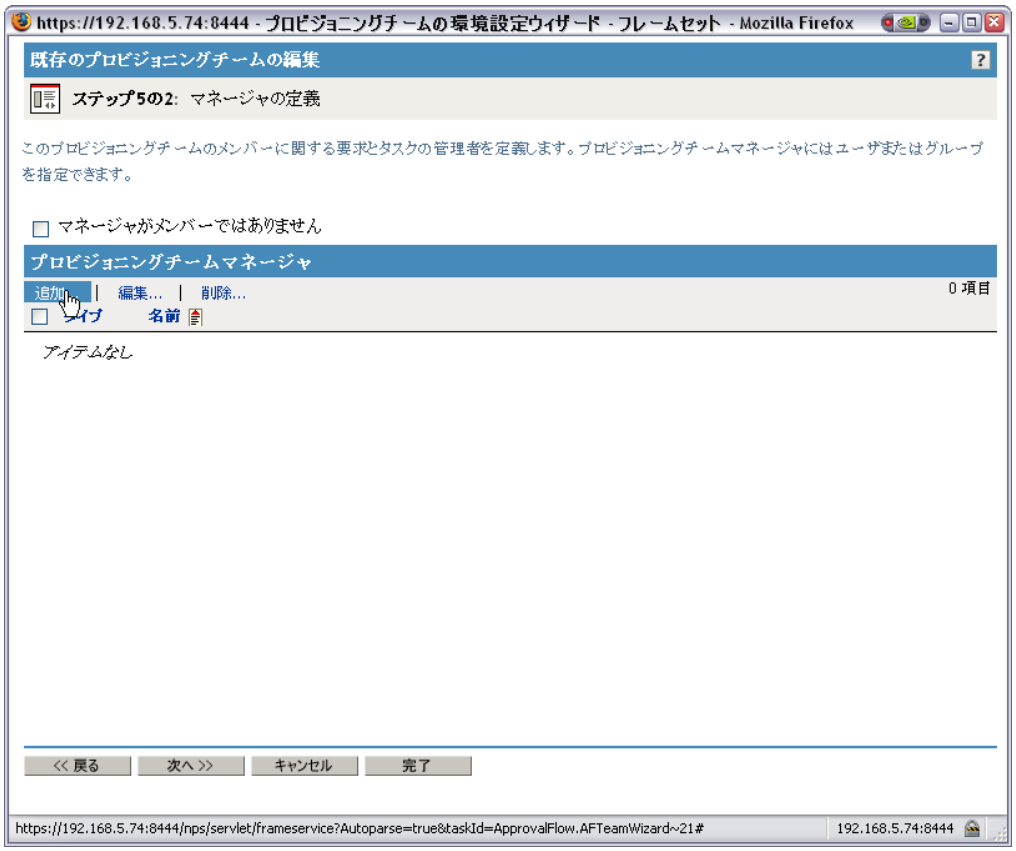
9 *[OK]* をクリックします。

チームマネージャの指定

この節では、チームのマネージャの指定方法について説明します。

チームマネージャを指定する

1 ［*追加*］をクリックします。



オブジェクトセレクタにインタフェースが表示されます。

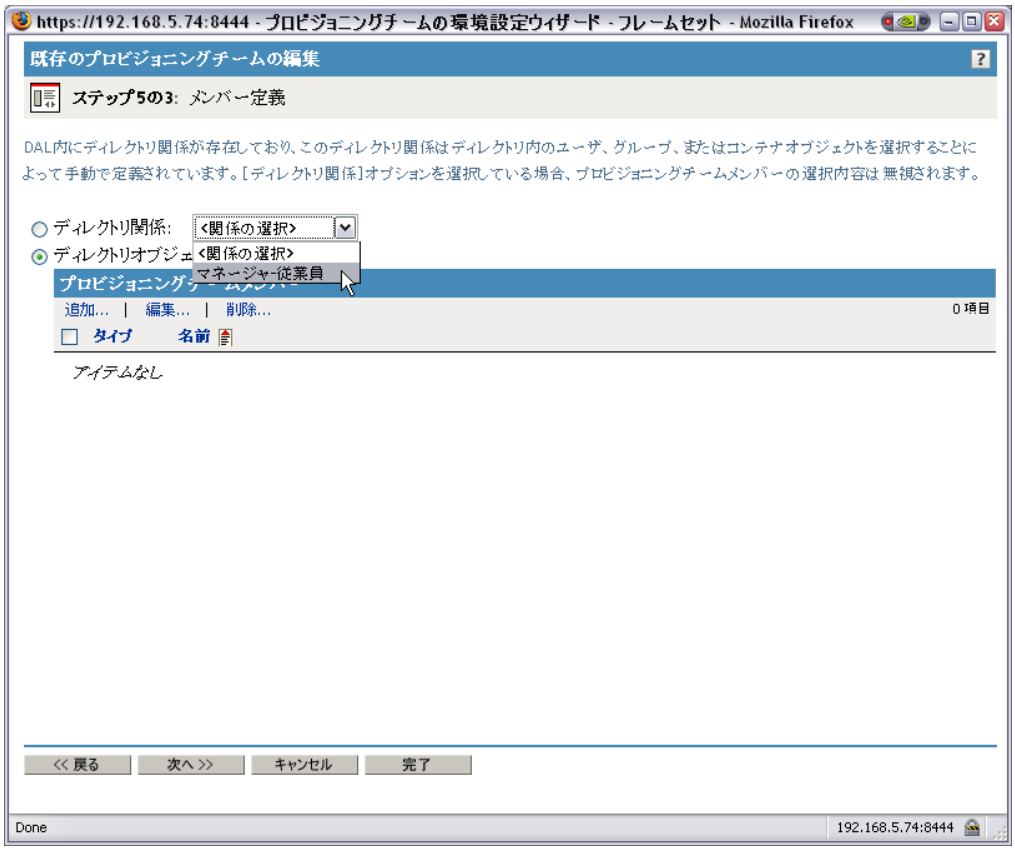
2 1 つまたは複数のユーザ／グループを選択して、*[OK]* をクリックします。

3 ［*次へ*］をクリックします。

チームメンバーの指定

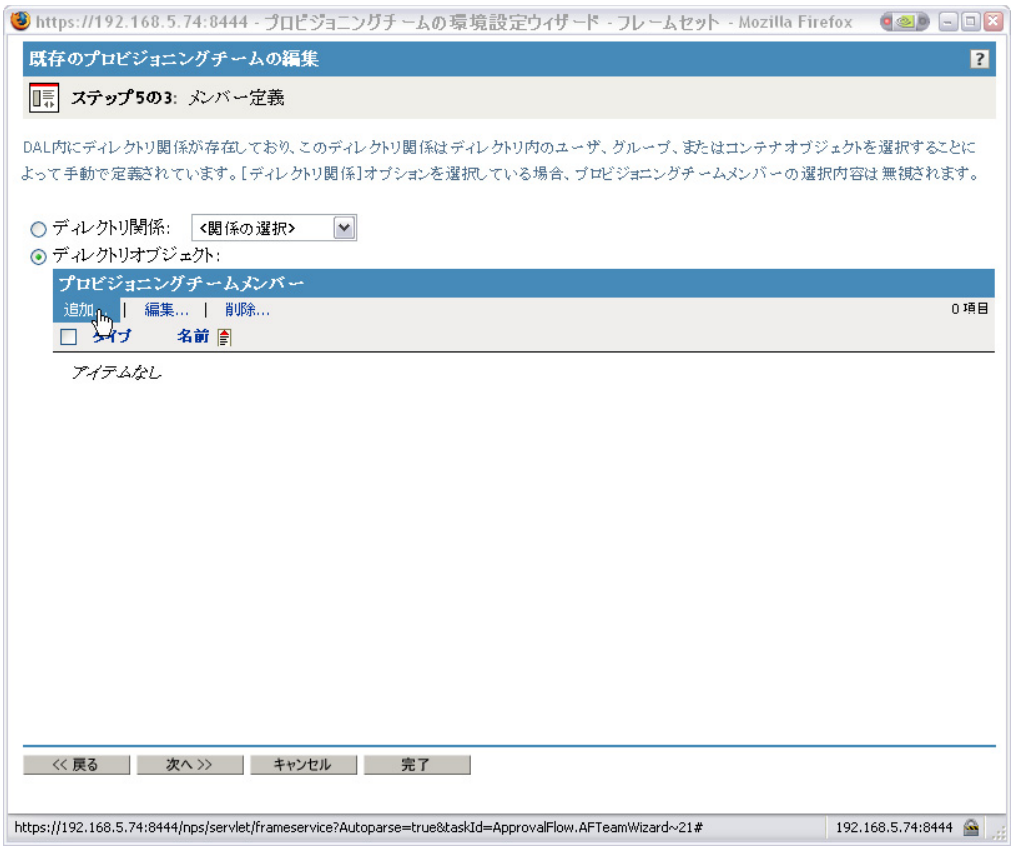
チームメンバーを指定する

1 ディレクトリ関係を使ってメンバーを定義するには、*[ディレクトリ関係]* をクリックして、次にドロップダウンリストから関係を選択します。



2 メンバーを個別に選択して定義する場合は、*[ディレクトリオブジェクト]* をクリックして、次の作業を行ってください。

2a *[追加]* をクリックします。



オブジェクトセレクタにインタフェースが表示されます。

2b 1 つまたは複数のユーザ、グループ、またはコンテナを選択して、*[OK]* をクリックします。

3 ［*次へ*］をクリックします。

チームオプションの指定

チームオプションを指定する

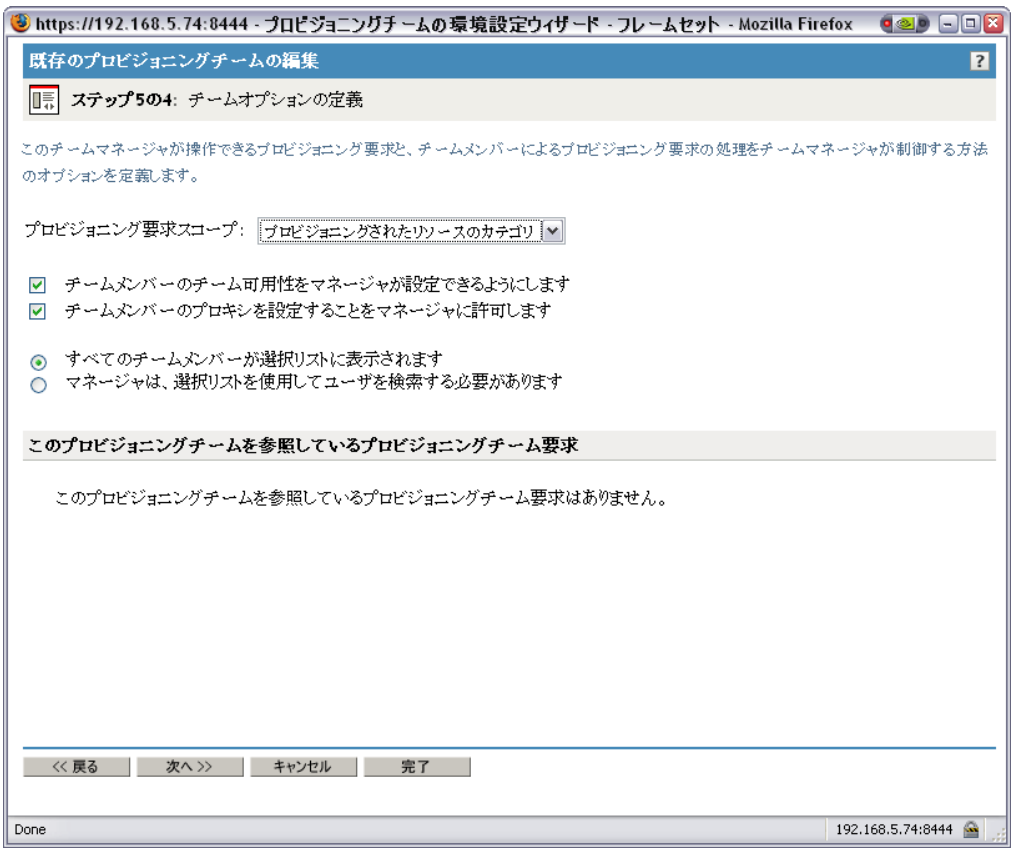
1 チームマネージャが処理する要求タイプを定義するには、*[プロビジョニング要求スコープ]* ドロップダウン リストからいずれかのオプションを選択します。

- *[個々のプロビジョニング要求]* は、このチーム定義が単一の要求タイプに適用されることを示します。チーム要求オブジェクトの定義時に、要求タイプを指定します。
- *プロビジョニング対象リソースのカテゴリ*は、このチーム定義がカテゴリに関連するすべての要求タイプに適用されることを示します。チーム要求オブジェクトの定義時に、カテゴリを指定します。
- *[すべてのプロビジョニング要求]* は、このチーム定義がすべての要求タイプに適用されることを示します。

2 チーム設定は次のように定義します。

設定	説明
チームメンバーのチーム可用性をマネージャが設定できるようにします	この設定を有効にすると、チームマネージャはユーザアプリケーションのナビゲーションメニューにある *[チームの可用性]* にアクセスできます。
チームメンバーの代理人を設定することをマネージャに許可します	この設定を有効にすると、チームマネージャはユーザアプリケーションのナビゲーションメニューにある *[チームの代理人割り当て]* にアクセスできます。
すべてのチームメンバーが選択リストに表示されます	このオプションを選択した場合、マネージャはドロップダウンリストからチームメンバーを選択できます。チームのメンバーが少ない場合に、このオプションを使用します。
マネージャは、選択リストを使用してユーザを検索する必要があります	このオプションを選択した場合、マネージャはチームメンバーの選択にオブジェクトセレクタを使用する必要があります。チームのメンバー数が多い場合に、このオプションを使用します。

特定のチーム定義によりマネージャが代理やチーム可用性を設定できないようになっている場合でも、マネージャは、管理者または他のチーム (自チームのメンバーが他に属している) のマネージャによって自チームのメンバーに定義された設定を表示できます。ただし、チームマネージャはこれらの設定の編集、詳細表示、新規代理割り当て／チーム可用性設定の作成はできません。



3 このチーム定義を参照するチーム要求オブジェクトがある場合、リストの *［このプロビジョニングチームを参照しているプロビジョニングチーム要求］* にあるオブジェクト名をクリックして、これらのオブジェクトに直接移動することができます。

チーム要求オブジェクトをクリックすると、*［プロビジョニング要求スコープ］* 設定のコミットを要求するメッセージが表示されます。*［OK］* をクリックしてこの設定をコミットすると、プロビジョニングチーム要求プラグインに移動します。ここから、チーム要求オブジェクトを変更できます。

19.2.3 プロビジョニングチームの削除

プロビジョニングチームを削除する

1 名前の横にあるチェックボックスを選択し、削除するプロビジョニングチームを選択します。
2 ［プロビジョニングチーム］パネルで、*［削除］* コマンドをクリックします。

19.3 プロビジョニングチーム要求権限の管理

プロビジョニングチーム要求オブジェクトを設定する前に、定義が含まれる Identity Manager ユーザアプリケーションドライバを選択する必要があります。ドライバを選択したら、新しいチーム要求を作成したり、既存の定義を編集したり、既存の定義を削除したりすることができます。

19.3.1 ドライバの選択

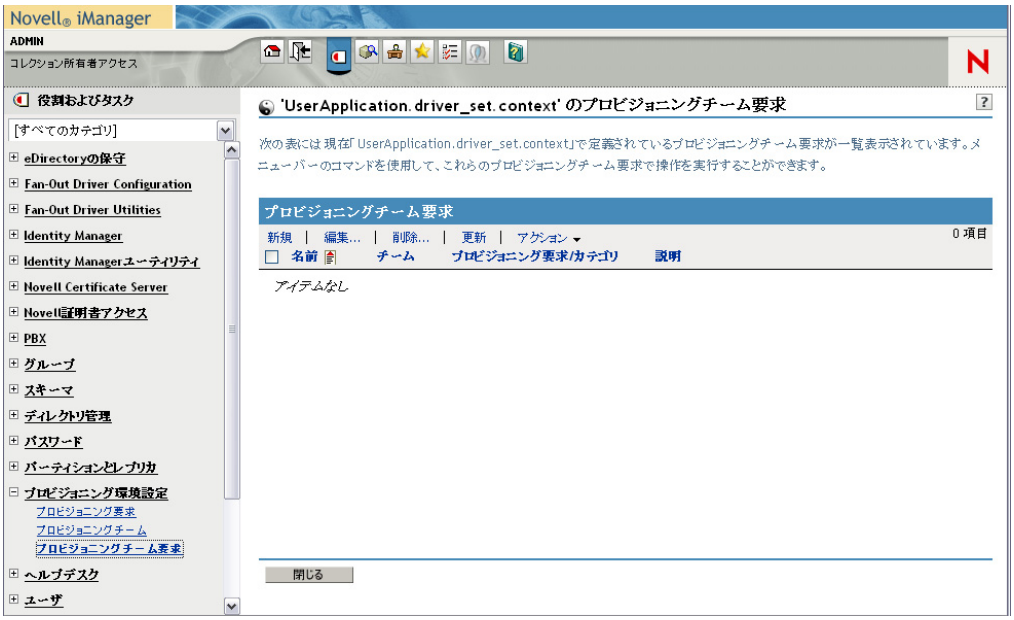
Identity Manager ユーザアプリケーションドライバを選択する

1 iManager で、*［Identity Manager］* カテゴリを選択します。
2 ［*Provisioning Configuration*］*役割を開きます。*
3 *［プロビジョニングチーム要求］* タスクをクリックします。

iManager に［ユーザアプリケーションドライバ］パネルが表示されます。

4 *［ユーザアプリケーションドライバ］* フィールドでドライバ名を指定し、*［OK］* をクリックします。

［プロビジョニングチーム要求］パネルが表示されます。［プロビジョニングチーム要求］パネルには、既存のチーム要求オブジェクトが表示されます。

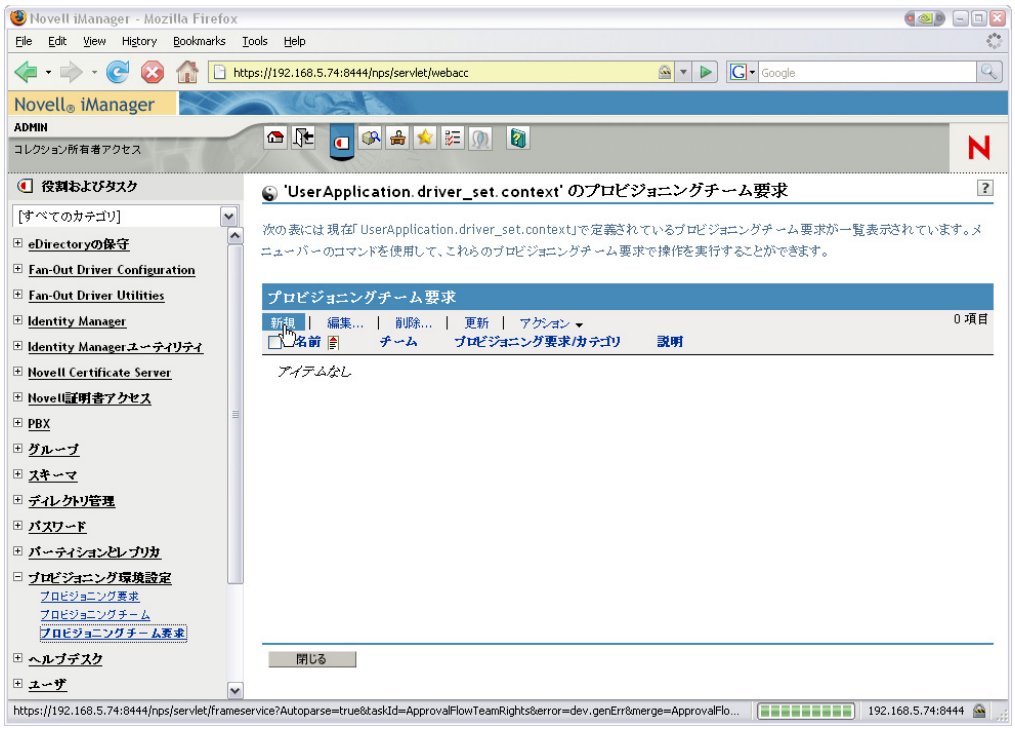


ドライバの変更 いったんドライバを選択すると、別のドライバを選択しない限り、iManager セッション中は選択したドライバが有効となります。新しいドライバを選択するには、*［アクション］*コマンドをクリックし、*［アクション］*メニューから *［ユーザアプリケーションドライバの選択］*を選択します。

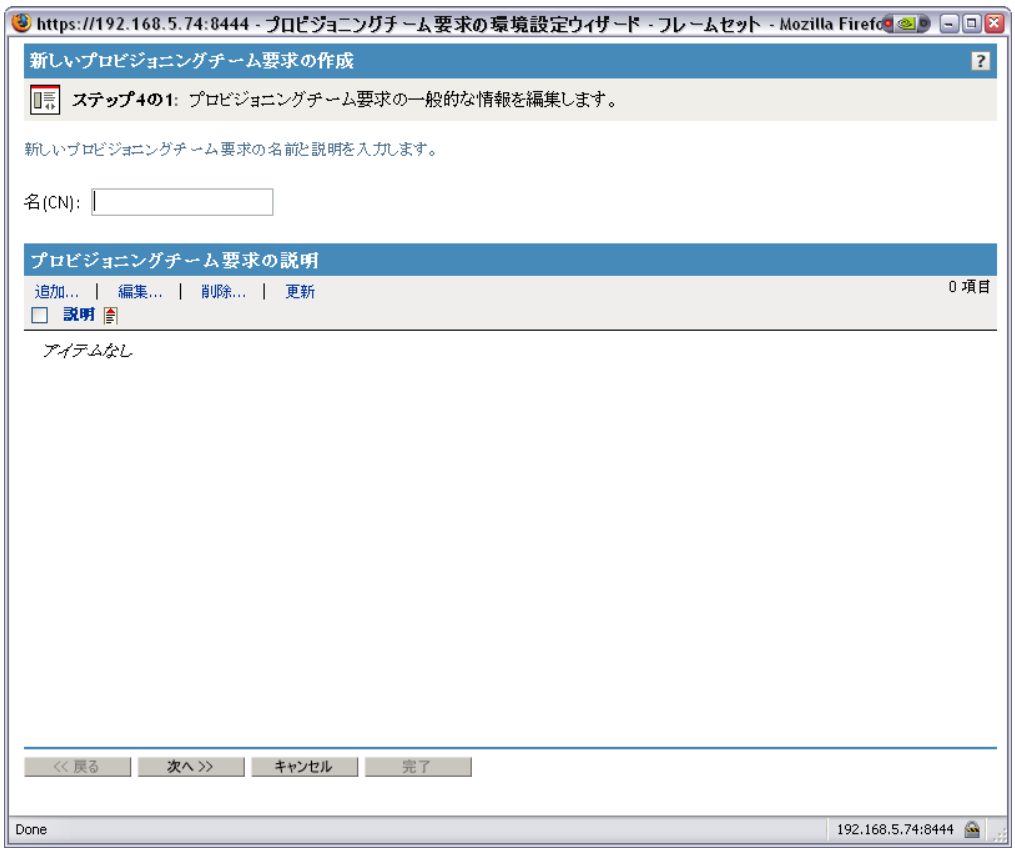
19.3.2 プロビジョニングチーム要求オブジェクトの作成または編集

新しいプロビジョニングチーム要求オブジェクトを作成する

1 ［プロビジョニングチーム要求］パネルで、*［新規］*コマンドをクリックします。



新しいプロビジョニングチーム要求の作成ウィザードの最初のページが表示されます。



2 *[名前 (CN)]* フィールドに、新しいオブジェクトの共通名を入力します。

3 チーム要求オブジェクトに追加する各説明にに対して、*[プロビジョニングチーム要求の説明]* の *[説明]* フィールドに説明を入力します。ここに入力した説明は、iManager でプロビジョニングチーム要求オブジェクトを識別するために使用されます。

4 各チーム要求オブジェクトに新しい説明を追加するには、*[追加]* をクリックして説明を入力し、*[OK]* をクリックします。

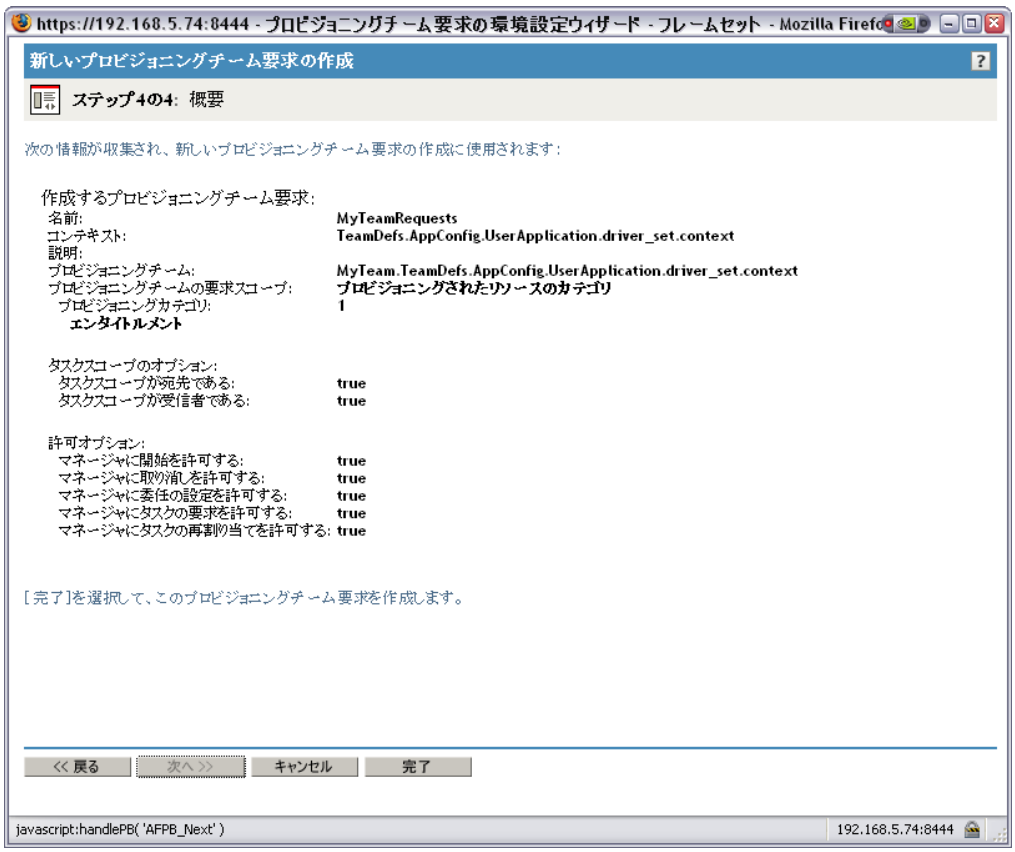
[プロビジョニングチーム要求の説明] の *[説明]* フィールドに入力した説明が表示されます。ここに入力した文字列は、[プロビジョニングチーム要求] パネルのチーム要求オブジェクトの説明に用いられます。

5 [*次へ*] をクリックします。

6 このチーム要求オブジェクトを適用するチーム定義を選択します。364 ページの「チーム要求オブジェクトのチーム定義の選択」を参照してください。

7 チーム要求オブジェクトのタスクスコープと許可オプションを指定します。365 ページの 「チーム要求オプションの指定」を参照してください。

8 設定内容を確認し、*[完了]* をクリックします。

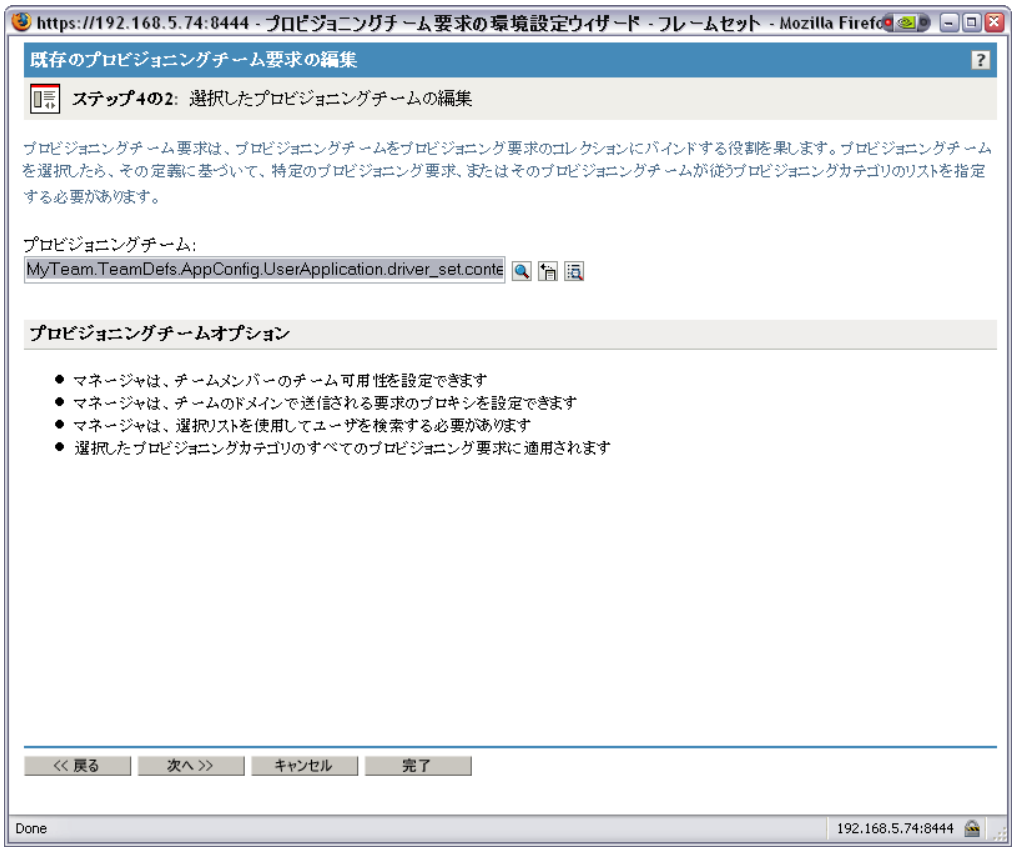


チーム要求オブジェクトのチーム定義の選択

チーム定義を選択する

1 オブジェクトセレクタを使ってチームを選択します。

選択したチームが *[プロビジョニングチーム]* フィールドに表示されます。*[プロビジョニングチームオプション]* には、そのチームのオプション設定が表示されます。



2 [*次へ*] をクリックします。

チーム要求オプションの指定

チーム要求オプションを指定する

1 チーム要求オブジェクトのスコープを定義します。

- チームのスコープが *[プロビジョニング対象リソースのカテゴリ]* の場合、*[使用できるカテゴリ]* リストから *[選択されたカテゴリ]* リストに、必要なチーム要求オブジェクトのカテゴリを移動します。

- チームのスコープが *[個々のプロビジョニング要求]* の場合、オブジェクトセレクタを使ってこのチーム要求オブジェクトのプロビジョニング要求を選択します。

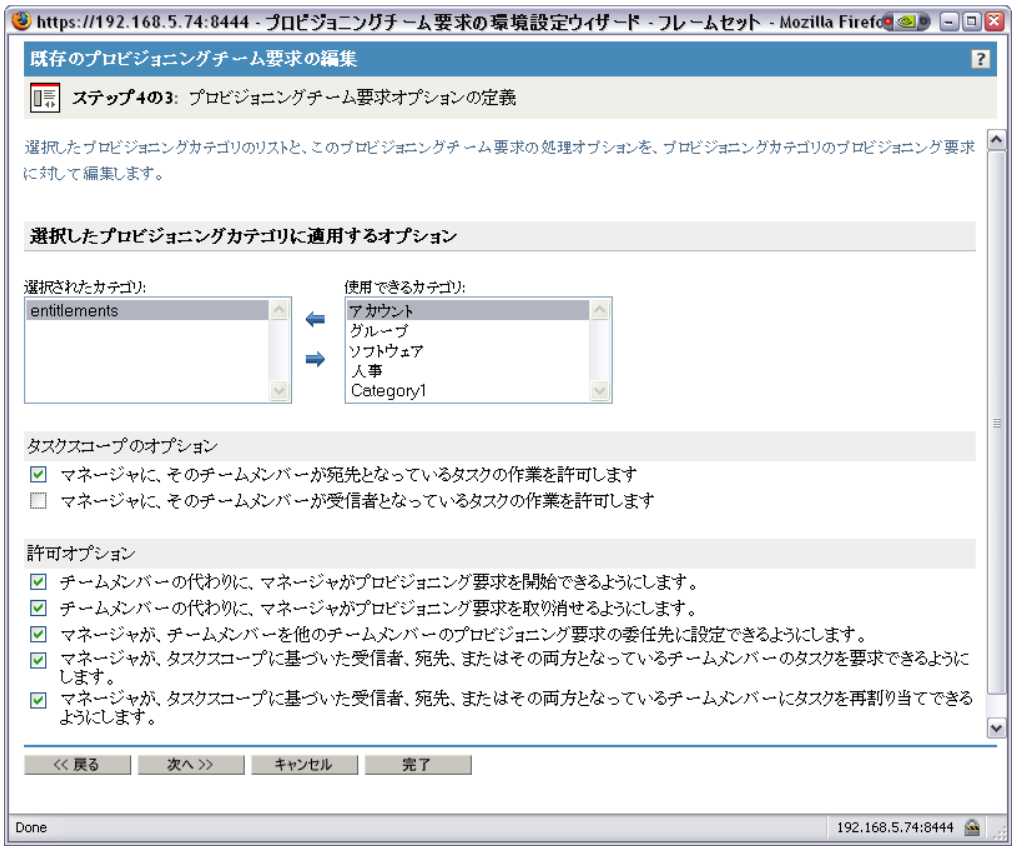
- チームのスコープが［すべてのプロビジョニング要求］の場合は、特に作業は必要ありません。

2 次のように、タスクスコープのオプションを定義していきます。

設定	説明
マネージャに、そのチームメンバーが宛先となっているタスクの作業を許可します	この設定を有効にすると、チームマネージャは *［チームタスク］* を使って、チームメンバーが宛先になっているタスクに対して作業を行うことができます。この作業には、要求の承認や拒否も含まれます。 チームマネージャに対してチームメンバーが宛先になっているタスクに対する作業を許可しない場合、タスクは表示できますが、その詳細を参照したり、何らかの作業を行うことはできません。
マネージャに、そのチームメンバーが受信者となっているタスクの作業を許可します	この設定を有効にすると、チームマネージャは *［チームタスク］* を使って、チームメンバーが受信者になっているタスクに対して作業を行うことができます。この作業には、要求の承認や拒否も含まれます。 チームマネージャに対してチームメンバーが受信者になっているタスクに対する作業を許可しない場合、タスクは表示できますが、その詳細を参照したり、何らかの作業を行うことはできません。 **注：**セキュリティ上の理由から、デフォルトでは受信者のタスクスコープオプションは無効になっています。チームマネージャに、要求の受信者がチームメンバーである場合のタスクの処理権限を与えると、いくつかのセキュリティ上の問題が発生することがあります。まず、マネージャは与えられているトラスティ権利に関係なく、ワークフローの実行中に表示される任意のフォームに含まれているデータを参照できてしまいます。また、許可オプション (後述) によっては、チームマネージャはタスクの引き受けや承認によって、またはタスクの再割り当てによって、承認プロセスを迂回できてしまいます。

3 次のように、許可オプションを設定していきます。

設定	説明
チームメンバーの代わりに、マネージャがプロビジョニング要求を開始できるようにします	この設定を有効にすると、ユーザアプリケーションの *[チームリソースの要求]* ページに表示されるリソースのリストに、このチームのスコープ内のリソースが含まれます。この設定を無効にすると、これらのリソースは含まれません。
チームメンバーの代わりに、マネージャがプロビジョニング要求を取り消せるようにします	この設定を有効にすると、*[チーム要求]* ページに［撤回］ボタンが表示されます。このボタンを使って、このチームのスコープ内の要求を撤回することができます。この設定を無効にすると、［撤回］ボタンは表示されません。
マネージャが、チームメンバーを他のチームメンバーのプロビジョニング要求の委任先に設定できるようにします	このオプションを有効にした場合、マネージャは *[チームの委任割り当て]* アクションを使用してあるチームメンバーを他のチームメンバーのプロビジョニング要求の委託先 (委任) に指名することができます。 このオプションを無効にした場合、マネージャは管理者またはユーザが所属する他のチームのマネージャが定義した委任設定を参照することができます。ただし、チームマネージャはこれらの設定の編集または削除、詳細表示、新規委任の割り当てはできません。
マネージャが、タスクスコープに基づいた受信者、宛先、またはその両方となっているチームメンバーのタスクを要求できるようにします	この設定を有効にすると、*[チームのタスク]* ページに［引き受け］ボタンが表示されます。このボタンを使って、このチームのスコープ内の要求を引き受けることができます。この設定を無効にすると、［引き受け］ボタンも無効になります。
マネージャが、タスクスコープに基づいた受信者、宛先、またはその両方となっているチームメンバーにタスクを再割り当てできるようにします	この設定を有効にすると、*[チームのタスク]* ページに［再割り当て］ボタンが表示されます。このボタンを使って、このチームのスコープ内の要求を再割り当てすることができます。この設定を無効にすると、［再割り当て］ボタンも無効になります。



4 ［*次へ*］をクリックします。

注 : プロビジョニングチーム要求プラグインでは、同じプロビジョニング要求、または同じチームの異なる許可セットを持つカテゴリを使用する、2 つの異なるチーム要求オブジェクトを設定することができます。この場合、チームと許可の関係が不明瞭になり、競合が発生する可能性があります。このような競合を回避するために、同じプロビジョニング要求／カテゴリに対して異なる許可セットを指定した 2 つの異なるチーム要求オブジェクトを定義しないようにしてください。

19.3.3 プロビジョニングチーム要求オブジェクトの削除

プロビジョニングチーム要求オブジェクトを削除する

1 削除するプロビジョニングチーム要求オブジェクト名の隣にあるチェックボックスを選択します。

2 ［プロビジョニングチーム要求］パネルで、*［削除］*コマンドをクリックします。

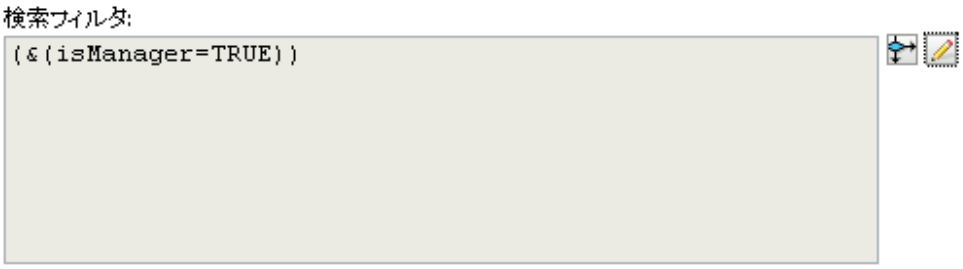
19.4 ダイレクトレポートを管理するチームの作成

ダイレクトレポートを管理するチームを定義する

1 iManager で、「Managers」と言う名前のダイナミックグループを作成します。

1a *［検索スコープ］*に *［サブコンテナを検索］*を設定します。

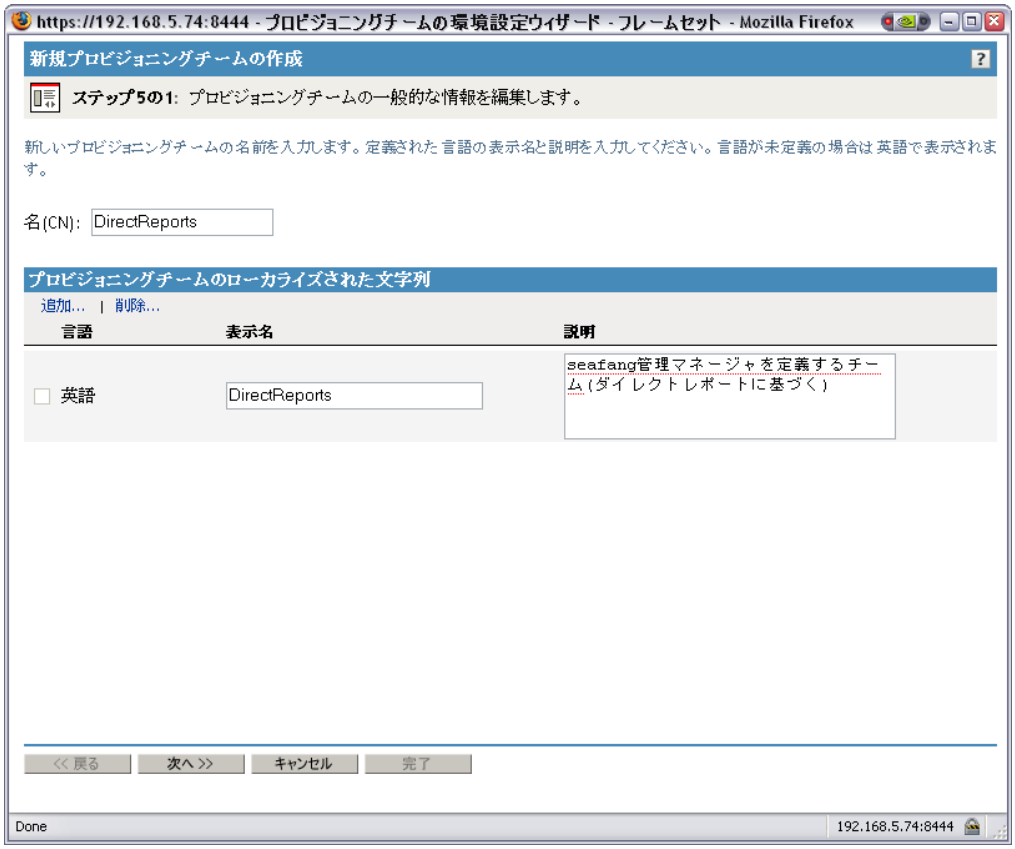
1b *［検索フィルタ］*に［(&(isManager=TRUE))］を設定します。



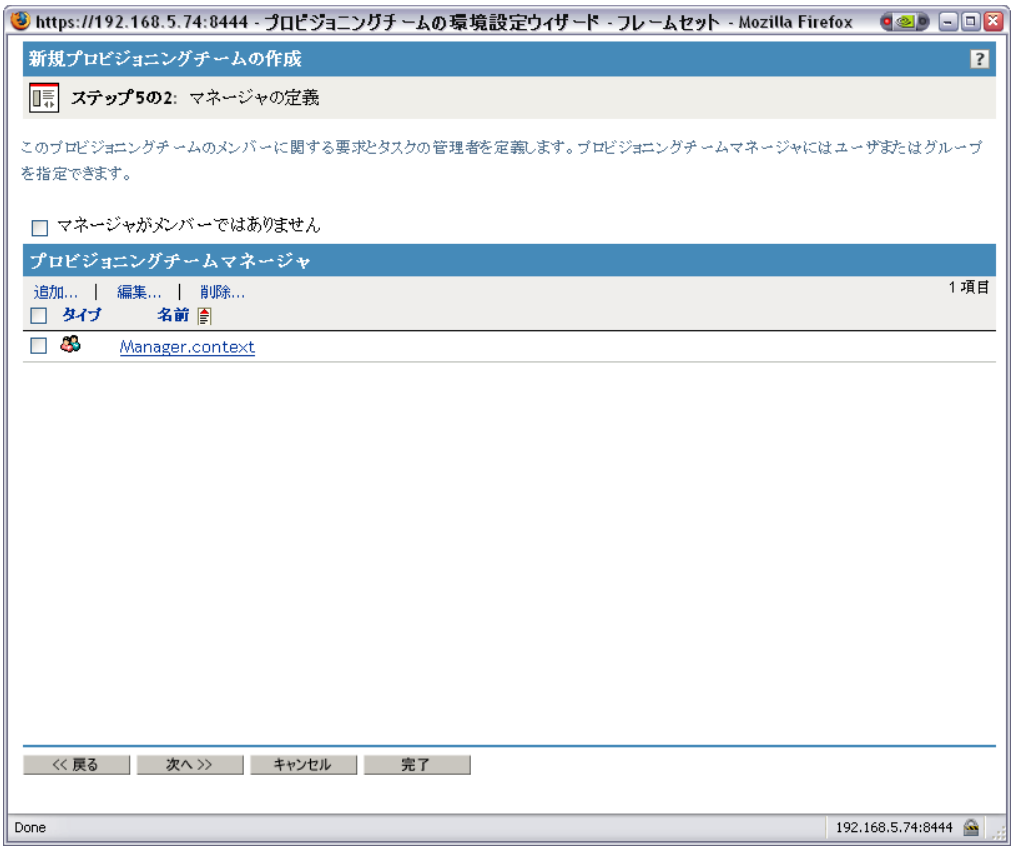
ダイナミックグループの作成の詳細については、『*Novell Identity Manager: 管理ガイド*』を参照してください。

2 iManager で、*［プロビジョニング環境設定］*の *［プロビジョニングチーム］*を選択してプロビジョニングチームを定義します。

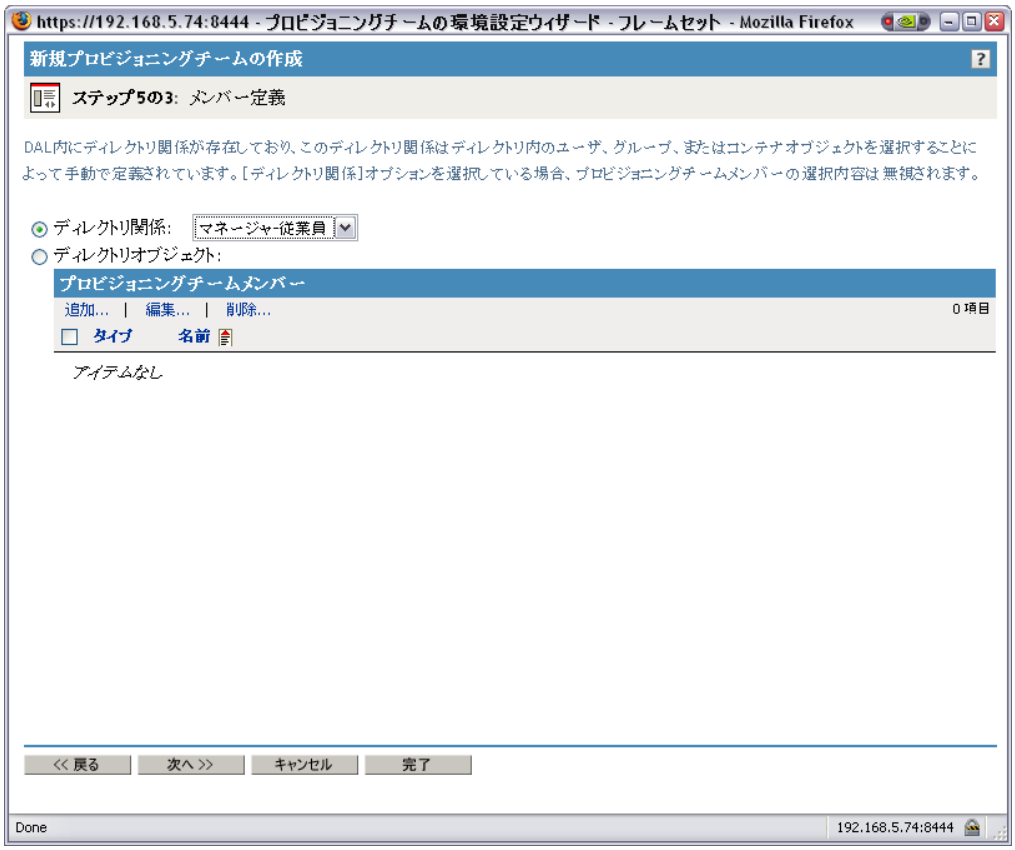
2a チーム DirectReports の名前を指定します。



2b チームマネージャを指定するために、先ほど作成したダイナミックグループ Managers を選択します。

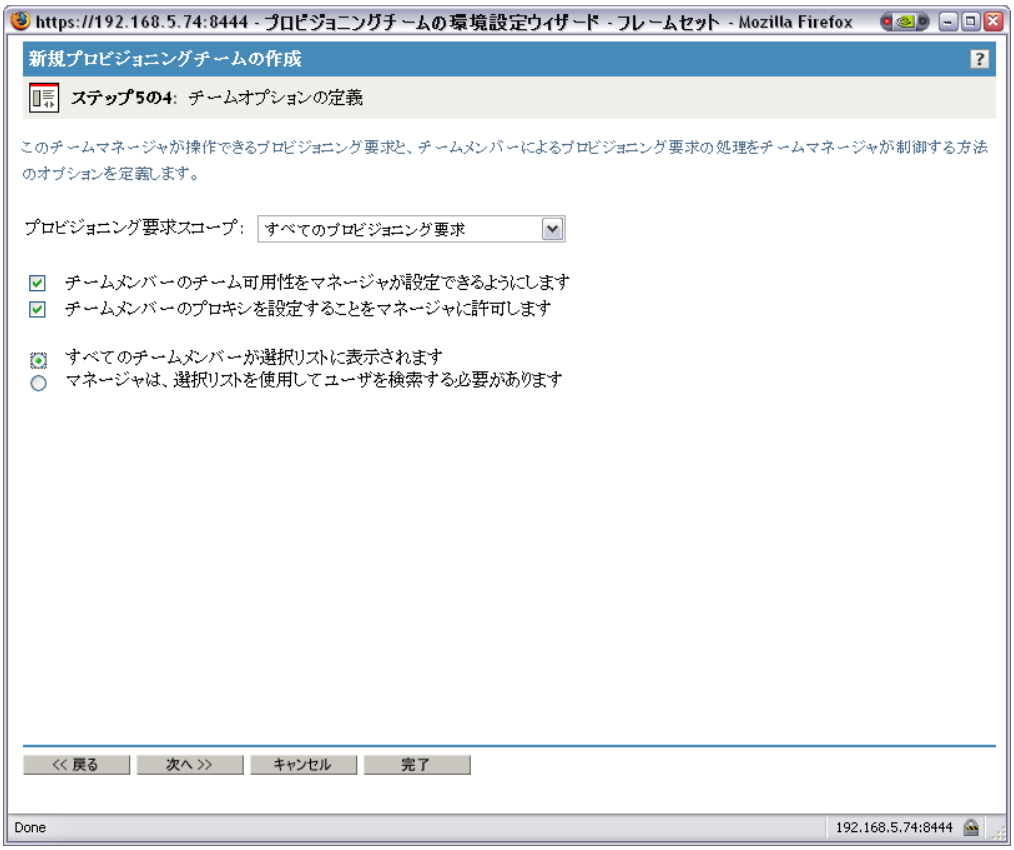


2c チームメンバーを指定するには、*[マネージャ - 従業員]* 関係を選択します。



2d チームオプションを定義する

- *[プロビジョニング要求スコープ]* を *[すべてのプロビジョニング要求]* に設定します。
- *[チームメンバーのチーム可用性をマネージャが設定できるようにします]* を選択します。
- *[チームメンバーのプロキシを設定することをマネージャに許可します]* を選択します。
- *[すべてのチームメンバーが選択リストに表示されます]* を選択します。



2e ［概要］ページの内容を確認し、*［終了］*をクリックします。

3 iManager で、*［プロビジョニング環境設定］*の *［プロビジョニングチーム要求］*を選択してプロビジョニングチーム要求オブジェクトを定義します。

3a チーム DirectReportsTeamRequestRights の名前を指定します。

3b 関連するチームを指定するには、先ほど作成した DirectReports プロビジョニングチームを選択します。このチームを選択すると、チームの設定が表示されます。

3c タスクスコープオプションを指定する

- *［マネージャに、そのチームメンバーが宛先となっているタスクの作業を許可します］*を選択します。
- *［マネージャに、そのチームメンバーが受信者となっているタスクの作業を許可します］*の選択を解除します。

3d 許可オプションを定義する

- *［チームメンバーの代わりに、マネージャがプロビジョニング要求を開始できるようにします］*を選択します。
- *［チームメンバーの代わりに、マネージャがプロビジョニング要求を取り消せるようにします］*を選択します。
- *［マネージャが、チームメンバーを他のチームメンバーのプロビジョニング要求の委任先に設定できるようにします］*を選択します。

- *[マネージャが、タスクスコープに基づいた受信者、宛先、またはその両方となっているチームメンバーのタスクを要求できるようにします]* を選択します。
- *[マネージャが、タスクスコープに基づいた受信者、宛先、またはその両方となっているチームメンバーにタスクを再割り当てできるようにします]* を選択します。

3e［概要］ページの内容を確認し、*[終了]* をクリックします。

Web サービス参照

VI

これらの節では、ユーザアプリケーションに用意されている Web サービスエンドポイントについて説明します。

プロビジョニング Web サービス 20

この節では、SOAP クライアントがプロビジョニング機能にアクセスするための、プロビジョニング Web サービスについて説明します。主なトピックは次のとおりです。

20.1 プロビジョニング Web サービスについて

Identity Manager のプロビジョニングモジュールには、承認フローを実行するワークフローシステムが用意されています。ワークフロープロセスは、プロビジョニング要求定義に基づいています。プロビジョニング要求定義は、アイデンティティボールトに保管される XML ドキュメントです。プロビジョニング要求定義は、アクティビティとリンクを使って任意のトポロジを記述しています。たとえば、エンタイトルメントを付与するプロビジョニング要求に、関連するユーザから承認を収集し、ディレクトリにエンタイトルメントを書き込むワークフローを入れることができます。

サードパーティのソフトウェアアプリケーションによるアクセスをサポートするために、プロビジョニングワークフローシステムには Web サービスエンドポイントが用意されています。このエンドポイントは、すべてのプロビジョニング機能 (例 :SOAP クライアントに新しい承認ワークフローの開始を許可したり、現在実行中のフローを表示する) を提供しています。Web サービスは、Novell Web Service SDK(WSSDK) を使って作成されています。この SDK は、WS-I 基本プロファイルをサポートしており、他の標準ベースの SOAP 実装との相互運用性が保証されます。

この付録では、プロビジョニング Web サービスの詳細と、Web を使ったアクセス方法、および Java または C# クライアントを作成することによるアクセス方法を説明しています。また、SOAP エンドポイントの操作の概要と、Web インタフェースの使用方法についても説明しています。さらに、Identity Manager のプロビジョニングに用意されている SOAP ツールキットを使った Java クライアントの開発方法や、Mono を使った C# クライアントの作成方法についても説明していきます。Java クライアントのサンプルソースコードと、関連する ANT ビルドファイルも記載されています。

20.1.1 プロビジョニング Web サービスの概要

Identity Manager は、アイデンティティボールトとワークフローアプリケーションの 2 つのメインシステムで構成されています。アイデンティティボールトは、データベース、金融システム、および他の企業アプリケーションなど、多数の異なるシステムに接続し、これらのシステムを同期化することができます。リモートシステムを同期化するルールは大変複雑になることがあります。アイデンティティボールトエンジンは、このルールを表記するための、洗練されたスクリプト言語をサポートしています。

ワークフローアプリケーションは、複数のサブシステムで構成されています。ユーザアプリケーションは、ワークフロー用のユーザインタフェースを提供しています。ユーザアプリケーションは、承認フローを要求、管理するための、Web アプリケーションです。Web アプリケーションはポータル内で動作します。これには、管理ポートレットも含まれま

す。ワークフローアプリケーションには、セキュリティ層、ディレクトリ抽象化層、および Novell Audit や Novell Sentinel にログを送信できるログサブシステムが含まれています。ワークフローサブシステムは、承認フローの実行を担当します。ユーザアプリケーションは、アプリケーションサーバ (例 :JBoss) 上で動作し、データベース (例 :Oracle、MySQL) にデータを保存します。

ワークフローシステムの Web サービスは、ユーザアプリケーションドライバのみが使用します。ユーザアプリケーションドライバは、アイデンティティボールトエンジンから発行される特定のイベントを待機して、それらのイベントを適切な SOAP メッセージに変換できます。たとえば、アイデンティティボールト内の特定の属性が変更された場合、アイデンティティボールトエンジンがイベントを発行します。ユーザアプリケーションは、このイベントを購読者チャネルから取得します。次に、ユーザアプリケーションドライバは、SOAP メッセージをプロビジョニング Web サービスに送信し、新しい承認フローを開始します。

20.1.2 プロビジョニング Web サービスメソッドのカテゴリ

プロビジョニング Web サービスエンドポイントが提供するメソッドは、6 種類のカテゴリに分類されます。

表 20-1 *プロビジョニング Web サービス操作カテゴリ*

カテゴリ	説明
コメント	コメントを取得したり、コメントを保留中のユーザアクティビティに追加するメソッドです。
設定	ワークフローシステムの環境設定パラメータを取得、設定するメソッドです (たとえば、タイムアウトやスレッドプールの設定など)。
その他	複数の無関係なメソッドです (たとえば、プロビジョニング要求のトポロジーがある JPG の取得、プロビジョニング要求の XML 定義の取得、および要求フォームの XML の取得など)。
プロセス	実行中および完了したワークフロープロセスに関する情報を取得するメソッドです。
プロビジョニングリクエスト	プロビジョニング要求を使って作業するメソッドです (たとえば、利用可能なプロビジョニング要求の表示、プロビジョニングカテゴリの表示など)。
ワークエントリ	ワークエントリ (承認待ちの項目) を取得、操作するためのメソッドです。

プロビジョニング Web サービスが提供するメソッドの詳細は、389 ページのセクション 20.3「プロビジョニング Web サービスの API」を参照してください。

20.2 プロビジョニング Web サービス用クライアントの開発

この項では、次のトピックについて説明します。

20.2.1 プロビジョニング Web サービスへの Web アクセス

通常 SOAP ベースの Web サービスは、HTTP Post 要求の本文に SOAP メッセージを挿入することによりアクセスします。プロビジョニング Web サービスを作成するために使われる Web サービスツールキットは、HTTP GET を使ったアクセスもサポートしています。つまり、ブラウザを使って Web サービスエンドポイントの URL を開き、Web サービスとやり取りすることができます。プロビジョニング Web サービスでは、各操作を起動することができます。

テストページへのアクセス

プロビジョニング Web サービスエンドポイントへのアクセスには、次のような URL を使用します。

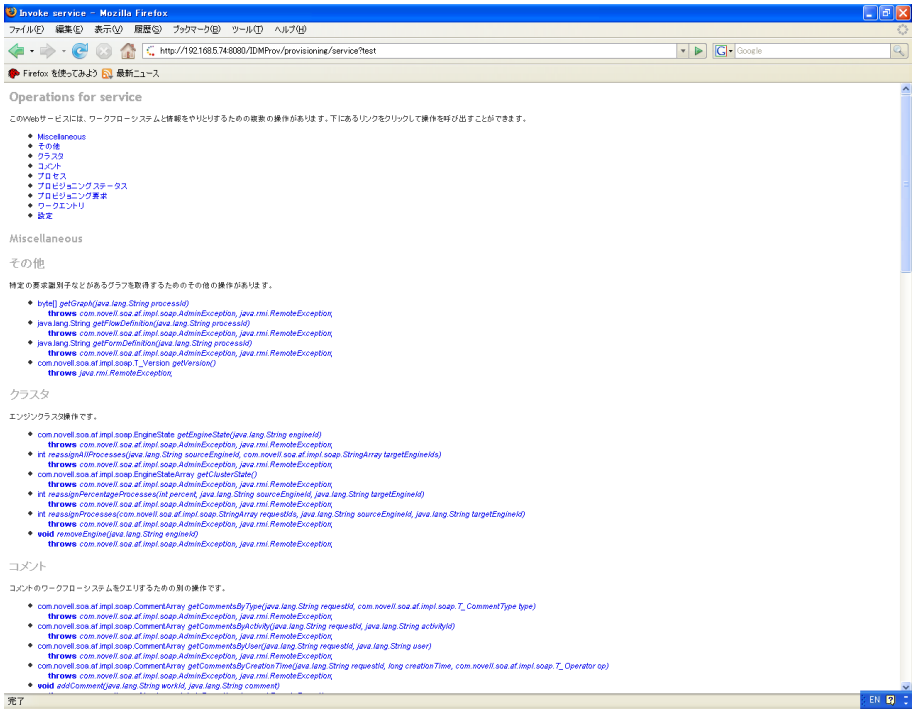
```
http://server:port/warcontext/provisioning/service?test
```

たとえば、サーバ名が「myserver」で、ユーザアプリケーションがポート 8080 で待機しており、ユーザアプリケーション WAR ファイル名が「IDMPROV」の場合、URL は次のようになります。

```
http://myserver:8080/IDMPROV/provisioning/service?test
```

次のページが表示されます。

図 20-1 Web サービステストページ



操作用の引数の入力

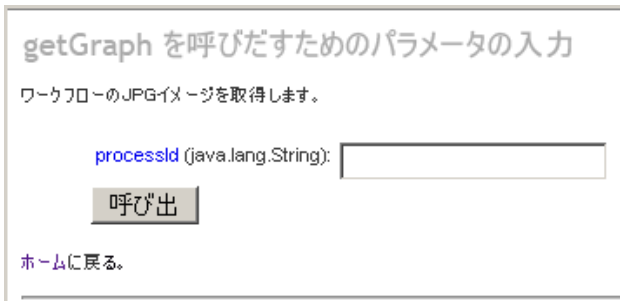
ブラウザから起動する場合に役立つ操作の例は、*[その他]* セクションまでスクロールして、*[getGraph]* をクリックします。

注 : アプリケーションサーバと IDM ユーザアプリケーションが動作するコンピュータには、Graphviz プログラムをインストールしておく必要があります。Graphviz の詳細は、「Graphviz (http://www.graphviz.org)」を参照してください。

---

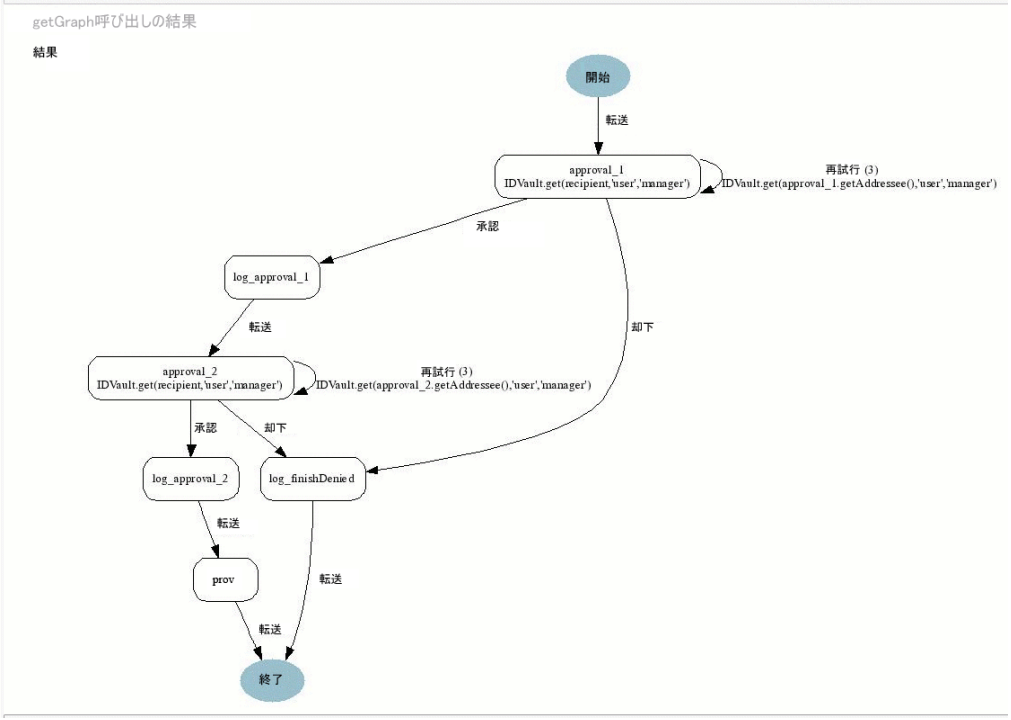
getGraph メソッドのパラメータを入力できるページが表示されます。

図 20-2 getGraph メソッドのパラメータ



このメソッドは、1 つの引数を受け取ります。この引数は、プロビジョニング要求の識別名です。DN を入力すると、該当するワークフローが JPG ファイルとして表示されます。

図 20-3 getGraph の出力



20.2.2 プロビジョニング Web サービスの Java クライアント

この節では、ワークフローシステム内のすべてのプロセスを表示する、プロビジョニング Web サービス用の単純な Java クライアントの開発方法について説明します。このクライアントの完全なソースコードについては、385 ページの 「Java クライアントのサンプルコード」を参照してください。

前提条件

Java クライアントを開発するにはサポートする Java Developer’s Kit をインストールする必要があります (「Identity Manager のシステム要件 (http://www.novell.com/documentation/idm35/index.html?page=/documentation/idm35/install/data/b2mbjps.html)」を参照)。また、クライアントプログラムには、次の JAR ファイルが必要です。

activation.jar
commons-httpclient.jar
IDMfw.jar
log4j.jar
saaj-api.jar
wssdk.jar
commons-codec-1.3.jar
commons-logging.jar
jaxrpc-api.jar
mail.jar
workflow.jar

xpp3.jar

Java クライアントの開発

Web サービスにアクセスするクライアントの開発は、次の 2 つのステップで構成されます。

- リモートサービスを表すオブジェクトのスタブを取得する
- リモートサービスで利用できる 1 つまたは複数の操作を起動する

Web サービス用の Java プログラミングモデルは、RMI ととてもよく似ています。まず、最初のステップでは、JNDI を使ってスタブをルックアップします。

```
InitialContext ctx = new InitialContext();
ProvisioningService service = (ProvisioningService)
ctx.lookup("xmlrpc:soap:com.novell.soa.af.impl.soap.ProvisioningServic
e");
Provisioning prov = service.getProvisioningPort();
```

コードの最初の行は、JNDI ルックアップの初期コンテキストを作成します。2 行目は、プロビジョニング Web サービスのスタブを取得するために使用できるサービスオブジェクトをルックアップしています。最後の行は、サービスからプロビジョニングスタブを取得しています。

プロビジョニングスタブで操作を起動する前に、サービスの認証に使用する資格情報やエンドポイント URL などの、いくつかのプロパティを設定する必要があります。

```
Stub stub = (Stub) prov;
// set username and password
stub._setProperty(Stub.USERNAME_PROPERTY, USERNAME);
stub._setProperty(Stub.PASSWORD_PROPERTY, PASSWORD);
// set the endpoint URL
stub._setProperty(Stub.ENDPOINT_ADDRESS_PROPERTY, url);
```

これらのプロパティと他のスタブプロパティの詳細は、383 ページの 「よく使用されるスタブ定数」を参照してください。スタブを正しく環境設定したら、getAllProcesses 操作を起動して、コンソールに返される各プロセスの情報をダンプできます。

```
// invoke the getAllProcesses method
ProcessArray array = prov.getAllProcesses();
Process[] procs = array.getProcess();
// print process array
System.out.println("list of all processes:");
if (procs != null) {
for (int i = 0; i < procs.length; i++) {
System.out.println(" process with request identifier " +
procs[i].getRequestId());
System.out.println(" initiator = " + procs[i].getInitiator());
System.out.println(" recipient = " + procs[i].getRecipient());
System.out.println(" processId = " + procs[i].getProcessId());
System.out.println(" created = " +
8
9
procs[i].getCreationTime().getTime());
if (null != procs[i].getCompletionTime()) {
System.out.println(" completed = " +
procs[i].getCompletionTime().getTime());
```

```
}
System.out.println(" approval status = " +
procs[i].getApprovalStatus());
System.out.println(" process status = " +
procs[i].getProcessStatus());
if (i != procs.length - 1)
System.out.println();
}
}
```

スタブのメソッド起動により、SOAP メッセージが HTTP トランスポートを使って、プロビジョニング Web サービスに送信されます。引数のある操作に対して、スタブはそれらの Java オブジェクトの XML への処理をマーシャルします。Web サービスは SOAP メッセージを返し、スタブ XML をアンマーシャルします。この場合、これが ProcessArray Java オブジェクトに変換されます。

クライアントの実行

サンプルの ANT ビルドファイルには、クライアントを実行するためのターゲットがあります (388 ページの 「サンプルの Ant ファイル」を参照)。クライアントでは、CLASSPATH 内に 381 ページの 「前提条件」に説明されている JAR ファイルが必要です。プロビジョニング Web サービス SOAP エンドポイント用の別のデフォルトアドレスをコードに指定したり、単にコマンドラインの引数として指定することができます。例 :

```
ant -Durl=http://www.company.com:80/IDMProv/provisioning/service run
```

よく使用されるスタブ定数

com.novell.soa.ws.portable スタブクラス (WSSDK の一部) は、スタブインスタンスの環境設定に使用できるさまざまなプロパティをサポートしています (たとえば、HTTP 通信をきめ細かくチューニングするため)。これらのプロパティの中で、よく使われる一部のプロパティを次の表に示します。

表 20-2 *プロビジョニング Web サービスのスタブ定数*

プロパティ	タイプ	説明
ENDPOINT_ADDRESS_PROPERTY	java.lang.String	Web サービスの URL です。サーバの要件に応じて URL プロトコルスキーマは、HTTP または HTTPS を使用できます。パスの部分は、次のようにする必要があります。 `/IDMProv/provisioning/service`
HTTP_HEADERS	java.util.Map	文字列名と値のペアとなる、追加の HTTP ヘッダです。
HTTP_TIME_OUT	java.lang.Integer	ホストとの接続を確立するために待機する秒数です。ここに指定した秒数を経過しても接続できない場合は、タイムアウトになります。
HTTP_MAX_TOTAL_CONNECTIONS	java.lang.Integer	このクライアントプログラムが、アクセスするすべてのサーバに対して確立できる、同時接続数です。デフォルトは 20 です。

プロパティ	タイプ	説明
HTTP_MAX_HOST_CONNECTIONS	java.lang.Integer	このクライアントプログラムが、個別の 1 台のサーバホストに対して確立できる、同時接続数です。デフォルトは 2 です。ここに指定する値は、HTTP_MAX_TOTAL_CONNECTIONS に指定する値以下でなければなりません。クライアントからサーバへの接続が 20 以上必要な場合は、HTTP_MAX_TOTAL_CONNECTIONS にもそれ以上の値を指定する必要があります。
USERNAME	java.lang.String	HTTP 認証のユーザ ID です。
PASSWORD	java.lang.String	HTTP 認証のパスワードです。
HTTP_PROXY_HOST	java.lang.String	プロキシのホスト DNS 名です。このプロパティを設定する場合、HTTP_PROXY_PORT も設定する必要があります。
HTTP_PROXY_PORT	java.lang.Integer	プロキシで使用するポートです。このプロパティを設定する場合、HTTP_PROXY_HOST も設定する必要があります。
HTTP_PROXY_AUTH_SCHEME	java.lang.Integer	プロキシで使用する認証スキーマです (Basic または Digest)。
HTTP_PROXY_USERNAME	java.lang.String	プロキシを使って HTTP 認証を行うためのユーザ ID です。
HTTP_PROXY_PASSWORD	java.lang.String	プロキシを使って HTTP 認証を行うためのパスワードです。

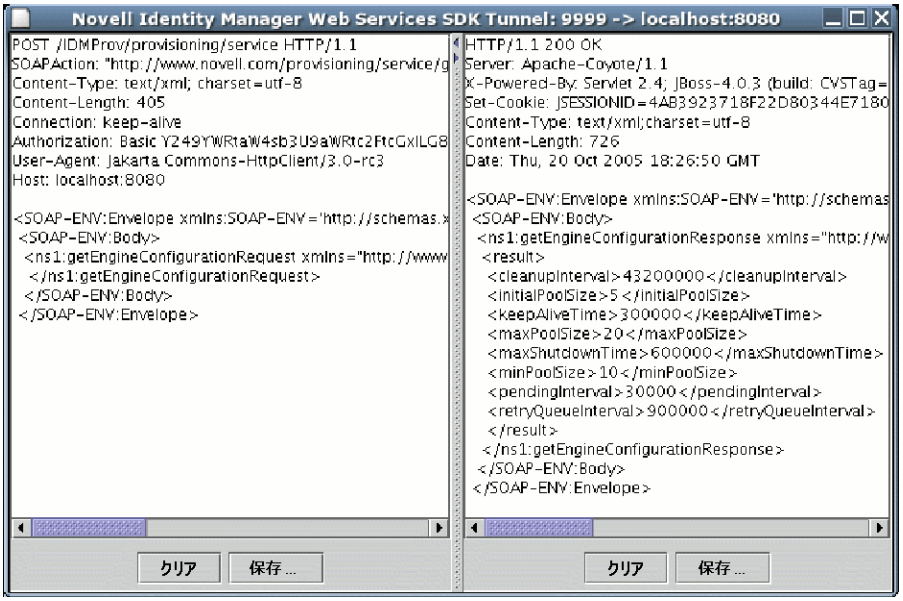
TCP トンネル

TCP トンネルは、クライアントとサーバ間で交わされる SOAP メッセージを参照する場合に役立つツールです。ANT ビルドファイル (388 ページの 「サンプルの Ant ファイル」を参照) には、トンネルを開始するターゲットがあります。トンネルを開始したら、トンネルが待機するポートと、リモート Web サービスのホスト／ポートを入力する必要があります。トンネルのデフォルトでは、ポート 9999 で待機し、localhost のポート 8080 で動作しているサービスに接続します。クライアントプログラム (382 ページの 「Java クライアントの開発」を参照) は、最初のコマンドラインパラメータを使って ENDPOINT_ADDRESS_PROPERTY を設定します。トンネルの開始後は、デフォルト値と次のコマンドを使って、クライアントを実行することができます。

```
ant -Durl=http://localhost:9999/IDMProv/provisioning/service run
```

図 20-4 に、左側のパネルに SOAP メッセージ要求、右側のパネルにメッセージがある TCP トンネルを示します。

図 20-4 TCP トンネル



Java クライアントのサンプルコード

ワークフローシステム内のすべてのプロセスを表示する、Java クライアントのコード例を次に示します。

```
package com.novell.examples;import javax.naming.InitialContext;
import com.novell.soa.af.impl.soap.AdminException;
import com.novell.soa.af.impl.soap.Process;
import com.novell.soa.af.impl.soap.ProcessArray;
import com.novell.soa.af.impl.soap.Provisioning;
import com.novell.soa.af.impl.soap.ProvisioningService;
import com.novell.soa.ws.portable.Stub;
public class Client
{
private static final String USERNAME = "admin";
private static final String PASSWORD = "test";
public static void main(String[] args)
{
try {
String url = args.length > 0 ? args[0] :
"http://localhost:8080/IDMProv/provisioning/service";
listProcesses(url);
} catch (AdminException ex) {
System.out.println("command failed: " + ex.getReason());
} catch (Exception ex) {
ex.printStackTrace();
}
}
private static void listProcesses(String url)
throws Exception
{
// get the stub
InitialContext ctx = new InitialContext();
```

```
ProvisioningService service = (ProvisioningService)
ctx.lookup("xmlrpc:soap:com.novell.soa.af.impl.soap.ProvisioningServic
e");
Provisioning prov = service.getProvisioningPort();
Stub stub = (Stub) prov;
// set username and password
stub._setProperty(Stub.USERNAME_PROPERTY, USERNAME);
stub._setProperty(Stub.PASSWORD_PROPERTY, PASSWORD);
// set the endpoint URL
stub._setProperty(Stub.ENDPOINT_ADDRESS_PROPERTY, url);
// invoke the getAllProcesses method
ProcessArray array = prov.getAllProcesses();
Process[] procs = array.getProcess();
// print process array
System.out.println("list of all processes:");
if (procs != null) {
for (int i = 0; i < procs.length; i++) {
System.out.println(" process with request identifier " +
procs[i].getRequestId());
System.out.println(" initiator = " + procs[i].getInitiator());
System.out.println(" recipient = " + procs[i].getRecipient());
System.out.println(" processId = " + procs[i].getProcessId());
System.out.println(" created = " +
procs[i].getCreationTime().getTime());
if (null != procs[i].getCompletionTime()) {
System.out.println(" completed = " +
procs[i].getCompletionTime().getTime());
}
17
18
System.out.println(" approval status = " +
procs[i].getApprovalStatus());
System.out.println(" process status = " +
procs[i].getProcessStatus());
if (i != procs.length - 1)
System.out.println();
}
}
}
}
```

20.2.3 Mono クライアントの開発

前の節では、Identity Manager の Web サービスツールキットと事前コンパイルされたスタブコードを使った、Java クライアントの作成方法を説明しました。この節では、プロビジョニング Web サービス用 WSDL を使ったクライアントの開発方法を説明します。この例では Mono を使用して、完了したワークフローのデフォルト保持時間 120 日を 30 日に変更する C# クライアントを作成しています。

前提条件

作業を開始するには、まず Mono をダウンロードしてシステムにインストールする必要があります (Mono Project Website (http://www.mono-project.com/) を参照)。このマニュアル作

成時に公開されているバージョンの Mono は、nillable 属性が true に設定された要素を持つ複雑なスキーマタイプはサポートされていません。プロビジョニング WSDL ではこの構造が用いられているため、Provisioning.WSDL ファイルを手動で編集し、「nillable="true"」が使われている 3ヶ所を削除する必要があります。

スタブの生成

382 ページの 「Java クライアントの開発」で説明した Java クライアントの開発と比べて、C# クライアントの作成にはもう 1 つの作業を行う必要があります。Web サービス SOAP エンドポイントへのアクセス用スタブが用意されていないため、WSDL ドキュメントからスタブを生成する必要があります。Mono には、WSDL ファイルを処理してスタブを作成する、wsdl と呼ばれるコンパイラが用意されています。WSDL ファイルは、次の URL にアクセスしてユーザアプリケーションサーバからダウンロードできます。

```
http://myserver:8080/IDMProv/provisioning/service?wsdl
```

ここで、「myserver」にはサーバ名を、「IDMProv」にはユーザアプリケーション WAR ファイル名を指定します。

WSDL ファイルをコンパイルするには、次のコマンドを使用します。

```
wsdl Provisioning.wsdl
```

これにより、ProvisioningService.cs と言う名前の C# ファイルが生成されます。このファイルは、次の Mono C# コンパイラコマンドを使って DLL にコンパイルする際に必要になります。

```
mcs /target:library /r:System.Web.Services.dll ProvisioningService.cs
```

Java クライアントに比べると、ProvisioningService.dll ファイルは workflow.jar と同等のファイルです。このファイルには、プロビジョニング Web サービスにアクセスするためのスタブコードとサポートするクラスが含まれています。フロー保持時間を設定して、新しい値をコンソールに表示する、単純な C# クライアントのソースコード例を次に示します。

```
using System;
using System.Net;
class provclient {
public static void Main(string [] args) {
// create the provisioning service proxy
ProvisioningService service = new ProvisioningService();
// set the credentials for basic authentication
service.Credentials = new NetworkCredential("admin", "test");
service.PreAuthenticate = true;
// set the value for completed request retention to 30 days
setCompletedProcessTimeoutRequest req = new
setCompletedProcessTimeoutRequest();
11
12
req.arg0 = 30;
service.setCompletedProcessTimeout(req);
// display the new value on the console
getCompletedProcessTimeoutResponse res =
service.getCompletedProcessTimeout(new
getCompletedProcessTimeoutRequest());
Console.WriteLine(res.result);
```

```
}
}
```

このファイルは、展開されている Identity Manager システムの管理者資格情報を使って編集する必要があります。クライアントをコンパイルするには、次のコマンドを使用します。

```
mcs /r:ProvisioningService.dll /r:System.Web provclient.cs
```

これにより、provclient.exe ファイルが生成されます。

クライアントの実行

クライアントを実行するには、次のコマンドを使用します。

```
mono provclient.exe
```

20.2.4 サンプルの Ant ファイル

このサンプルの Ant ファイルには、Identity Manager インストールからの必要な JAR ファイルの抽出、Java クライアントのコンパイルと実行、および TCP トンネルの開始に役立つターゲットが含まれています。

```
<?xml version="1.0"?>
<project name="client" default="all" basedir=".">
<target name="all" depends="clean, extract, compile"></target>
<!-- main clean target -->
<target name="clean">
<delete quiet="true" dir="classes"/>
<delete quiet="true" dir="lib"/>
</target>
<!-- init sets up the build environment -->
<target name="init">
<mkdir dir="classes"/>
<copy todir="${basedir}/lib">
<fileset dir="${basedir}" includes="log4j.properties"/>
</copy>
<!-- classpath -->
<path id="CLASSPATH">
<pathelement location="${basedir}/classes"/>
<fileset dir="${basedir}/lib" includes="*.jar"/>
</path>
</target>
<!-- extract -->
<target name="extract">
<property name="idm.home" value="/opt/novell/idm3"/>
<property name="jboss.lib" value="${idm.home}/jboss-4.0.3/server/
IDMProv/lib"/>
<mkdir dir="lib"/>
<unzip src="${idm.home}/IDMProv.war" dest="${basedir}/lib">
<patternset>
<include name="WEB-INF/lib/commons-codec-1.3.jar"/>
<include name="WEB-INF/lib/commons-httpclient.jar"/>
<include name="WEB-INF/lib/commons-logging.jar"/>
<include name="WEB-INF/lib/jaxrpc-api.jar"/>
<include name="WEB-INF/lib/saaj-api.jar"/>
```

```
<include name="WEB-INF/lib/xpp3.jar"/>
<include name="WEB-INF/lib/workflow.jar"/>
<include name="WEB-INF/lib/wssdk.jar"/>
<include name="WEB-INF/lib/IDMfw.jar"/></patternset>
</unzip>
<move todir="${basedir}/lib">
<fileset dir="${basedir}/lib/WEB-INF/lib" includes="*.jar"/>
</move>
<delete quiet="true" dir="${basedir}/lib/WEB-INF"/>
<copy todir="${basedir}/lib">
<fileset dir="${jboss.lib}" includes="activation.jar, mail.jar,
log4j.jar"/>
</copy>
</target>
18
19
<!-- tunnel -->
<target name="tunnel" depends="init">
<java classname="com.novell.soa.ws.impl.tools.tcptunnel.Tunnel"
fork="true"spawn="true">
<classpath refid="CLASSPATH"/>
</java>
</target>
<!-- compile -->
<target name="compile" depends="init">
<javac srcdir="${basedir}" destdir="classes"includes="Client.java">
<classpath refid="CLASSPATH"/>
</javac>
</target>
<!-- run -->
<target name="run" depends="init">
<property name="url" value="http://localhost:8080/IDMProv/
provisioning/service"/>
<java classname="com.novell.examples.Client" fork="true">
<arg line="${url}"/>
<classpath refid="CLASSPATH"/>
</java>
</target>
</project>
```

20.2.5 サンプルの log4J ファイル

次の log4j ファイルは、デフォルトログレベルを「error」に設定します。:

```
log4j.rootCategory=ERROR, R
log4j.appender.R=org.apache.log4j.ConsoleAppender
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R.layout.ConversionPattern=%-5p: %m%n
```

20.3 プロビジョニング Web サービスの API

この節では、プロビジョニング Web サービスのメソッドの詳細を説明します。

すべてのメソッドが、com.novell.soa.af.impl.soap.AdminException と java.rmi.RemoteException をスローします。読みやすくするために、メソッドの署名の throws 節は省略されています。

この項では、次のトピックについて説明します。

20.3.1 プロセス

この節は、各プロセスメソッドに関する参照情報を取り上げています。メソッドには次のものが含まれています。

getProcessesByQuery

プロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray
getProcessesByQuery(com.novell.soa.af.impl.soap.T_ProcessInfoQuery
query, int maxRecords)
```

例

```
        //
       // Query information about processes for a user that are running
and
        // have not been approved yet.
        String logic = "AND";
        T_ProcessInfoOrder order = T_ProcessInfoOrder.APPROVAL_STATUS;
        int CHOICE_SIZE = 4;
        Integer approvalStatusInteger = new
Integer(ProcessConstants.PROCESSING);
        Integer processStatusInteger = new
Integer(ProcessConstants.RUNNING);
        //
        // Setup the query with the above params
        T_ProcessInfoQueryChoice [] choice = new
T_ProcessInfoQueryChoice[CHOICE_SIZE];
        choice[0] = new T_ProcessInfoQueryChoice();
        choice[0].setApprovalStatus(approvalStatusInteger);
        choice[1] = new T_ProcessInfoQueryChoice();
        choice[1].setProcessStatus(processStatusInteger);
        choice[2] = new T_ProcessInfoQueryChoice();
        choice[2].setRecipient(recipient);
        choice[3] = new T_ProcessInfoQueryChoice();
        choice[3].setRequestId(requestId);
        int maxRecords = -1;
        T_ProcessInfoQuery processInfoQuery =                new
T_ProcessInfoQuery(T_Logic.fromString(logic), order, choice);
        ProcessArray processArray =
stub.getProcessesByQuery(processInfoQuery, maxRecords);
```

getProcessesByStatus

指定したステータス (たとえば、実行中のプロセスなど) のプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
public com.novell.soa.af.impl.soap.ProcessArray
getProcessesByStatus(com.novell.soa.af.impl.soap.T_ProcessStatus
status)
```

例

```
      T_ProcessStatus processStatus = T_ProcessStatus.Running;
      //
      // Get processes by status
       ProcessArray processArray =
stub.getProcessesByStatus(processStatus);
       Process [] process = processArray.getProcess();
```

getProcesses

プロセス ID で指定したプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray getProcesses(java.lang.String
id, long time,com.novell.soa.af.impl.soap.T_Operator op,
java.lang.String initiator, java.lang.String recipient)
```

パラメータ

パラメータ	説明
processId	プロセス ID です (java.lang.String)。
creationTime	プロセスが開始された時刻です (long)。
op	使用する演算子です。次の演算子を使用できます。 EQ - 等しい LT - 未満 LE - 以下 GT - より大きい GE - 以上
イニシエータ	ワークフローのイニシエータです。
recipient	承認アクティビティの受信者です。

例

```
       int processMatchCount = 0;
       T_Operator operator = T_Operator.GT;
       long currentTimeInMillis = System.currentTimeMillis();
       String [] requestIds = requestIdArray.getString();
       //
       // Initialize and start a provisioning request
       HashMap provMap = new HashMap();
       provMap.put(Helper.RECIPIENT, recipient);
       provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
       //
       // Start request
       // Calls method startProvisioningRequest on the provUtils
       // utility object which refers to a utility class that does not
       // ship with the Identity Manager User Application.
String requestId = provUtils.startProvisioningRequest(provMap, null);
       sleep(5);
       Process process = stub.getProcess(requestId);
       if(process != null)
       {
            String processId = process.getProcessId();
            String initiator = process.getInitiator();

            ProcessArray processArray = stub.getProcesses(processId,
currentTimeInMillis, operator, initiator, recipient);
       }
```

getAllProcesses

実行中および完了したすべてのプロビジョニング要求に関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray getAllProcesses()
```

例

```
    ProcessArray array = stub.getAllProcesses();
    Process [] processes = array.getProcess();
     if(_process != null)
    {
        sb = new StringBuffer();
        sb.append("\nProcess List:");
        for(int index = 0; index < _process.length; index++)
        {
            String processId = _process[index].getProcessId();
            String approvalStatus =
_process[index].getApprovalStatus();
            Calendar completionTime =
_process[index].getCompletionTime();
            Calendar creationTime = _process[index].getCreationTime();
            String engineId = _process[index].getEngineId();
            String proxy = _process[index].getProxy();
            String initiator = _process[index].getInitiator();
            String processName = _process[index].getProcessName();
            String processStatus = _process[index].getProcessStatus();
            String p_recipient = _process[index].getRecipient();
            String p_requestId = _process[index].getRequestId();
            int valueOfapprovalStatus =
_process[index].getValueOfApprovalStatus();
            int valueOfprocessStatus =
_process[index].getValueOfProcessStatus();
            String version = _process[index].getVersion();
        }
```

getProcessesArray

返されるプロセス数を制限する場合に使用します。ここに指定した値がシステムの制限値よりも小さい場合は、ここに指定した数のプロセスが返されます。システム制限値を超える値を指定した場合は、システム制限値に設定された数のプロセスが返されます。0 以下の値を指定した場合は、すべてのプロセスが返されます。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray getProcessesArray(int
maxRecords);
```

例

```
    /**
     * Method to augment the getAllProcesses() method that impose
limits
     * on the number of processes returned.
```

```
     * @throws TestProgramException
     */
    public void adding_Limits_To_getProcessArray_TestCase()
    throws TestProgramException
    {
        String recipient =
ServiceUtils.getInstance().getLoginData().getUsername(LoginData.RECIPI
ENT_TYPE);
        String requestNameToStart =
provUtils.getProvisioningResourceNameForRecipient(recipient,
"Enable Active Directory");
        //
        // Get the stub
        Provisioning stub =
ServiceUtils.getInstance().getProvisioningStub();
        try
        {
            //
            // Start multiple requests
            final int NUMBER_OF_REQUESTS_TO_START = 2;

            Map map = MapUtils.createAndSetMap(new Object[] {
                    Helper.RECIPIENT, recipient,

IProvisioningConstants.PROVISIONING_REQUEST_TO_START,
requestNameToStart});
            //
            // Start request(s)
            StringArray requestIdArray =
               provUtils.startMultipleProvisioningRequests(map, null,
NUMBER_OF_REQUESTS_TO_START);
            LoggerUtils.sleep(3);
            LoggerUtils.sendToLogAndConsole("Started " +
NUMBER_OF_REQUESTS_TO_START + " provisioning requests");
            //
            // New method to limit the number of processes returned
            //
            // Test Results : maxProcesses <= 0 returns all processes
            //                maxProcesses up to system limit returns
maxProcess count
           //               maxProcesses > system limit returns system
limit
            int maxProcesses = 10;
            ProcessArray processArray =
stub.getProcessesArray(maxProcesses);
            Process [] processes = processArray.getProcess();
            if(processes != null)
            {
                LoggerUtils.sendToLogAndConsole("Process count
returned: " + processes.length);
                Assert.assertEquals("Error: Processes returned
shouldn't exceed max count.",
                        maxProcesses, processes.length);
            }
```

```
        }
        catch(AdminException error)
        {
            RationalTestScript.logError(error.getReason() );
            throw new TestProgramException(error.getReason() );
        }
        catch(RemoteException error)
        {
            RationalTestScript.logError(error.getMessage() );
            throw new TestProgramException(error.getMessage() );
        }
    }
```

getProcessesById

プロセス ID で指定したプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray
getProcessesById(java.lang.String id)
```

例

```
    Process [] allProcesses = stub.getAllProcesses().getProcess();
    if(allProcesses != null)
    {
          String processId = allProcesses[0].getProcessId;
          ProcessArray array = stub.getProcessesById(processId);
          Process [] processes = array.getProcess();
     }
```

terminate

実行中のプロビジョニング要求を停止する場合に使用します。

メソッドの署名

```
void terminate(java.lang.String requestId,
com.novell.soa.af.impl.soap.T_TerminationType state, java.lang.String
comment)
```

パラメータ

パラメータ	説明
requestId	プロビジョニング要求の ID です。
都道府県	プロセス停止の理由です。次の選択肢があります。
	RETRACT
	エラー
comment	停止アクションに関するコメントを追加します。

例

```
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
String requestId = provUtils.startProvisioningRequest(provMap, null);
     sleep(5);
     //
     // Now retract the request
     T_TerminationType terminationType = T_TerminationType.RETRACT;
    stub.terminate(requestId, terminationType,
terminationType.getValue() + " the request");
```

getProcess

要求 ID で指定する、動作中または完了したプロビジョニング要求に関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.Process getProcess(java.lang.String
requestId)
```

例

```
     //     // Initialize and start a provisioning request     HashMap
provMap = new HashMap();    provMap.put(Helper.RECIPIENT, recipient);
provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");     //     // Start
request
      // Calls method startProvisioningRequest on the provUtils      /
/ utility object which refers to a utility class that does not       /
/ ship with the Identity Manager User Application.     String requestId
= provUtils.startProvisioningRequest(provMap, null);     sleep(5);
Process process = stub.getProcess(requestId);
    if(process != null)
    {
        boolean bMatchProcess = false;
        if( (recipient.compareTo(process.getRecipient()) == 0) &&
(requestId.compareTo(process.getRequestId()) == 0) )
        {
            bMatchProcess = true;
        }
        if(bMatchProcess)
        {
            String msg = "Found process with requestId : " + requestId;
            LoggerUtils.sendToLogAndConsole(msg);
        }
```

```
        //
        // Assert if we could not find a match
       Assert.assertTrue("Could not find process with request id: " +
requestId, bMatchProcess);
    }
```

getProcessesByCreationTime

ワークフロープロセスの作成時刻から現在の時刻までに作成されたプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray
getProcessesByCreationTime(long time,
com.novell.soa.af.impl.soap.T_Operator op)
```

パラメータ

パラメータ	説明
creationTime	プロセスが開始された時刻です。
op	使用する演算子です。次の演算子を使用できます。 EQ - 等しい LT - 未満 LE - 以下 GT - より大きい GE - 以上

例

```
     T_Operator operator = T_Operator.GT;
    //
    // Get processes with operator relative to the current time
    long currentTime = System.currentTimeMillis();//
currentDateTime.getTime();
    ProcessArray processArray =
stub.getProcessesByCreationTime(currentTime, operator);
```

getProcessesByApprovalStatus

指定した承認ステータス (Approved(承認)、Denied(拒否)、Retracted(撤回)) のプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray
getProcessesByApprovalStatus(com.novell.soa.af.impl.soap.T_ApprovalSta
tus status)
```

例

```
    T_ApprovalStatus approvalStatus = T_ApprovalStatus.Approved;
    //
    // Get all the processes based upon approval status above
```

```
    ProcessArray processArray =
stub.getProcessesByApprovalStatus(approvalStatus);
    Process [] processes = processArray.getProcess();
```

getProcessesByRecipient

特定の受信者 ID を持つプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray
getProcessesByRecipient(java.lang.String recipient)
```

例

```
    String recipient = "cn=ablake,ou=users,ou=idmsample-
komodo,o=novell";
    //
    // Get processes by recipient
    ProcessArray processArray =
stub.getProcessesByRecipient(recipient);
    Process [] process = processArray.getProcess();
```

getProcessesByInitiator

特定のイニシエータ ID を持つプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray
getProcessesByInitiator(java.lang.String initiator)
```

例

```
    String initiator = "cn=admin,ou=idmsample-komodo,o=novell";

    //
    // Get processes by initiator
    ProcessArray processArray =
stub.getProcessesByInitiator(initiator);
    Process [] process = processArray.getProcess();
```

setResult

以前に完了したプロビジョニング要求のエンタイトルメント結果 (承認ステータス) を設定する場合に使用します。

メソッドの署名

```
void setResult(java.lang.String requestId,
com.novell.soa.af.impl.soap.T_EntitlementState state,
com.novell.soa.af.impl.soap.T_EntitlementStatus status,
java.lang.String message)
```

パラメータ

パラメータ	説明
requestId	プロビジョニング要求の ID です。
都道府県	プロビジョニング要求の状態です次の値を使用できます。 Unknown (不明)、 Granted (認可)、 Revoked (取り消し)
status	プロビジョニング要求のステータスです次の値を使用できます。 Unknown (不明)、 Success (成功)、 Warning (警告)、 Error (エラー)、 Fatal (致命的エラー)、 Submitted (送信済み)
メッセージ	エンタイトルメント結果に関するメッセージです。

例

```
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);

     //
     // Get the process id for this running process
     Process process = stub.getProcess(requestId);
     String processId = null;
     if (process != null)
        processId = process.getProcessId();
    //
    // Reset the state of the provisioning request
    T_EntitlementState newEntitlementState =
T_EntitlementState.Revoked;
    T_EntitlementStatus newEntitlementStatus =
T_EntitlementStatus.Success;
    String comment = "Revoked the provisioning request";
```

```
    stub.setResult(processId, newEntitlementState,
newEntitlementStatus, comment);
```

getProcessesByCreationInterval

2 つの時間で指定された期間内に開始されたプロセスに関する情報を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProcessArray
getProcessesByCreationInterval(long start, long end)
```

パラメータ

パラメータ	説明
startTime	開始時刻 (YYYY/MM/DD) です。
endTime	終了時刻 (YYYY/MM/DD) です。

例

```
     long startTime = System.currentTimeMillis();
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);

    long endTime = System.currentTimeMillis();
    //
    // Get all the processes between the start and end time
ProcessArray processArray =
stub.getProcessesByCreationInterval(startTime, endTime);
    Process [] processes = processArray.getProcess();
```

20.3.2 プロビジョニング

この節は、各プロビジョニングメソッドに関する参照情報を取り上げています。プロビジョニングメソッドには次のものが含まれています。

- 401 ページの 「multiStart」
- 402 ページの 「start」
- 404 ページの 「getAllProvisioningRequests」

- 405 ページの 「getProvisioningRequests」
- 406 ページの 「getProvisioningCategories」
- 406 ページの 「」
- 407 ページの 「getProvisioningStatuses」
- 409 ページの 「startWithDigitalSignature」
- 411 ページの 「startAsProxyWithDigitalSignature」
- 412 ページの 「startWithCorrelationId」

multiStart

指定した各受信者に対するワークフロー要求を開始する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.StringArray multiStart(java.lang.String
processId, com.novell.soa.af.impl.soap.StringArray recipients,
com.novell.soa.af.impl.soap.DataItemArray items)
```

パラメータ

パラメータ	説明
processId	開始するプロビジョニング要求の ID です。
受信者	各受信者の DN です。
dataItem	プロビジョニング要求のデータ項目のリストです。

例

```
     ProvisioningRequestArray requestArray =
stub.getAllProvisioningRequests(recipient);
    //
    // If there are some then,
    if(requestArray != null)
    {
        String Id = " ";
        StringArray requestIdStringArray = null;
        String [] listOfRecipients = {recipient, addressee};
        //
        // Select a provisioning resource
        String requestNameToStart = "Enable Active Directory Account
(Mgr Approve-No Timeout)";
        //
        // Loop thru and find the request that we want to start
        ProvisioningRequest [] requests =
requestArray.getProvisioningrequest();
        for(int index = 0; index < requests.length; index++)
        {
            //
            // Is this the name of the request to start?
           if(requests[index].getName().compareTo(requestNameToStart)
== 0)
```

```
            {
                //
                // Get the current associated data items. Replicate a
new
                // dataitem array excluding the null values.
                Id = requests[index].getId();
                DataItem [] dataItem =
requests[index].getItems().getDataitem();
                if(dataItem != null)
                // Call method replicateDataItemArray on the
                // provUtils utility object, which refers to a
                // utility class that does not ship with the
                // Identity Manager User Application.
                {
                    DataItemArray newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
                    //
                   // Create a string array initializing with multiple
recipients
                    StringArray listOfRecipientsStringArray = new
StringArray(listOfRecipients);
                    //
                   // Start the request for multiple recipients
                    logStep("Calling stub.multiStart(" + Id +
",listOfRecipientsStringArray,newDataItemArray)");
                    requestIdStringArray = stub.multiStart(Id,
listOfRecipientsStringArray, newDataItemArray);
            }
        }
}
```

start

プロビジョニング要求を開始する場合に使用します。

メソッドの署名

```
java.lang.String start(java.lang.String processId, java.lang.String
recipient, com.novell.soa.af.impl.soap.DataItemArray items)
```

パラメータ

パラメータ	説明
processId	開始するプロビジョニング要求の ID です。
recipient	各受信者の DN です。
dataItem	プロビジョニング要求のデータ項目のリストです。

例

```
        //
        // Initialize and start a provisioning request
        HashMap provMap = new HashMap();
```

```
       provMap.put(Helper.RECIPIENT, recipient);
       provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
       //
       // Start request
       // Calls method startProvisioningRequest on the provUtils
       // utility object which refers to a utility class that does not
       // ship with the Identity Manager User Application.
       String requestId = provUtils.startProvisioningRequest(provMap,
null);
       sleep(5);
```

上記の例では、startProvisioningRequest メソッドを呼び出しています。このメソッドは、IDM ユーザアプリケーションの一部ではありません。ここでは、例を完了させるためにこれを使用しています。

```
          /**
           *Method to start a provisioning request using the supplied
           *Map and dataitem object. Handling of digital certificate
           *resources is also handled.
           * @param _map
           * @param _in_dataItem
           * @return String
           * @throws TestProgrammException
           */
          public String startProvisioningRequest(Map _map, DataItem []
          _in_dataItem) throws TestProgramException
         {
            String requestId = null;
            try
            {
               String recipient  =(String)_map.get(Helper.RECIPIENT);
               String requestToStart =
(String)_map.get(IProvisioningConstants.PROVISIONING_REQUEST_TO_START)
;
               String proxyUser
=(String)_map.get(IWorkFlowConstants.PROXY_USER);
               String digitalSignature =
String)_map.get(IDigitalSignatureConstants.DIGITAL_SIGNATURE);
               RationalTestScript.logInfo("Step: Calling
startProvisioningRequest(_map)");
               //
               //Get the stub
               Provisioning stub =
ServiceUtils.getInstance().getProvisioningStub();
               //
       //Get all the available resource requests for the recipient
       RationalTestScript.logInfo("Step: Calling
stub.getAllProvisioningRequests(" + recipient + ")");
       ProvisioningRequestArray requestArray =
stub.getAllProvisioningRequests(recipient);
       if(requestArray != null)
       {
           //
           //Get the provisioning request from the array
```

```
            ProvisioningRequest request =
getProvisioningRequestFromArray(requestArray, requestToStart);
            if(request != null)
        {
        DataItem [] dataItem = null;
        DataItemArray newDataItemArray = null;
        //
        // If the supplied data item is null then just replicate
        // what currently exists with the request.
        if(_in_dataItem == null)
        {
            //
            // Use the current data item associated with the request
            dataItem = request.getItems().getDataitem();
            if(dataItem != null)
            {
                newDataItemArray = replicateDataItemArray(dataItem);
            }
        }
        else
        {
            //
            // Set the incoming data item array
            newDataItemArray = new DataItemArray();
            newDataItemArray.setDataitem(_in_dataItem);
        }
        //
        // Start the Provisioning request for the recipient
        if(proxyUser == null && digitalSignature == null)
           {
              RationalTestScript.logInfo("Step: Calling stub.start(" +
request.getId() + "," + recipient + "dataItemArray)");
                          requestId = stub.start(
                                  request.getId(),
                                  recipient,
                                  newDataItemArray);
           }
            else if(proxyUser != null && digitalSignature == null)
            }
                  .
            .
           .
```

getAllProvisioningRequests

利用可能なプロビジョニング要求の配列を返す場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProvisioningRequestArray
getAllProvisioningRequests(java.lang.String recipient)
```

例

```
    //
    // Get all the provisioning requests for this recipient

    ProvisioningRequestArray provReqArray =
stub.getAllProvisioningRequests(recipient);
    ProvisioningRequest [] provRequest =
provReqArray.getProvisioningrequest();
    if(provRequest != null)
    {
        String description = provRequest[0].getDescription();
        String category = provRequest[0].getCategory();
        String digitialSignatureType =
provRequest[0].getDigitalSignatureType();
        String requestId = provRequest[0].getId();
        DataItemArray itemArray = provRequest[0].getItems();
        String legalDisclaimer = provRequest[0].getLegalDisclaimer();
        String name = provRequest[0].getName();
        String operation = provRequest[0].getOperation();
    }
```

getProvisioningRequests

特定のカテゴリや操作のプロビジョニング要求の配列を返す場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProvisioningRequestArray
getProvisioningRequests(java.lang.String recipient, java.lang.String
category, java.lang.String operation)
```

パラメータ

パラメータ	説明
recipient	プロビジョニング要求の受信者です。
category	プロビジョニング要求のカテゴリです。
operation	プロビジョニング要求の操作です (0= 認可、1= 取り消し、2= 両方)。

例

```
    String operation = IProvisioningRequest.GRANT;
    try
    {
        //
        // Get the stub
        Provisioning stub =
ServiceUtils.getInstance().getProvisioningStub();
        logStep("Calling stub.getProvisioningCategories()");
        StringArray categoriesStringArray =
stub.getProvisioningCategories();
        String [] categories = categoriesStringArray.getString();
```

```
        //
        // Loop thru and get the provisioning requests for each
category
        for(int index = 0; index < categories.length; index++)
        {
            //
            // Get the provisioning request based upon recipient
           logStep("Calling stub.getProvisioningRequests(" + recipient
+ "," + categories[index] + "," + operation + ")");
            ProvisioningRequestArray provRequestArray =
stub.getProvisioningRequests(recipient, categories[index], operation);
            ProvisioningRequest [] provRequests =
provRequestArray.getProvisioningrequest();
        }
```

getProvisioningCategories

メソッドの署名

```
com.novell.soa.af.impl.soap.StringArray getProvisioningCategories()
```

例

```
       StringArray categoriesStringArray =
stub.getProvisioningCategories();
       String [] categories = categoriesStringArray.getString();
```

メソッドの署名

```
java.lang.String startAsProxy(java.lang.String processId,
java.lang.String recipient, com.novell.soa.af.impl.soap.DataItemArray
items, java.lang.String proxyUser)
```

パラメータ

パラメータ	説明
processId	プロビジョニング要求の ID です。
recipient	プロビジョニング要求の受信者です。
項目	プロビジョニング要求のデータ項目です。
proxyUser	代理人ユーザの DN です。

例

```
    ProvisioningRequestArray requestArray =
stub.getAllProvisioningRequests(recipient);
    //
    // If there are some then,
```

```
    if(requestArray != null)
    {
        String Id = " ";
        String requestId = " ";
        String requestNameToStart = "Enable Active Directory Account
(Mgr Approve-No Timeout)";
        //
        // Loop thru and find the request that we want to start
        ProvisioningRequest [] requests =
requestArray.getProvisioningrequest();
        for(int index = 0; index < requests.length; index++)
        {
            //
            // Is this the name of the request to start?
           if(requests[index].getName().compareTo(requestNameToStart)
== 0)
            {
                //
                // Get the current associated data items. Replicate a
new
                // dataitem array excluding the null values.
                Id = requests[index].getId();
                DataItem [] dataItem =
requests[index].getItems().getDataitem();
                if(dataItem != null)
                {
                    // Call method replicateDataItemArray on the
                    // provUtils utility object, which refers to a
                    // utility class that does not ship with the
                    // Identity Manager User Application.
                    DataItemArray newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
                    //
                   // Start the Provisioning request for the recipient
                    logStep("Calling stub.startAsProxy(" + Id + "," +
recipient + ",newDataItemArray," + proxyUser + ")");
                    requestId = stub.startAsProxy(Id, recipient,
newDataItemArray, proxyUser);
                }
            }
        }
    }
```

getProvisioningStatuses

プロビジョニング要求のステータスを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.ProvisioningStatusArray
getProvisioningStatuses(com.novell.soa.af.impl.soap.T_ProvisioningStat
usQuery query, int maxRecords)
```

パラメータ

パラメータ	説明
クエリー	プロビジョニングステータスクエリを指定する場合に使用します。クエリには、次のコンポーネントがあります。 ◆ choice - 結果をフィルタリングする場合に使用するパラメータです。複数のパラメータを指定することができます。次のパラメータを使用できます。 Recipient - DN、 RequestID、 ActivityID、 Status (整数)、 State (整数)、 ProvisioningTime (YYYY/MM/DD)、 ResultTime (YYYY/MM/DD) ◆ logic - AND または OR です。 ◆ order - 結果のソート順を指定します。orderに指定できる値を次に示します。 ACTIVITY_ID、 RECIPIENT、 PROVISIONING_TIME 、 RESULT_TIME 、 STATE 、 STATUS 、 REQUEST_ID 、 MESSAGE 、
maxRecords	取得する最大レコード数を指定する場合に使用します。-1 を指定すると、返されるレコード数に制限はありません。

例

```
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);
     //
```

```
     //
    T_ProvisioningStatusQueryChoice [] choice = new
T_ProvisioningStatusQueryChoice[3];
    choice[0] = new T_ProvisioningStatusQueryChoice();
    choice[0].setRecipient(recipient);
    choice[1] = new T_ProvisioningStatusQueryChoice();
    choice[1].setRequestId(requestId);
    choice[2] = new T_ProvisioningStatusQueryChoice();
    choice[2].setStatus(new Integer(ProcessConstants.PROCESSING) );
    //
    // Initialize the query
    T_ProvisioningStatusQuery query = new
T_ProvisioningStatusQuery(T_Logic.AND,
T_ProvisioningStatusOrder.STATUS, choice);
    //
    // Make the query
    StringBuffer sb = new StringBuffer();
    int maxRecords = -1;

    ProvisioningStatusArray provStatusArray =
stub.getProvisioningStatuses(query, maxRecords);
```

startWithDigitalSignature

ワークフローを開始し、デジタル署名が必要なことを指定する場合に使用します。

メソッドの署名

```
java.lang.String startWithDigitalSignature(java.lang.String processId,
java.lang.String recipient, com.novell.soa.af.impl.soap.DataItemArray
items, java.lang.String digitalSignature,
com.novell.soa.af.impl.soap.SignaturePropertyArray
digitalSignaturePropertyArray)
```

パラメータ

パラメータ	説明
processId	要求の ID です。
recipient	要求の受信者です。
項目	プロビジョニング要求のデータ項目です。
デジタル署名	デジタル署名です。
digitalSignaturePropertyArray.	デジタル署名プロパティマップです。

例

```
    String recipient =
ServiceUtils.getInstance().getLoginData().getUsername(LoginData.RECIPI
ENT_TYPE);
    //
    // Get the digital signature string for admin
    String digitalSignature =
```

```
DigitalSignatureUtils.getDigitalSignatureFromFile(IDigitalSignatureCon
stants.ADMIN_DIGITAL_SIGNATURE_FILENAME);


    ProvisioningRequestArray requestArray =
stub.getAllProvisioningRequests(recipient);
    //
    // If there are some then,

   if(requestArray != null)
    {
        String Id = " ";
        String requestId = " ";
        String requestNameToStart = "Enable Active Directory Account
(Mgr Approve-No Timeout)";
        //
        // Loop thru and find the request that we want to start
        ProvisioningRequest [] requests =
requestArray.getProvisioningrequest();
        for(int index = 0; index < requests.length; index++)
        {
            //
            // Is this the name of the request to start?
           if(requests[index].getName().compareTo(requestNameToStart)
== 0)
            {
                //
                // Get the current associated data items. Replicate a
new
                // dataitem array excluding the null values.
                Id = requests[index].getId();
                DataItem [] dataItem =
requests[index].getItems().getDataitem();
                if(dataItem != null)
                {
                    // Call method replicateDataItemArray on the
                    // provUtils utility object, which refers to a
                    // utility class that does not ship with the
                    // Identity Manager User Application.
                    DataItemArray newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
                    //
                    // Start a digitally signed provisioning resource
for the recipient
                    requestId =
stub.startWithDigitalSignature(request.getId(), recipient,
newDataItemArray, digitalSignature, null); // Don't get any property
values (optional)
                }
            }
        }
    }
```

startAsProxyWithDigitalSignature

代理人をイニシエータとしてワークフローを開始し、デジタル署名が必要なことを指定する場合に使用します。

メソッドの署名

```
java.lang.String startAsProxyWithDigitalSignature(java.lang.String
processId, java.lang.String recipient,
com.novell.soa.af.impl.soap.DataItemArray items, java.lang.String
digitalSignature, com.novell.soa.af.impl.soap.SignaturePropertyArray
digitalSignaturePropertyArray, java.lang.String proxyUser)
```

パラメータ

パラメータ	説明
processId	要求の ID です。
recipient	要求の受信者です。
項目	プロビジョニング要求のデータ項目です。
デジタル署名	デジタル署名です。
digitalSignaturePropertyArray.	デジタル署名プロパティマップです。
proxyUser	代理人ユーザの DN です。

例

```
    //
    // Get the digital signature string for admin
    String digitalSignature =
DigitalSignatureUtils.getDigitalSignatureFromFile(IDigitalSignatureCon
stants.ADMIN_DIGITAL_SIGNATURE_FILENAME);

    ProvisioningRequestArray requestArray =
stub.getAllProvisioningRequests(recipient);
    //
    // If there are some then,
    if(requestArray != null)
    {
        String Id = " ";
        String requestId = " ";
        String requestNameToStart = "Enable Active Directory Account
(Mgr Approve-No Timeout)";
        //
        // Loop thru and find the request that we want to start
        ProvisioningRequest [] requests =
requestArray.getProvisioningrequest();
        for(int index = 0; index < requests.length; index++)
        {
            //
            // Is this the name of the request to start?
           if(requests[index].getName().compareTo(requestNameToStart)
== 0)
```

```
            {
                //
                // Get the current associated data items. Replicate a
new
                // dataitem array excluding the null values.
                Id = requests[index].getId();
                DataItem [] dataItem =
requests[index].getItems().getDataitem();
                if(dataItem != null)
                {
                    // Call method replicateDataItemArray on the
                    // provUtils utility object, which refers to a
                    // utility class that does not ship with the
                    // Identity Manager User Application.
                    DataItemArray newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
                    //
                    // Start a digitally signed provisioning resource
as  proxy for the recipient
                     requestId =
stub.startAsProxyWithDigitalSignature(request.getId(), recipient,
newDataItemArray, digitalSignature, null, proxyUser);
                }
            }
        }
    }
```

startWithCorrelationId

correlation ID を使ってワークフローを開始する場合に使用します。correlation ID は、関連するワークフロープロセスセットを追跡する方法を提供しています。このメソッドで開始したワークフロープロセスは、クエリしたり、correlation ID でソートすることができます。

メソッドの署名

```
java.lang.String startWithCorrelationId(java.lang.String processId,
java.lang.String recipient, com.novell.soa.af.impl.soap.DataItemArray
items, java.lang.String signature,
com.novell.soa.af.impl.soap.SignaturePropertyArray props,
java.lang.String proxyUser, java.lang.String correlationId)    throws
com.novell.soa.af.impl.soap.AdminException, java.rmi.RemoteException;
```

パラメータ

パラメータ	説明
processId	要求の ID です。
recipient	要求の受信者です。
項目	プロビジョニング要求のデータ項目です。
デジタル署名	デジタル署名です。

パラメータ	説明
digitalSignaturePropertyArray	デジタル署名プロパティマップです。
proxyUser	代理人ユーザの DN です。
correlationID	correlation ID を識別する文字列です。correlation ID は 32 文字以下でなければなりません。

20.3.3 ワークエントリ

この節は、各ワークエントリメソッドに関する参照情報を取り上げています。ワークエントリメソッドには、次のものが含まれています。

- 413 ページの 「転送」
- 415 ページの 「reassignWorkTask」
- 416 ページの 「getWork」
- 417 ページの 「forwardWithDigitalSignature」
- 419 ページの 「forwardAsProxy」
- 421 ページの 「unclaim」
- 422 ページの 「forwardAsProxyWithDigitalSignature」
- 425 ページの 「reassign」
- 426 ページの 「getWorkEntries」
- 428 ページの 「getQuorumForWorkTask」
- 429 ページの 「resetPriorityForWorkTask」

転送

タスクを、ワークフロー内の適切なアクション (承認、却下、拒否) がある次のアクティビティに転送する場合に使用します。

メソッドの署名

```
void forward(java.lang.String wid,
com.novell.soa.af.impl.soap.T_Action action,
com.novell.soa.af.impl.soap.DataItemArray items, java.lang.String
comment)
```

パラメータ

パラメータ	説明
wid	ワーク ID です。
action	行うアクションです (承認、却下、拒否)。
項目	ワークフローが必要とするデータ項目です。
comment	コメントです。

例

```
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);
     //
    // Get the process id for this running process
    Process process = stub.getProcess(requestId);
    String processId = null;
    if(process != null)
        processId = process.getProcessId();
    T_Action action = T_Action.APPROVE;

    T_Logic logic = T_Logic.AND;

    T_WorkEntryOrder workEntryOrder = T_WorkEntryOrder.REQUEST_ID;
    T_WorkEntryQueryChoice [] workEntryqueryChoice =  new
T_WorkEntryQueryChoice[3];
    workEntryqueryChoice[0] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[0].setRecipient(recipient);
    workEntryqueryChoice[1] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[1].setRequestId(requestId);
    workEntryqueryChoice[2] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[2].setProcessId(processId);
    //
    // Create work entry query
    T_WorkEntryQuery query = new T_WorkEntryQuery(logic,
_workEntryOrder, workEntryqueryChoice);
    //
    // Get all work entries (max records)
    WorkEntryArray workEntryArray = stub.getWorkEntries(query, -1);

    WorkEntry [] workEntry = workEntryArray.getWorkentry();

    if(workEntry != null

    {
         for(int wIndex = 0; wIndex < workEntry.length; wIndex++)
        {
            String workId = workEntry[wIndex].getId();
            //
            //
            LoggerUtils.sendToLogAndConsole("Forwarding : " +
```

```
workEntry[wIndex].getActivityName() + " work id: " + workId);
            //
           // Get the dataitem for this item of work
            DataItemArray dataItemArray = stub.getWork(workId);
            DataItem [] dataItem = dataItemArray.getDataitem();
            DataItemArray newDataItemArray = null;
            if(dataItem != null)
                // Call method replicateDataItemArray on the
                // provUtils utility object, which refers to a
                // utility class that does not ship with the
                // Identity Manager User Application.
                newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
            else
                throw new TestProgramException("DataItem is null.");
            //
            // Claim request for recipient
            String comment = _action.toString() + " this request: " +
requestId + " for " + recipient;
            stub.forward(workId, _action, newDataItemArray, comment);
        }

    }
```

reassignWorkTask

あるユーザから別のユーザにタスクを再割り当てする場合に使用します。

メソッドの署名

```
void reassignWorkTask(java.lang.String wid, java.lang.String
addressee, java.lang.String comment)
```

パラメータ

パラメータ	説明
wid	タスクの ID です。
addressee	タスクの宛先です。
comment	タスクに関するコメントです。

例

```
       //
       // Initialize and start a provisioning request
       HashMap provMap = new HashMap();
       provMap.put(Helper.RECIPIENT, recipient);
       provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
       //
       // Start request
       // Calls method startProvisioningRequest on the provUtils
       // utility object which refers to a utility class that does not
```

```
       // ship with the Identity Manager User Application.
       String requestId = provUtils.startProvisioningRequest(provMap,
null);
       sleep(5);
      //
      // Get the process id for this running process
      Process process = stub.getProcess(requestId);
      if(process != null)
      {
            String processId = process.getProcessId();
            String initiator = process.getInitiator();
            //
            // Setup for the query
            HashMap map = new HashMap();
            map.put(Helper.REQUESTID, requestId);
            map.put(Helper.RECIPIENT, recipient);
            map.put(Helper.PROCESSID, processId);
            map.put(Helper.INITIATOR, initiator);
            WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
T_WorkEntryOrder.REQUEST_ID, T_Logic.AND);
            if(workEntry == null)                throw new
TestProgramException("Work list is empty.");
            //
            // Reassign the work entry from recipient to the addressee
            //
            // Should only be one item
            String reassignComment = null;
            String workId = workEntry[0].getId();
            if(workId != null)
            {
                //
                // Reassign work entry(s) to addressee
               reassignComment = "Reassigning work entry " + workId +
" from " + recipient + " to " + addressee;
                stub.reassign(workId, addressee, reassignComment);
               LoggerUtils.sendToLogAndConsole("Reassign work entry "
+ workId + " from " + recipient + " to " + addressee);
            }
    }
```

getWork

タスクの ID(UUID) により識別されるワークエントリのデータ項目を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.DataItemArray getWork(java.lang.String
workId)
```

例

```
       //
       // Initialize and start a provisioning request
```

```
       HashMap provMap = new HashMap();
       provMap.put(Helper.RECIPIENT, recipient);
       provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
       //
       // Start request
       // Calls method startProvisioningRequest on the provUtils
       // utility object which refers to a utility class that does not
       // ship with the Identity Manager User Application.
       String requestId = provUtils.startProvisioningRequest(provMap,
null);
       sleep(5);
       //
       // Get the process id for this running process
      Process process = stub.getProcess(requestId);
      if(process != null)
      {
            String processId = process.getProcessId();
            String initiator = process.getInitiator();
            //
            // Setup for the query
            HashMap map = new HashMap();
            map.put(Helper.REQUESTID, requestId);
            map.put(Helper.RECIPIENT, recipient);
            map.put(Helper.PROCESSID, processId);
            map.put(Helper.INITIATOR, initiator);
            WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
T_WorkEntryOrder.REQUEST_ID, T_Logic.AND);
            //
            // Do assertion here
           Assert.assertNotNull("WorkEntry is null for recipient : " +
recipient + " with request id : " + requestId, workEntry);
            DataItemArray dataItemArray =
stub.getWork(workEntry[0].getId() );
            DataItem [] dataItem = dataItemArray.getDataitem();
            if(dataItem != null)
               LoggerUtils.sendToLogAndConsole(dataItem[0].getName());
      }
```

forwardWithDigitalSignature

デジタル署名とオプションのデジタル署名プロパティ付きのプロビジョニング要求を転送する場合に使用します。たとえば、管理者がユーザ向きアクティビティの承認、却下、拒否を強制させる場合などに使用します。

メソッドの署名

```
void forwardWithDigitalSignature(java.lang.String wid,
com.novell.soa.af.impl.soap.T_Action action,
com.novell.soa.af.impl.soap.DataItemArray items, java.lang.String
comment, java.lang.String digitalSignature,
com.novell.soa.af.impl.soap.SignaturePropertyArray
digitalSignaturePropertyArray)
```

パラメータ

パラメータ	説明
wid	ワーク ID です。
action	行うアクションです (承認、却下、拒否)。
項目	ワークフローが必要とするデータ項目です。
comment	アクションに関するコメントです。
digitalSignature	デジタル署名です。
digitalSignaturePropertyArray	デジタル署名プロパティマップです。

例

```
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);
     //
    // Get the process id for this running process
    Process process = stub.getProcess(requestId);
    String processId = null;
    if(process != null)
        processId = process.getProcessId();
    T_Action action = T_Action.APPROVE;
    T_Logic logic = T_Logic.AND;

    T_WorkEntryOrder workEntryOrder = T_WorkEntryOrder.REQUEST_ID;

 // Get the digital signature string for admin
 String digitalSignature =
DigitalSignatureUtils.getDigitalSignatureFromFile(IDigitalSignatureCon
stants.ADMIN_DIGITAL_SIGNATURE_FILENAME);

    T_WorkEntryQueryChoice [] workEntryqueryChoice =  new
T_WorkEntryQueryChoice[3];
    workEntryqueryChoice[0] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[0].setRecipient(recipient);
    workEntryqueryChoice[1] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[1].setRequestId(requestId);
    workEntryqueryChoice[2] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[2].setProcessId(processId);
```

```
    //
    // Create work entry query
    T_WorkEntryQuery query = new T_WorkEntryQuery(logic,
_workEntryOrder, workEntryqueryChoice);
    //
    // Get all work entries (max records)
    WorkEntryArray workEntryArray = stub.getWorkEntries(query, -1);

    WorkEntry [] workEntry = workEntryArray.getWorkentry();

    if(workEntry != null

    {
         for(int wIndex = 0; wIndex < workEntry.length; wIndex++)
        {
            String workId = workEntry[wIndex].getId();
            //
            //
            LoggerUtils.sendToLogAndConsole("Forwarding : " +
workEntry[wIndex].getActivityName() + " work id: " + workId);
            //
            // Get the dataitem for this item of work
            DataItemArray dataItemArray = stub.getWork(workId);
            DataItem [] dataItem = dataItemArray.getDataitem();
            DataItemArray newDataItemArray = null;
            if(dataItem != null)
                  // Call method replicateDataItemArray on the
                  // provUtils utility object, which refers to a
                  // utility class that does not ship with the
                  // Identity Manager User Application.
                newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
            else
                throw new TestProgramException("DataItem is null.");
            //
            // Claim request for recipient
            String comment = _action.toString() + " this request: " +
requestId + " for " + recipient;
            stub.forwardWithDigitalSignature(workId, _action,
newDataItemArray, comment, digitalSignature, null);
      }

    }
```

forwardAsProxy

プロビジョニング要求を転送する場合に使用します。たとえば、管理者がユーザ向きアクティビティの承認、却下、拒否を強制させる場合などに使用します。

メソッドの署名

```
void forwardAsProxy(java.lang.String wid,
com.novell.soa.af.impl.soap.T_Action action,
```

```
com.novell.soa.af.impl.soap.DataItemArray items, java.lang.String
comment, java.lang.String proxyUser)
```

パラメータ

パラメータ	説明
wid	ワーク ID(アクティビティ ID) です。
action	行うアクションです (承認、却下、拒否)。
項目	ワークフローが必要とするデータ項目です。
comment	アクティビティに追加するコメントです。
proxyUser	代理人ユーザの DN です。

例

```
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);
     //
    // Get the process id for this running process
    Process process = stub.getProcess(requestId);
    String processId = null;
    if(process != null)        processId = process.getProcessId();
    T_Action action = T_Action.APPROVE;

    T_Logic logic = T_Logic.AND;

    T_WorkEntryOrder workEntryOrder = T_WorkEntryOrder.REQUEST_ID;


    T_WorkEntryQueryChoice [] workEntryqueryChoice =  new
T_WorkEntryQueryChoice[3];
    workEntryqueryChoice[0] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[0].setRecipient(recipient);
    workEntryqueryChoice[1] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[1].setRequestId(requestId);
    workEntryqueryChoice[2] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[2].setProcessId(processId);
    //
    // Create work entry query
```

```
    T_WorkEntryQuery query = new T_WorkEntryQuery(logic,
_workEntryOrder, workEntryqueryChoice);
    //
    // Get all work entries (max records)
    WorkEntryArray workEntryArray = stub.getWorkEntries(query, -1);

    WorkEntry [] workEntry = workEntryArray.getWorkentry();

    if(workEntry != null

    {
         for(int wIndex = 0; wIndex < workEntry.length; wIndex++)
        {
            String workId = workEntry[wIndex].getId();
            //
            //
            LoggerUtils.sendToLogAndConsole("Forwarding : " +
workEntry[wIndex].getActivityName() + " work id: " + workId);
            //
            // Get the dataitem for this item of work
            DataItemArray dataItemArray = stub.getWork(workId);
            DataItem [] dataItem = dataItemArray.getDataitem();
            DataItemArray newDataItemArray = null;
            if(dataItem != null)
                // Call method replicateDataItemArray on the
                // provUtils utility object, which refers to a
                // utility class that does not ship with the
                // Identity Manager User Application.
                newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
            else
                throw new TestProgramException("DataItem is null.");
            //
            // Claim request for recipient
            String comment = _action.toString() + " this request: " +
requestId + " for " + recipient;
            String proxyUser =
ServiceUtils.getInstance().getLoginData().getUsername(LoginData.PROXY_
TYPE);
            stub.forwardAsProxy(workId, _action, newDataItemArray,
comment, proxyUser);        }

    }
```

unclaim

プロビジョニング要求のクレーム (引き受け) を解除する場合に使用します。このメソッドは、ユーザアプリケーションで要求が引き受けられた場合にのみ機能します。forward API メソッド (413 ページの 「転送」 を参照) では、1 つの操作で引き受けと転送を行うため、SOAP インタフェースを使って引き受けが転送された場合、要求の引き受けを解除することはできません。

メソッドの署名

```
void unclaim(java.lang.String wid, java.lang.String comment)
```

パラメータ

パラメータ	説明
workId	引き受けを解除するアクティビティの ID です。
comment	アクションに関するコメントです。

例

```
    // Action and Approval Types
    final int SELECTED_ACTION = 0; final int CLAIMED_SELECTED_ACTION =
0;
    T_Action [] action = {T_Action.APPROVE, T_Action.REFUSE,
T_Action.DENY};
    T_ApprovalStatus [] claimedAction = {T_ApprovalStatus.Approved,
T_ApprovalStatus.Retracted, T_ApprovalStatus.Denied};
    //
    // Get the process id for this running process
    Process process = stub.getProcess(requestId);
    String processId = null;
    if(process != null)
        processId = process.getProcessId();

    HashMap map = new HashMap();
    map.put(Helper.REQUESTID, requestId);
    map.put(Helper.RECIPIENT, recipient);
    map.put(Helper.PROCESSID, processId);
    //
    // Claim the request
    WorkEntry workEntry = workEntryUtils.claimWorkEntry(map,
action[SELECTED_ACTION]);
    if(workEntry != null)
    {
        //
        // Now unclaim the entry
        String workId = workEntry.getId();
       stub.unclaim(workId, "Unclaiming this work item : " + workId +
" for request id : " + requestId);
   }
```

forwardAsProxyWithDigitalSignature

デジタル署名とデジタル署名プロパティ付きのプロビジョニング要求を転送する場合に使用します。たとえば、管理者がユーザ向きアクティビティの承認、却下、拒否を強制させる場合などに使用します。

メソッドの署名

```
void forwardAsProxyWithDigitalSignature(java.lang.String wid,
com.novell.soa.af.impl.soap.T_Action action,
com.novell.soa.af.impl.soap.DataItemArray items, java.lang.String
```

```
comment, java.lang.String digitalSignature,
com.novell.soa.af.impl.soap.SignaturePropertyArray
digitalSignaturePropertyArray, java.lang.String proxyUser)
```

パラメータ

パラメータ	説明
wid	ワーク ID(アクティビティ ID) です。
action	行うアクションです (承認、却下、拒否)。
項目	ワークフローが必要とするデータ項目です。
comment	アクティビティに追加するコメントです。
digitalSignature	デジタル署名です。
digitalSignaturePropertyArray	デジタル署名プロパティマップです。
proxyUser	代理人ユーザの DN です。

例

```
     //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);
     //
    // Get the process id for this running process
    Process process = stub.getProcess(requestId);
    String processId = null;
    if(process != null)
        processId = process.getProcessId();

    T_Action action = T_Action.APPROVE;

    T_Logic logic = T_Logic.AND;

    T_WorkEntryOrder workEntryOrder = T_WorkEntryOrder.REQUEST_ID;
    T_WorkEntryQueryChoice [] workEntryqueryChoice =  new
T_WorkEntryQueryChoice[3];
    workEntryqueryChoice[0] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[0].setRecipient(recipient);
    workEntryqueryChoice[1] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[1].setRequestId(requestId);
```

```
    workEntryqueryChoice[2] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[2].setProcessId(processId);
    //
    // Create work entry query
    T_WorkEntryQuery query = new T_WorkEntryQuery(logic,
_workEntryOrder, workEntryqueryChoice);
    //
    // Get all work entries (max records)
    WorkEntryArray workEntryArray = stub.getWorkEntries(query, -1);

    WorkEntry [] workEntry = workEntryArray.getWorkentry();

    if(workEntry != null

    {
         for(int wIndex = 0; wIndex < workEntry.length; wIndex++)
        {
            String workId = workEntry[wIndex].getId();
            //
            //
            LoggerUtils.sendToLogAndConsole("Forwarding : " +
workEntry[wIndex].getActivityName() + " work id: " + workId);
            //
            // Get the dataitem for this item of work
            DataItemArray dataItemArray = stub.getWork(workId);
            DataItem [] dataItem = dataItemArray.getDataitem();
            DataItemArray newDataItemArray = null;
            if(dataItem != null)
                // Call method replicateDataItemArray on the
                // provUtils utility object, which refers to a
                // utility class that does not ship with the
                // Identity Manager User Application.
                newDataItemArray =
provUtils.replicateDataItemArray(dataItem);
            else
                throw new TestProgramException("DataItem is null.");
            //
            // Claim request for recipient
            String comment = _action.toString() + " this request: " +
requestId + " for " + recipient;
            String digitalSignature =
DigitalSignatureUtils.getDigitalSignatureFromFile(IDigitalSignatureCon
stants.MMACKENZIE_DIGITAL_SIGNATURE_FILENAME);
            String proxyUser =
ServiceUtils.getInstance().getLoginData().getUsername(LoginData.PROXY_
TYPE);

            stub.forwardAsProxyWithDigitalSignature(workId, _action,
newDataItemArray, comment, digitalSignature, null, proxyUser);
        }

    }
```

reassign

あるユーザから別のユーザにタスクを再割り当てする場合に使用します。

メソッドの署名

```
void reassign(java.lang.String wid, java.lang.String addressee,
java.lang.String comment)
```

パラメータ

パラメータ	説明
wid	再割り当てするアクティビティの ID です。
addressee	アクティビティの宛先です。
comment	アクションに関するコメントです。

例

```
        //
        // Initialize and start a provisioning request
        HashMap provMap = new HashMap();
        provMap.put(Helper.RECIPIENT, recipient);
        provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
        //
        // Start request
        // Calls method startProvisioningRequest on the provUtils
        // utility object which refers to a utility class that does not
        // ship with the Identity Manager User Application.
        String requestId = provUtils.startProvisioningRequest(provMap,
null);
        sleep(5);
        //
       // Get the process id for this running process
       Process process = stub.getProcess(requestId);
       if(process != null)
       {
             String processId = process.getProcessId();
             String initiator = process.getInitiator();
             //
             // Setup for the query
             HashMap map = new HashMap();
             map.put(Helper.REQUESTID, requestId);
             map.put(Helper.RECIPIENT, recipient);
             map.put(Helper.PROCESSID, processId);
             map.put(Helper.INITIATOR, initiator);
             WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
T_WorkEntryOrder.REQUEST_ID, T_Logic.AND);

             if(workEntry == null)
                 throw new TestProgramException("Work list is empty.");
             //
```

```
            // Reassign the work entry from recipient to the addressee
            //
            // Should only be one work item
            String reassignComment = null;
            String workId = workEntry[0].getId();
            if(workId != null)
            {
                //
                // Reassign work entry(s) to addressee
               reassignComment = "Reassigning work entry " + workId +
" from " + recipient + " to " + addressee;
                stub.reassign(workId, addressee, reassignComment);
               LoggerUtils.sendToLogAndConsole("Reassign work entry "
+ workId + " from " + recipient + " to " + addressee);
            }
    }
```

getWorkEntries

ワークエントリ (アクティビティ) にクエリして、クエリの条件を満たす WorkEntry オブジェクトのリストを返す場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.WorkEntryArray
getWorkEntries(com.novell.soa.af.impl.soap.T_WorkEntryQuery query, int
maxRecords)
```

パラメータ

パラメータ	説明
クエリー	アクティビティのリストを取得するクエリを指定する場合に使用します。クエリには、次のコンポーネントがあります。 ◆ choice - 結果をフィルタリングする場合に使用するパラメータです。複数のパラメータを指定することができます。次のパラメータを使用できます。 Addresse - DN、 ProcessId、 RequestId、 ActivityId、 Status(整数)、 Owner、 Priority、 CreationTime(YYYY/MM/DD)、 ExpTime(YYYY/MM/DD)、 CompletionTime(YYYY/MM/DD)、 Recipient、 Initiator、 ProxyFor ◆ logic - AND または OR です。 ◆ order - 結果のソート順を指定します。orderに指定できる値を次に示します。 ACTIVITY_ID、 RECIPIENT、 PROVISIONING_TIME、 RESULT_TIME、 STATE、 STATUS、 REQUEST_ID、 MESSAGE
maxRecords	取得する最大レコード数を指定する場合に使用します。-1 を指定すると、返されるレコード数に制限はありません。

例

```
    T_Action action = T_Action.APPROVE;

    T_Logic logic = T_Logic.AND;

    T_WorkEntryOrder workEntryOrder = T_WorkEntryOrder.REQUEST_ID;
    T_WorkEntryQueryChoice [] workEntryqueryChoice =  new
T_WorkEntryQueryChoice[3];
    workEntryqueryChoice[0] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[0].setRecipient(recipient);
```

```
    workEntryqueryChoice[1] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[1].setRequestId(requestId);
    workEntryqueryChoice[2] = new T_WorkEntryQueryChoice();
    workEntryqueryChoice[2].setProcessId(processId);
    //
    // Create work entry query
    T_WorkEntryQuery query = new T_WorkEntryQuery(logic,
_workEntryOrder, workEntryqueryChoice);
    //
    // Get all work entries (max records)
    WorkEntryArray workEntryArray = stub.getWorkEntries(query, -1);

    WorkEntry [] workEntry = workEntryArray.getWorkentry();
```

getQuorumForWorkTask

ワークフローアクティビティの定数に関する情報を取得する場合に使用します。このメソッドが機能するためには、ワークフロー設計者によりワークフローアクティビティの定数が実際に指定されていなければなりません。

メソッドの署名

```
com.novell.soa.af.impl.soap.Quorum
getQuorumForWorkTask((java.lang.String workId)
```

例

```
     //

     // Note: Provisioning resource must contain a quorum in the flow
for this api method to work

    //
    // Action and Approval Types
    final int SELECTED_ACTION = 0; final int CLAIMED_SELECTED_ACTION =
0;
    T_Action [] action = {T_Action.APPROVE, T_Action.REFUSE,
T_Action.DENY};
    T_ApprovalStatus [] claimedAction = {T_ApprovalStatus.Approved,
T_ApprovalStatus.Retracted, T_ApprovalStatus.Denied};
    //
    // Get the process id for this running process
    Process process = stub.getProcess(requestId);
    String processId = null;
    if(process != null)
        processId = process.getProcessId();
    //
    // Setup for the query
    HashMap map = new HashMap();
    map.put(Helper.REQUESTID, requestId);
    map.put(Helper.RECIPIENT, recipient);
    map.put(Helper.PROCESSID, processId);
    map.put(Helper.INITIATOR, process.getInitiator() );
    WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
```

```
T_WorkEntryOrder.REQUEST_ID, T_Logic.AND);
    Assert.assertNotNull("WorkEntry is null for recipient : " +
recipient + " with request id : " + requestId, workEntry);
    //
    //
    String workId = workEntry[0].getId();

    Quorum quorum = stub.getQuorumForWorkTask(workId);

    Assert.assertNotNull("Quorum for work task is null for recipient :
" + recipient + " with request id : " + requestId, quorum);
    //

    // Extract some data
    int approvalCondition = quorum.getApprovalCondition();
    int status = quorum.getStatus();
    int approveCount = quorum.getApproveCount();
    int participantCount = quorum.getParticipantCount();
    int refuseCount = quorum.getRefuseCount();
```

resetPriorityForWorkTask

タスクの優先度をリセットする場合に使用します。このメソッドは、承認の分岐が 1 つのプロビジョニング要求にのみ使用してください。

メソッドの署名

```
void resetPriorityForWorkTask(java.lang.String workId, int priority,
java.lang.String comment)
```

パラメータ

パラメータ	説明
workId	アクティビティの ID です。
優先度	アクティビティに設定する優先度です。
comment	アクションに関するコメントです。

例

```
// Calls method getProvisioningResourceNameForRecipient
// on the provUtils utility object, which refers to a utility class
// that does not ship with the Identity Manager User Application.
String requestNameToStart =
provUtils.getProvisioningResourceNameForRecipient(recipient, "Enable
Active Directory Account");
    Map map = MapUtils.createAndSetMap(new Object[] {
            Helper.RECIPIENT, recipient,
            IProvisioningConstants.PROVISIONING_REQUEST_TO_START,
requestNameToStart});
    //
    // Try and start the provisioning request
     String requestId =
```

```
provWrapper.startProvisioningRequest(recipient, requestNameToStart);
    RationalTestScript.sleep(5);
    //
    // Get the process id for this running process
    Process process = stub.getProcess(requestId);
    if(process != null)
    {
        //
        // Setup for the query
        HashMap map = new HashMap();
        map.put(Helper.REQUESTID, requestId);
        map.put(Helper.RECIPIENT, recipient);
        map.put(Helper.PROCESSID, process.getProcessId());
        map.put(Helper.INITIATOR, process.getInitiator());
        WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
T_WorkEntryOrder.REQUEST_ID, T_Logic.AND);
        //
        // Now reset the priority for this work item.
        String workId = workEntry[0].getId();
        String comment = "Resetting priority for this work item.";
        int priority = 0;
        stub.resetPriorityForWorkTask(workId, priority, comment);
}
```

20.3.4 コメント

この節は、各コメントメソッドに関する参照情報を取り上げています。コメントメソッドには次のものが含まれています。

- 430 ページの 「getCommentsByType」
- 432 ページの 「getCommentsByActivity」
- 433 ページの 「getCommentsByUser」
- 433 ページの 「getCommentsByCreationTime」
- 434 ページの 「addComment」
- 435 ページの 「getComments」

getCommentsByType

特定のタイプ (ユーザ、システムなど) のワークフローコメントを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.CommentArray
getCommentsByType(java.lang.String requestId,
com.novell.soa.af.impl.soap.T_CommentType type)
```

パラメータ

パラメータ	説明
requestId	プロセス ID です。
type	コメントタイプ (USER または SYSTEM) です。

例

```
       //
       // Initialize and start a provisioning request
       HashMap provMap = new HashMap();
       provMap.put(Helper.RECIPIENT, recipient);
       provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
       //
       // Start request
       // Calls method startProvisioningRequest on the provUtils
       // utility object which refers to a utility class that does not
       // ship with the Identity Manager User Application.
       String requestId = provUtils.startProvisioningRequest(provMap,
null);
      sleep(5);
      //
      // Get the comments by type : either User or System
      T_CommentType [] commentTypes = {T_CommentType.User,
T_CommentType.System};

     for(int types = 0; types < commentTypes.length; types++)
    {
        CommentArray commentArray = stub.getCommentsByType(requestId,
commentTypes[types]);
        Comment [] comments = commentArray.getComment();
        if(comments != null)
        {
            for(int index = 0; index < comments.length; index++)
            {
                LoggerUtils.sendToLogAndConsole(" \nComment Type = " +
commentTypes[types].getValue() + "\n" +
                            "Activity Id: " +
comments[index].getActivityId() + "\n" +
                           "Comment : " + comments[index].getComment()
+ "\n" +
                          "User : " + comments[index].getUser() + "\n"
+
                            "System comment : " +
comments[index].getSystemComment() + "\n" +
                            "Time stamp : " +
comments[index].getTimestamp().getTime().toString() );
            }
        }
    }
```

getCommentsByActivity

特定のアクティビティのコメントを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.CommentArray
getCommentsByActivity(java.lang.String requestId, java.lang.String
aid)
```

パラメータ

パラメータ	説明
requestId	プロセス ID です。
aid	アクティビティ ID です。

例

```
       //
       // Initialize and start a provisioning request
       HashMap provMap = new HashMap();
       provMap.put(Helper.RECIPIENT, recipient);
       provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
       //
       // Start request
       // Calls method startProvisioningRequest on the provUtils
       // utility object which refers to a utility class that does not
       // ship with the Identity Manager User Application.
       String requestId = provUtils.startProvisioningRequest(provMap,
null);
      sleep(5);
      //
      // Get the process id for this running process
      Process process = stub.getProcess(requestId);
      if(process != null)
      {
            String processId = process.getProcessId();
            String initiator = process.getInitiator();
            //
            // Setup for the query
            HashMap map = new HashMap();
            map.put(Helper.REQUESTID, requestId);
            map.put(Helper.RECIPIENT, recipient);
            map.put(Helper.PROCESSID, processId);
            map.put(Helper.INITIATOR, initiator);
            WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
T_WorkEntryOrder.REQUEST_ID, T_Logic.AND);
            //
            // Get the activity id associated with the item of work
            String activityId = workEntry[0].getActivityId();
            //
```

```
            // Get the comments based on activity
            if(activityId != null)
            {
                CommentArray commentArray =
stub.getCommentsByActivity(requestId, activityId);
                Comment [] comments = commentArray.getComment();
            }

    }
```

getCommentsByUser

特定のユーザが行ったコメントを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.CommentArray
getCommentsByUser(java.lang.String requestId, java.lang.String user)
```

パラメータ

パラメータ	説明
requestId	プロセス ID です。
user	コメントを作成したユーザ (受信者) の DN です。

例

```
       //
       // Initialize and start a provisioning request
       HashMap provMap = new HashMap();
       provMap.put(Helper.RECIPIENT, recipient);
       provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
       //
       // Start request
       // Calls method startProvisioningRequest on the provUtils
       // utility object which refers to a utility class that does not
       // ship with the Identity Manager User Application.
       String requestId = provUtils.startProvisioningRequest(provMap,\
null);
      sleep(5);
     //
      // Get the comments by recipient (should be the same as user)
      CommentArray commentArray = stub.getCommentsByUser(requestId,
recipient);
      Comment [] comments = commentArray.getComment();
```

getCommentsByCreationTime

特定の時刻に行われたコメントを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.CommentArray
getCommentsByCreationTime(java.lang.String requestId, long time,
com.novell.soa.af.impl.soap.T_Operator op)
```

パラメータ

パラメータ	説明
requestId	プロセス ID です。
タイム	タイムスタンプです。
op	使用するクエリ演算子です。次の演算子を使用できます。 EQ - 等しい LT - 未満 LE - 以下 GT - より大きい GE - 以上

例

```
        //
        // Initialize and start a provisioning request
        HashMap provMap = new HashMap();
        provMap.put(Helper.RECIPIENT, recipient);
        provMap.put(I"Provisioning_Request_To_Start_Key", "Enable
Active Directory Account (Mgr Approve-No Timeout)");
        //
        // Start request
        // Calls method startProvisioningRequest on the provUtils
        // utility object which refers to a utility class that does not
        // ship with the Identity Manager User Application.
        String requestId = provUtils.startProvisioningRequest(provMap,
null);
      sleep(5);
      //
      // Get comments by creation time for the provisioning request
started above.
      long currentTime = System.currentTimeMillis();
      LoggerUtils.sendToLogAndConsole("-->Current date = " + new
java.util.Date(currentTime).toString() );
      //
      //
      T_Operator operator = T_Operator.GT;
      CommentArray commentArray =
stub.getCommentsByCreationTime(requestId, currentTime, operator);
      Comment [] comments = commentArray.getComment();
```

addComment

ワークフローアクティビティにコメントを追加する場合に使用します。

メソッドの署名

```
void addComment(java.lang.String workId, java.lang.String comment)
```

パラメータ

パラメータ	説明
workId	アクティビティ ID(UUID) です。
comment	アクティビティに関するコメントです。

例

```
       //       // Initialize and start a provisioning request
HashMap provMap = new HashMap();       provMap.put(Helper.RECIPIENT,
recipient);       provMap.put(I"Provisioning_Request_To_Start_Key",
"Enable Active Directory Account (Mgr Approve-No Timeout)");     //
// Start request
      // Calls method startProvisioningRequest on the provUtils      /
/ utility object which refers to a utility class that does not       /
/ ship with the Identity Manager User Application.
       String requestId = provUtils.startProvisioningRequest(provMap,
null);      sleep(5);      //      // Setup for the query      HashMap
map = new HashMap();      map.put(Helper.REQUESTID, requestId);
map.put(Helper.RECIPIENT, recipient);      WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
T_WorkEntryOrder.REQUEST_ID, T_Logic.AND);      //      // Add comment
to the work entry      String workId = workEntry[0].getId();
String processId = workEntry[0].getProcessId();      String addComment
= "Test comment for work id " + workId;      stub.addComment(workId,
addComment);      sleep(2);
```

getComments

ワークフローからコメントを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.CommentArray getComments(java.lang.String
workId, int maxRecords)
```

パラメータ

パラメータ	説明
workId	アクティビティ ID(UUID) です。
maxRecords	取得する最大レコード数を指定する整数です。

例

```
      //
      // Setup for the query
      HashMap map = new HashMap();
      map.put(Helper.RECIPIENT, addressee);
```

```
      WorkEntry [] workEntry =
workEntryUtils.getWorkEntriesUsingQuery(map,
T_WorkEntryOrder.ADDRESSEE, T_Logic.OR);
      //
      // Get all the comment records for this workId
      int maxRecords = -1;
     CommentArray commentArray = stub.getComments(workId, maxRecords);
      Comment [] comment = commentArray.getComment();
```

20.3.5 設定

この節は、各環境設定メソッドに関する参照情報を取り上げています。環境設定メソッドには次のものが含まれています。

setCompletedProcessTimeout

完了したプロセスのタイムアウトを設定する場合に使用します。タイムアウトに指定した日数より前に完了したプロセスは、システムから削除されます。デフォルトは 120 日です。0 ～ 365 日の値を指定できます。

メソッドの署名

```
void setCompletedProcessTimeout(int time)
```

例

```
accessConfigurationSettings(SET_COMPLETED_PROCESS_TIMEOUT, new
Integer(212) );
```

setEngineConfiguration

ワークフローエンジン環境設定パラメータを設定する場合に使用します。

メソッドの署名

```
void setEngineConfiguration(com.novell.soa.af.impl.soap.Configuration
config)
```

パラメータ

パラメータ	説明
minPoolSize	最小スレッドプールサイズです。
maxnPoolSize	最大スレッドプールサイズです。
initialPoolSize	初期スレッドプールサイズです。
keepAliveTime	スレッドプールの有効時間です。
pendingInterval	クラスタ同期時刻です。
cleanupInterval	データベースからプロセスをパージする間隔です。
retryQueueInterval	失敗したプロセスを再試行する間隔です。
maxShutdownTime	エンジンをシャットダウンする前に、スレッドを完了させるために待機する最大時間です。
userActivityTimeout	デフォルトのユーザアクティビティタイムアウト時間です。
completedProcessTimeout	デフォルトの完了プロセスタイムアウトです。
webServiceActivityTimeout	デフォルトの Web サービスアクティビティタイムアウトです。
emailNotification	電子メール通知をオンまたはオフにします。
processCacheInitialCapacity	プロセスキャッシュ初期容量です。
processCacheMaxCapacity	プロセスキャッシュ最大容量です。
processCacheLoadFactor	プロセスキャッシュロードファクタです。
heartbeatInterval	ハートビート間隔です。
heartbeatFactor	ハートビートファクタです。

例

```
accessConfigurationSettings(SET_ENGINE_CONFIGURATION, new Integer(313)
);
```

getCompletedProcessTimeout

完了したプロセスのタイムアウトを取得する場合に使用します。

メソッドの署名

```
int getCompletedProcessTimeout()
```

例

```
accessConfigurationSettings(GET_COMPLETED_PROCESS_TIMEOUT, new
Integer(121) );
```

setEmailNotifications

電子メール通知をグローバルに有効／無効にする場合に使用します。

メソッドの署名

```
void setEmailNotifications(boolean enable)
```

パラメータ

パラメータ	説明
有効	true の場合、電子メール通知が有効になります。そうでない場合は、無効になります。

例

```
accessConfigurationSettings(SET_EMAIL_NOTIFICATIONS, new
Boolean(false) );
```

clearNIMCaches

Novell Integration Manager(以前の exteNd Composer) キャッシュを消去します。

メソッドの署名

```
void clearNIMCaches()
```

例

```
accessConfigurationSettings(CLEAR_NIM_CACHES, new Object() );
```

setWebServiceActivityTimeout

Web サービスアクティビティのタイムアウトを設定する場合に使用します。デフォルト値は 50 分です。1 分から 7 日までの範囲の値を指定できます。

メソッドの署名

```
void setWebServiceActivityTimeout(int time)
```

パラメータ

パラメータ	説明
タイム	タイムアウト値 (分) です。

例

```
accessConfigurationSettings(SET_WEBSERVICE_ACTIVITY_TIMEOUT, new
Integer(767) );
```

getUserActivityTimeout

ユーザ向きアクティビティのタイムアウトを取得する場合に使用します。

メソッドの署名

```
int getUserActivityTimeout()
```

例

```
accessConfigurationSettings(GET_USER_ACTIVITY_TIMEOUT, new
Integer(3767) );
```

getEmailNotifications

グローバル電子メール通知が有効か、または無効かを判断する場合に使用します。

メソッドの署名

```
boolean getEmailNotifications()
```

例

```
accessConfigurationSettings(GET_EMAIL_NOTIFICATIONS, new Boolean(true)
);
```

setUserActivityTimeout

ユーザ向きアクティビティのタイムアウトを設定する場合に使用します。デフォルトではタイムアウトはありません (ゼロ)。1 時間から 365 日までの範囲の値を指定できます。

メソッドの署名

```
void setUserActivityTimeout(int time)
```

パラメータ

パラメータ	説明
タイム	タイムアウト値 (時間) です。

例

```
accessConfigurationSettings(SET_USER_ACTIVITY_TIMEOUT, new
Integer(1767) );
```

getEngineConfiguration

ワークフローエンジン環境設定パラメータを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.Configuration getEngineConfiguration()
```

例

```
accessConfigurationSettings(GET_ENGINE_CONFIGURATION, new Integer(141)
);
```

getWebServiceActivityTimeout

Web サービスアクティビティのタイムアウトを取得する場合に使用します。

メソッドの署名

```
int getWebServiceActivityTimeout()
```

例

```
accessConfigurationSettings(GET_WEBSERVICE_ACTIVITY_TIMEOUT, new
Integer(808) );
```

20.3.6 その他

この節は、その他のメソッドに関する参照情報を取り上げています。その他のメソッドには次のものが含まれています。

- 440 ページの 「getGraph」
- 441 ページの 「getFlowDefinition」
- 442 ページの 「getFormDefinition」
- 443 ページの 「getVersion」

getGraph

ワークフローの JPG イメージを取得する場合に使用します。アプリケーションサーバと IDM ユーザアプリケーションが動作するコンピュータには、Graphviz プログラムをインストールしておく必要があります。Graphviz の詳細は、「Graphviz (http://www.graphviz.org)」を参照してください。

メソッドの署名

```
byte[] getGraph(java.lang.String processId)
```

パラメータ

パラメータ	説明
processId	要求 ID です。

例

```
 //
      // Initialize and start a provisioning request
      HashMap provMap = new HashMap();
      provMap.put(Helper.RECIPIENT, recipient);
      provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
      //
      // Start request
      // Calls method startProvisioningRequest on the provUtils
      // utility object which refers to a utility class that does not
      // ship with the Identity Manager User Application.
      String requestId = provUtils.startProvisioningRequest(provMap,
null);
      sleep(5);
      //

      //
      Process process = stub.getProcess(requestId);
      if(process != null)
     {
```

```
        byte [] graph = null;
        if( (recipient.compareTo(process.getRecipient()) == 0) &&
(requestId.compareTo(process.getRequestId()) == 0) )
        {
            graph = stub.getGraph(process.getProcessId() );
        }
        //
        // Do assert
        Assert.assertNotNull("Graph is null.", graph);
    }
```

getFlowDefinition

プロビジョニング要求の XML を取得する場合に使用します。

メソッドの署名

```
java.lang.String getFlowDefinition(java.lang.String processId)
```

パラメータ

パラメータ	説明
processId	要求 ID です。

例

```
 //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);
     //

     //

     Process process = stub.getProcess(requestId);
     if(process != null)
    {
        String XMLFlowDefinition = null;
        if( (recipient.compareTo(process.getRecipient()) == 0) &&
(requestId.compareTo(process.getRequestId()) == 0) )
        {
            XMLFlowDefinition =
stub.getFlowDefinition(process.getProcessId() );
        }
```

```
        //
        // Do assert
        Assert.assertNotNull("Flow Definition is null.",
XMLFlowDefinition);
    }
```

getFormDefinition

プロビジョニング要求のフォームの XML を取得する場合に使用します。

メソッドの署名

```
java.lang.String getFormDefinition(java.lang.String processId)
```

パラメータ

パラメータ	説明
processId	要求 ID です。

例

```
 //
     // Initialize and start a provisioning request
     HashMap provMap = new HashMap();
     provMap.put(Helper.RECIPIENT, recipient);
     provMap.put(I"Provisioning_Request_To_Start_Key", "Enable Active
Directory Account (Mgr Approve-No Timeout)");
     //
     // Start request
     // Calls method startProvisioningRequest on the provUtils
     // utility object which refers to a utility class that does not
     // ship with the Identity Manager User Application.
     String requestId = provUtils.startProvisioningRequest(provMap,
null);
     sleep(5);
     //

     //

     Process process = stub.getProcess(requestId);
     if(process != null)
    {

         String XMLFormDefinition = null;
         if( (recipient.compareTo(process.getRecipient()) == 0) &&
(requestId.compareTo(process.getRequestId()) == 0) )
        {
            XMLFormDefinition =
stub.getFormDefinition(process.getProcessId() );
        }
        //
        // Do assert
        Assert.assertNotNull("Form Definition is null.",
```

```
XMLFormDefinition);
    }
```

getVersion

ワークフローシステムのバージョンを取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.T_Version getVersion()
```

例

```
 StringBuffer result = new StringBuffer();

     T_Version version = stub.getVersion();
     if (version != null)
    {
        result.append(" Major = " + version.getMajor() );
        result.append(" Minor = " + version.getMinor() );
        result.append(" Revision = " + version.getRevision() );

       System.out.println("Version Information " + result.toString());
    }
```

20.3.7 クラスタ

この節は、各クラスタメソッドに関する参照情報を取り上げています。クラスタメソッドには次のものが含まれています。

- 443 ページの 「getEngineState」
- 444 ページの 「reassignAllProcesses」
- 444 ページの 「getEngineState」
- 445 ページの 「reassignPercentageProcesses」
- 446 ページの 「reassignProcesses」
- 446 ページの 「removeEngine」

getEngineState

エンジン ID で指定された、ワークフローエンジンの IEngineState を取得する場合に使用します。

メソッドの署名

```
com.novell.soa.af.impl.soap.EngineState
getEngineState(java.lang.String engineId)
```

パラメータ

パラメータ	説明
engineId	ワークフローエンジンの ID です。

例

```
 EngineStateArray engineStateArray = stub.getClusterState();
    EngineState [] engineState = engineStateArray.getEngineStates();
    if(engineState != null)
    {
        LoggerUtils.sendToLogAndConsole("EngineCount in cluster:" +
engineState.length);
        for(int index = 0; index < engineState.length; index++)
        {
            EngineState engine =
stub.getEngineState(engineState[index].getEngineId() );
            LoggerUtils.sendToLogAndConsole(
                "Engine Id: " + engine.getEngineId() + "\n" +
                "Engine status: " + engine.getEngineStatus() + "\n" +
                "Value of engine status: " +
engine.getValueOfEngineStatus() + "\n" +
                "Heartbeat: " + ( (engine.getHeartbeat() != null) ?
engine.getHeartbeat().getTime().toString() : "null") + "\n" +
                "Shutdown time: " + ((engine.getShutdownTime()!= null)
? engine.getShutdownTime().getTime().toString() : "null") + "\n" +
                "Start time: " + ((engine.getStartTime() != null) ?
engine.getStartTime().getTime().toString() : "null") );
        }
    }
```

reassignAllProcesses

ソースエンジンのすべてのプロセスを、ターゲットエンジンのリストに再割り当てする場合に使用します。

メソッドの署名

```
int reassignAllProcesses(java.lang.String sourceEngineId,
com.novell.soa.af.impl.soap.StringArray targetEngineIds)
```

パラメータ

パラメータ	説明
sourceEngineId	ソースワークフローエンジンの ID です。
targetEngineIds	ターゲットワークフローエンジンの ID です。

getEngineState

クラスタ内の各エンジンに対して、IEngineState オブジェクトを含むリストを取得する場合に使用します。

メソッドの署名

```
public com.novell.soa.af.impl.soap.EngineState
getEngineState(java.lang.String engineId)
```

パラメータ

パラメータ	説明
engineId	ワークフローエンジンの ID です。

例

```
 EngineStateArray engineStateArray = stub.getClusterState();
    EngineState [] engineState = engineStateArray.getEngineStates();
    if(engineState != null)
    {
        LoggerUtils.sendToLogAndConsole("EngineCount in cluster:" +
engineState.length);
        for(int index = 0; index < engineState.length; index++)
        {
            EngineState engine =
stub.getEngineState(engineState[index].getEngineId() );
            LoggerUtils.sendToLogAndConsole(
                "Engine Id: " + engine.getEngineId() + "\n" +
                "Engine status: " + engine.getEngineStatus() + "\n" +
                "Value of engine status: " +
engine.getValueOfEngineStatus() + "\n" +
                "Heartbeat: " + ( (engine.getHeartbeat() != null) ?
engine.getHeartbeat().getTime().toString() : "null") + "\n" +
                "Shutdown time: " + ((engine.getShutdownTime()!= null)
? engine.getShutdownTime().getTime().toString() : "null") + "\n" +
                "Start time: " + ((engine.getStartTime() != null) ?
engine.getStartTime().getTime().toString() : "null") );
        }
    }
```

reassignPercentageProcesses

ソースエンジンからターゲットエンジンに、パーセンテージを指定してプロセスを再割り当てする場合に使用します。

メソッドの署名

```
int reassignPercentageProcesses(int percent, java.lang.String
sourceEngineId, java.lang.String targetEngineId)
```

パラメータ

パラメータ	説明
percent	再割り当てするプロセスの割合 (パーセント) を表す整数です。
sourceEngineId	ソースワークフローエンジンの ID です。
targetEngineIds	ターゲットワークフローエンジンの ID です。

reassignProcesses

ソースエンジンからターゲットエンジンに、1 つまたは複数のプロセスを再割り当てする場合に使用します。

メソッドの署名

```
int reassignProcesses(com.novell.soa.af.impl.soap.StringArray
requestIds, java.lang.String sourceEngineId, java.lang.String
targetEngineId)
```

パラメータ

パラメータ	説明
requestIds	再割り当てするプロセスの要求 ID(requestIds) のリストです。
sourceEngineId	ソースワークフローエンジンの ID です。
targetEngineIds	ターゲットワークフローエンジンの ID です。

removeEngine

クラスタ状態テーブルからエンジンを作成する場合に使用します。エンジンの状態は SHUTDOWN または TIMEDOUT でなければなりません。

メソッドの署名

```
void removeEngine(java.lang.String engineId)
```

パラメータ

パラメータ	説明
engineId	削除するワークフローエンジンの ID です。

メトリクス Web サービス 21

この節では、プロビジョニングワークフローのメトリクスを提供する、メトリクス Web サービスについて説明します。主なトピックは次のとおりです。

21.1 メトリクス Web サービスについて

ワークフローエンジンには、ワークフローメトリクスを収集する Web サービスが含まれています。ワークフローエンジンにメトリクス Web サービスを追加することにより、承認フロープロセスを監視することができます。また、ビジネスマネージャがプロセスを変更して最適化するために使用するインジケータとしても利用できます。

メトリクスは、メトリクスは実用的でなければならないとする従来のビジネスプロセスフロー管理の原則をベースにしています。そのため、マネージャがビジネスフローを分析および最適化するときに期待する多くの操作が、メトリクスにより提供されています。メトリクスはボトルネックを特定したり、他の作業に役立つ様々なキャパシティインジケータを提供します。メトリクス Web サービスを利用すれば、測定値を共通の確立されたデータセットに絞り込むことができます。ビジネスプロセスフローに対して作成される無数のメトリクスおよびレポートを処理する必要はありません。

メトリクス Web サービスで作業を行う場合、このサービスは汎用メトリクスシステムを意図していないことに注意してください。

- メトリクス Web サービスは、レポートツールでもなければ、レポートエンジンでもありません。つまり、複雑なクエリ言語は使用していません。
- メトリクス Web サービスは、汎用パフォーマンス管理システムを目的としては設計されていません。そのため、監視対象ライブシステムに対して、必要なクエリが与える影響を制限することができます。

操作管理は、プロセスフローの本質を取得する、3 つの主要な内部プロセスパフォーマンス測定値を重視します。これらの 3 つの測定値は (フロー時間、フローレート、およびインベントリ)、顧客満足度を評価する代表的なインジケータとしての役割を果たします。

操作マネージャはこれらの測定値を使って、次のような疑問に回答できます。

- プロセス境界内でプロビジョニング要求が消費する平均時間は ?(フロー時間)
- 単位時間当たりにプロセス内を流れる平均プロビジョニング要求数は ?(フローレート)
- 任意の時間に、プロセス境界内にある平均プロビジョニング要求数は ? (インベントリ)

これらの 3 つの測定値は、リトルの公式に関係しています。

```
Inventory=Flow Rate*Flow Time
```

21.1.1 Web サービスのセマンティクス

メトリクス Web サービスには、次のセマンティクスが適用されます。

- メトリクス Web サービス内のアクティビティは、ユーザ向きアクティビティ (承認アクティビティ) のみを表します。他のアクティビティは実行時間が短く制御が不可能なため、これらに対する測定値の取得は不適切になってしまいます。
- メトリクス Web サービスは、営業日と暦日を区別しています。暦日は、2 つの日付間のすべての日数です。営業日は、2 つの日付間の営業日のみを表します。営業日は環境によって異なるため、営業日メソッドでは生のデータセットが返されます。これを使って、適切な計算を行うことができます。そのような詳細が必要ない場合は、暦日メソッドを使用すると、適切な測定値が返されます。

21.1.2 Web サービスエンドポイント

メトリクス Web サービスのエンドポイントには、次の URL でアクセスできます。
`http://server:port/warcontext/metrics/service`

21.1.3 セキュリティ許可別にグループ化された Web サービスメソッド

本サービスは、基本認証を使って保護されています。そのため、サービスには SSL を使って接続する必要があります。本サービスは、ユーザアプリケーションと同じセキュリティ層を使用しているため、すべてのユーザに対してすべてのサービス操作が許可されている訳ではありません。プロビジョニング管理者のみが、すべてのメソッドに対する無条件のアクセスを許されています。一方チームマネージャは、自分のチームおよびチームメンバーに関連する測定値にのみアクセスできます。

そのため、メトリクス Web サービスの操作は、役割とセキュリティ許可に応じて 3 種類のカテゴリに分類されています。

- チームマネージャ操作
- プロビジョニングアプリケーション管理者操作
- ユーティリティ操作

チームメトリクス

チームマネージャは、自分がマネージャとなっているチームのメトリクスのみを取得できます。チームマネージャが利用できるメソッドを次に示します。

表 21-1 *チームメトリクスメソッド*

方法	説明
getClaimedFlowTimeCalendarDays	指定した時間間隔内でプロビジョニング要求が引き受けられた平均時間 (時間) を返します
getClaimedFlowTimeWorkingDays	指定された時間間隔でプロビジョニング要求が引き受けられた平均時間を算出するために必要な結果セットを返します

方法	説明
getToClaimedFlowTimeCalendarDays	宛先が利用できるようになった時から、プロビジョニング要求が引き受けられるまでにかかった平均時間 (時間) を返します
getToClaimedFlowTimeWorkingDays	指定した時間間隔内で、宛先が利用できるようになった時から、プロビジョニング要求が引き受けられるまでにかかった平均時間 (時間) を返します
getClaimedInventory	指定した時間間隔内で、引き受けられた平均プロビジョニング要求数を返します
getClaimedThroughputWorkingDays	指定した時間間隔内で、引き受けられた平均プロビジョニング要求数を返します
getTeamLongestRunning	チームのメンバーが宛先の、最も長く実行されている要求の結果セット (秒) を返します
getTeamFlowHistory	指定したプロビジョニング要求リストの、アクティビティの結果セット、宛先、および宛先メッセージを返します
getTeamHistoryForInitiators	チームのメンバーがイニシエータである、プロビジョニング要求とステータスの結果セットを返します
getTeamHistoryForRecipients	チームのメンバーが受信者である、プロビジョニング要求とステータスの結果セットを返します
getTeamRunningTime	指定したプロビジョニング要求が動作している平均時間 (秒) を返します
getTeamDecisionCount	指定したプロビジョニング要求の宛先として、チームが行った決定数を返します
getTeamInitiatedCount	チームが開始したプロビジョニング要求数を返します
getTeamRecipientCount	チームのメンバーが受信者のプロビジョニング要求を返します

プロビジョニング管理者メトリクス

この役割に制限はありません。任意のサービス操作を行えます。これらのメソッドは、プロビジョニング管理者のみが利用できます。

***表 21-2** プロビジョニング管理者メトリクスメソッド*

方法	説明
getActivityFlowTimeCalendarDays	ユーザアクティビティが完了するまでの平均時間を返します
getActivityFlowTimeWorkingDays	ユーザアクティビティが完了するまでの平均時間を算出するために必要な結果セットを返します
getActivityInventory	指定したユーザアクティビティの任意の時点における平均プロビジョニング要求数を返します

方法	説明
getActivityThroughputCalendarDays	指定した時間間隔内で、指定したユーザアクティビティを終了した、時間当たりの平均プロビジョニング要求数を返します
getActivityThroughputWorkingDays	指定した時間間隔内で、プロビジョニング要求の完了に費やした平均時間を算出するために必要な結果セットを返します
getFlowTimeCalendarDays	指定した時間間隔内で、プロビジョニング要求の完了に費やした平均時間 (時間) を返します
getFlowTimeWorkingDays	指定した時間間隔内で、プロビジョニング要求の完了に費やした平均時間を算出するために必要な結果セットを返します
getInventory	指定した時間間隔内の任意の時点で、システム内に存在した平均プロビジョニング要求数を返します
getLongestClaimed	引き受けられたけど処理されていないプロビジョニング要求の結果セットを返します (秒)
getLongestRunning	もっとも長く動作しているプロビジョニング要求の結果セットを返します (秒)
getFlowCount	プロビジョニング要求数を返します
getFlowHistory	指定したプロビジョニング要求リストの、アクティビティの結果セット、宛先、および宛先メッセージを返します
getFlowHistoryForInitiators	指定したイニシエータのプロビジョニング要求とステータスのリストを返します
getFlowHistoryForRecipients	指定した受信者のプロビジョニング要求とステータスのリストを返します
getRunningTime	現在動作中のプロビジョニング要求の平均動作時間を返します
getThroughputCalendarDays	指定した時間間隔内に完了した、1 時間当たりの平均プロビジョニング要求数を返します
getThroughputWorkingDays	指定した時間間隔内に完了した、1 時間当たりの平均プロビジョニング要求数を算出するために必要な結果セットを返します

ユーティリティ操作

これらの操作は、チームマネージャと管理者の両方が実行できます。

表 21-3 ユーティリティ操作

方法	説明
getVersion	Web サービスのバージョンを返します。クライアントコードとサーバコード間のバージョンが一致することを保証するために使用します。
getAllProvisioningFlows	ログインしているユーザが参照できる、プロビジョニングフローのリストを返します
getUserActivityOnlyFlow	プロビジョニングワークフローの GraphViz DOT(http://www.graphviz.org/) 版を返します
getTeams	ログインしているユーザが管理しているチームのリストを返します
getTeamMembers	指定したチームのメンバーリストを返します

21.1.4 フィルタの指定

前述したように、メトリクス Web サービスでは、複雑なクエリ言語は使われていません。代わりにフィルタを使って期間や承認ステータスなどの条件を指定して、結果を絞り込むことができます。

指定できるフィルタを次に示します (サービスの WSDL のタイプ FilterConstants を参照)。

表 21-4 メトリクス結果を絞り込むフィルタ

フィルタ	説明
KEY_ACTIVITY_ID	プロビジョニング要求定義に定義されているユーザアクティビティ ID です。
KEY_APPROVAL_STATUS	プロビジョニング要求の承認ステータスです。次の値を指定できます。 ◆ ApprovalStatusProcessing ◆ ApprovalStatusDenied ◆ ApprovalStatusRefused ◆ ApprovalStatusApproved ◆ ApprovalStatusRetract ◆ ApprovalStatusError
KEY_ENTITLEMENT_STATE	プロビジョニング要求に関連付けられたエンタイトルメントの状態です。次の値が可能です。 ◆ EntitlementUnknown ◆ EntitlementGranted ◆ EntitlementRevoked

フィルタ	説明
KEY_ENTITLEMENT_STATUS	プロビジョニング要求に関連付けられたエンタイトルメントのステータスです。次の値を指定できます。 ◆ EntitlementSuccess ◆ EntitlementWarning ◆ EntitlementError ◆ EntitlementFatal
KEY_INITIATOR	ワークフローイニシエータのユーザ DN です。
KEY_L_COMPLETION_TIME	ワークフロー完了の間隔の開始を示す日付です
KEY_S_COMPLETION_TIME	ワークフロー完了の間隔の終了を示す日付です
KEY_L_ENTITLEMENT_TIME	エンタイトルメント時間の間隔の開始を示す日付です
KEY_S_ENTITLEMENT_TIME	エンタイトルメント時間の間隔の終了を示す日付です
KEY_S_START_TIME	ワークフロー開始の間隔の開始を示す日付です
KEY_L_START_TIME	ワークフロー開始の間隔の終了を示す日付です
KEY_PROCESS_ID	プロビジョニング要求の DN です
KEY_PROCESS_STATUS	プロビジョニング要求のステータスです次の値を指定できます。 ◆ ProcessStatusRunning ◆ ProcessStatusStopped ◆ ProcessStatusTerminated ◆ ProcessStatusCompleted
KEY_PROCESS_VERSION	ワークフローのバージョンに対応するプロセスのバージョンです
KEY_RECIPIENT	ワークフロー受信者のユーザ DN です。
KEY_REQUEST_ID	ワークフローインスタンスに関連付けられた一意の ID です。

Java の例を次に示します。Web サービスクライアントに使用するプラットフォームによっては、コードが大幅に異なることに注意してください。

```
HashMap map=new HashMap();
map.put(MetricsFilter.KEY_PROCESS_STATUS,
MetricsFilter.ProcessStatusRunning);
double flowtime = metrics.getFlowTimeCalendarDays(processId,
      processVer, activity, 5, calendar1.getTime(),
      calendar2.getTime(), MetricsFilter.ACTIVITY_CLAIMED,
      MetricsFilter.ACTIVITY_FORWARDED, map);
...
```

詳細は、WebService WSDL にお問い合わせください。

http://*server*:*port*/*warcontext*/metrics/service?WSDL

21.1.5 スタブクラスの検索

メトリクス Web サービスに必要なスタブクラスは、製品に同梱されている wsclient.jar にあります。この JAR ファイルは、idmuserapp/lib フォルダにあります。作成したコードがこれらのスタブクラスを探せるようにするために、クラスパスにこの JAR ファイルを追加する必要があります。

21.1.6 リモートインタフェースの取得

メトリクス Web サービスのメソッドを呼び出すには、リモートインタフェースへの参照が必要です。

リモートインタフェースを取得するコードの例を次に示します。

```
import java.util.Locale;
import java.util.Properties;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.xml.rpc.Stub;
import com.novell.qa.soap.common.util.LoggerUtils;
import com.novell.qa.soap.common.util.LoginData;
import com.novell.qa.soap.common.util.ServiceUtils;
import com.novell.soa.af.ClusterException;
import com.novell.soa.af.impl.soap.Provisioning;
import com.novell.soa.af.impl.soap.ProvisioningService;
import com.novell.test.automator.framework.TestProgramException;
import com.rational.test.ft.script.RationalTestScript;
import com.novell.soa.af.metrics.soap.MetricsClientHelper;
import com.novell.soa.af.metrics.soap.MetricsStubWrapper;
import com.novell.soa.af.metrics.soap.impl.MetricsService;
import com.novell.soa.af.metrics.soap.impl.MetricsServiceException;
import com.novell.soa.af.metrics.soap.impl.IRemoteMetrics;
/**
* Method to obtain the remote interface to the Metrics endpoint
* @param _url
* @param _username
* @param _password
* @return IRemoteMetrics interface
* @throws Exception
*/
private IRemoteMetrics getStub(String _url, String _username, String
_password) throws Exception

{
   Properties properties = new Properties();
   properties.put(Context.INITIAL_CONTEXT_FACTORY,
"org.jnp.interfaces.NamingContextFactory");
    String lookup =
"xmlrpc:soap:com.novell.soa.af.metrics.soap.impl.MetricsService";
    InitialContext ctx = new InitialContext();
    MetricsService svc = (MetricsService) ctx.lookup(lookup);
```

```
    Stub stub = (Stub)svc.getIRemoteMetricsPort();

    stub._setProperty(Stub.USERNAME_PROPERTY, _username);
    stub._setProperty(Stub.PASSWORD_PROPERTY, _password);
    stub._setProperty(Stub.SESSION_MAINTAIN_PROPERTY, Boolean.TRUE);
    stub._setProperty(Stub.ENDPOINT_ADDRESS_PROPERTY, _url);
    return (IRemoteMetrics) stub;
}
```

上で定義したメソッドを呼び出すコードの例を次に示します。

```
IRemoteMetrics stub = null;
        try
        {
            //
            // Get the stub
            String url = m_loginData.getURL();
            stub = getStub(url,_username,_password);
        }
        catch(Exception e)
        {
            String msg = e.getMessage();
            LoggerUtils.logError(msg);
            throw new TestProgramException(msg);
        }
        return stub;
```

このコードが機能するためには、getStub() メソッドに渡す URL が SOAP エンドポイントを指していなければなりません。次に例を示します。

```
http://myserver:8080/IDMProv/metrics/service
```

ユーザ名は、次のような完全修飾 DN でなければなりません。

```
"cn=admin,ou=idmsample,o=novell"
```

21.1.7 メトリクス環境設定

メトリクス Web サービスがライブシステムに影響を与える設定は、4 つしかありません。これらの設定は、IDMfw.jar/WorkflowService-conf/config.xml ファイルで変更できます。

表 21-5 *メトリクス環境設定*

config.xml ファイル内のキー	説明
<key>Metrics/TimeRequiredBetweenClientRequests</key>	クライアント要求間に必要な時間 (ミリ秒、デフォルトは 250)
<key>Metrics/MaxClients</key>	最大同時クライアントセッション数 (デフォルトは 10)
<key>Metrics/MaxRows</key>	クエリが返すことができる最大行数
<key>Metrics/MaxTeamMembers</key>	最大チームメンバー数

config.xml ファイル内のキー	説明
<key>Metrics/SecondsToAnythingDivider</key>	すべてのスループット計算で使用されるディバイダ (デフォルトは 3600) 元の値はすべて秒です。つまり、すべてのスループットは 1 時間当たりになります。

これらの設定のいずれかが制限値に達すると、問題が発生したことを知らせる Web サービス障害が生成されます。また、設定 1 と 2 の場合は、エラーコードも返されます。

- 障害が TimeRequiredBetweenClientRequests エラーの場合、エラーコードは 100 になります。
- 障害が MaxClients エラーの場合、エラーコードは 200 になります。
- クライアント接続が切断されたことによるエラーの場合、エラーコードは 300 になります。

メトリクス Web サービスのクライアントコンシューマは、要求を再試行するために各自のコードプロビジョンを入れる必要があります。これを行うための Java の簡単な例を次に示します。

```
    try {
          for (int i = 0; i < retries; i++) {
              try {
                  return metrics.getFlowCount(strDN, strId, new
                        HashMap());
              } catch (MetricsServiceException e) {
                      if (e.getErrorCode() == 100 //subsequent call
                            error
                          || e.getErrorCode() == 200) { //too many
                            clients
                          try {
                              Thread.sleep(retryPause);
                          }

                          catch (Exception ex) {
                              // to nothing
                          }
                      } else {
                          throw e2;
                      }
                  } else {
                      throw new RuntimeException(e);
                  }
              } catch (Exception e) {
                 throw e;
              }
          }
          throw new RuntimeException("Did not succeed making
            webservice call");
      } catch (Exception e) {
         throw e;
      }
   }
...
```

21.2 メトリクス Web サービス API

この節では、メトリクス Web サービスで利用できるメソッドの詳細を説明します。

すべてのメソッドが MetricsServiceException と RemoteException をスローします。読みやすくするために、メソッドの署名の throws 節は省略されています。

21.2.1 チームマネージャメソッド

この節は、チームマネージャが利用できる各メソッドの参照情報を取り上げています。

getClaimedFlowTimeCalendarDays

構文 : メソッドの署名を次に示します。

```
double getClaimedFlowTimeCalendarDays(String processId, String
processVersion, Date startCompletionTime, Date endCompletionTime,
String teamDN,Map filters)
```

getClaimedFlowTimeWorkingDays

構文 : メソッドの署名を次に示します。

```
MetricsResultset getClaimedFlowTimeWorkingDays(String processId,
String processVersion, Date startCompletionTime, Date
endCompletionTime,String teamDN, Map filters)
```

getToClaimedFlowTimeCalendarDays

構文 : メソッドの署名を次に示します。

```
 double getToClaimFlowTimeCalendarDays(String processId, String
processVersion, Date startCompletionTime, Date endCompletionTime,
String teamDN,Map filters)
```

getToClaimedFlowTimeWorkingDays

構文 : メソッドの署名を次に示します。

```
  MetricsResultset getToClaimFlowTimeWorkingDays(String processId,
String processVersion, Date startCompletionTime, Date
endCompletionTime,String teamDN, Map filters)
```

getClaimedInventory

構文 : メソッドの署名を次に示します。

```
 double getClaimedInventory(String processId, String processVersion,
Date startCompletionTime, Date endCompletionTime, String teamDN, Map
filters)
```

getClaimedThroughputCalendarDays

構文 : メソッドの署名を次に示します。

```
  double getClaimedThroughputCalendarDays(String processId, String
processVersion, Date startCompletionTime, Date endCompletionTime,
String teamDN Map filters)
```

getClaimedThroughputWorkingDays

構文 : メソッドの署名を次に示します。

```
   MetricsResultset getClaimedThroughputWorkingDays(String processId,
String processVersion, Date startCompletionTime, Date
endCompletionTime, String teamDN, Map filters)
```

getTeamLongestRunning

構文 : メソッドの署名を次に示します。

```
MetricsResultset getTeamLongestRunning(String processId, String
processVersion, String teamDN, Map filters)
```

getTeamLongestClaimed

構文 : メソッドの署名を次に示します。

```
MetricsResultset getTeamLongestClaimed(String processId, String
processVersion, String teamDN, Map filters)
```

getTeamFlowHistory

構文 : メソッドの署名を次に示します。

```
MetricsResultset getTeamFlowHistory(List requestIds)
```

getTeamHistoryForInitiators

構文 : メソッドの署名を次に示します。

```
MetricsResultset getTeamHistoryForInitiators(String teamDN, Map
filters)
```

getTeamHistoryForRecipients

構文 : メソッドの署名を次に示します。

```
MetricsResultset getTeamHistoryForRecipients(String teamDN, Map
filters)
```

getTeamRunningTime

構文 : メソッドの署名を次に示します。

```
double getTeamRunningTime(String processId, String processVersion,
String teamDN, Map filters)
```

getTeamDecisionCount

構文 : メソッドの署名を次に示します。

```
int getTeamDecisionCount(String processId, String processVersion,
String teamDN, Map filters)
```

getTeamInitiatedCount

構文 : メソッドの署名を次に示します。

```
 int getTeamInitiatedCount(String processId, String processVersion,
String teamDN, Map filters)
```

getTeamRecipientCount

構文 : メソッドの署名を次に示します。

```
 int getTeamRecipientCount(String processId, String processVersion,
String teamDN, Map filters)
```

21.2.2 プロビジョニングアプリケーション管理者メソッド

この節は、プロビジョニングアプリケーション管理者が利用できる各メソッドに関する参照情報を取り上げています。

getActivityFlowTimeCalendarDays

構文 : メソッドの署名を次に示します。

```
double getActivityFlowTimeCalendarDays(String processId, String
processVer, String activityId, Date startTime, Date completeTime, Map
filters)
```

getActivityFlowTimeWorkingDays

構文 : メソッドの署名を次に示します。

```
    MetricsResultset getActivityFlowTimeWorkingDays(String processId,
String processVer, String activityId, Date startTime, Date
completeTime, Map filters)
```

getActivityInventory

構文 : メソッドの署名を次に示します。

```
double getActivityInventory(String processId, String processVersion,
String activityId, Date startTime, Date completeTime, Map filters)
```

getActivityThroughputCalendarDays

構文 : メソッドの署名を次に示します。

```
double getActivityThroughputCalendarDays(String processId, String
processVersion, String activityId, Date startTime, Date
completiontime, Map filters)
```

getActivityThroughputWorkingDays

構文 : メソッドの署名を次に示します。

```
 MetricsResultset getActivityThroughputWorkingDays(String processId,
String processVersion, String activityId, Date startTime, Date
completiontime,    Map filters)
```

getInventory

構文 : メソッドの署名を次に示します。

```
double getInventory(String processId, String processVersion, Date
startTime, Date completeTime, Map filters)
```

getLongestClaimed

構文 : メソッドの署名を次に示します。

```
 MetricsResultset getLongestClaimed(String processId, String
processVersion, Map filters)
```

getLongestRunning

構文 : メソッドの署名を次に示します。

```
MetricsResultset getLongestRunning(String processId, String
processVersion, Map filters)
```

getFlowCount

構文 : メソッドの署名を次に示します。

```
int getFlowCount(String processId, String processVersion, Map filters)
```

getFlowHistory

構文 : メソッドの署名を次に示します。

```
MetricsResultset getFlowHistory(List requestIds)
```

getFlowHistoryForInitiators

構文 : メソッドの署名を次に示します。

```
    MetricsResultset getFlowHistoryForInitiators(List initiators, Map
filters)
```

getFlowHistoryForRecipients

構文 : メソッドの署名を次に示します。

```
MetricsResultset getFlowHistoryForRecipients(List recipients, Map
filters)
```

getRunningTime

構文 : メソッドの署名を次に示します。

```
 double getRunningTime(String processId, String processVersion, Map
filters)
```

getThroughputCalendarDays

構文 : メソッドの署名を次に示します。

```
double getThroughputCalendarDays(String processId, String
processVersion, Date startTime, Date completiontime, Map filters)
```

getThroughputWorkingDays

構文 : メソッドの署名を次に示します。

```
MetricsResultset getActivityThroughputWorkingDays(String processId,
String processVersion, String activityId, Date startTime, Date
completiontime,    Map filters)
```

21.2.3 ユーティリティメソッド

この節は、各ユーティリティメソッドに関する参照情報を取り上げています。これらのメソッドは、チームマネージャと管理者の両方が呼び出すことができます。

getVersion

構文 : メソッドの署名を次に示します。

```
VersionVO getVersion()
```

getAllProvisioningFlows

構文 : メソッドの署名を次に示します。

```
MetricsResultset getAllProvisioningFlows()
```

getUserActivityOnlyFlow

構文 : メソッドの署名を次に示します。

```
BasicModelVO getUserActivityOnlyFlow(String processId, String
processVer)
```

getTeams

構文 : メソッドの署名を次に示します。

```
MetricsResultset getTeams()
```

getTeamMembers

構文 : メソッドの署名を次に示します。

```
MetricsResultset getTeamMembers(String teamDN)
```

21.3 メトリクス Web サービスの例

この節では、メトリクス Web サービスを使ったワークフローメトリクスの収集方法の例を説明していきます。この例では、453 ページのセクション 21.1.6「リモートインタフェースの取得」に記載されているスタブを取得し、それをエラー状態を処理するオブジェクト (454 ページのセクション 21.1.7「メトリクス環境設定」を参照) にラップすることを前提にしています。

21.3.1 一般的な例

この例では、KEY_APPROVAL_STATUS フィルタを使って、プロビジョニング要求タイプの決定の成果を比較します。これを使って、たとえば円グラフを生成できます。

```
FilterConstants constants=new FilterConstants();
Map<MetricsFilter, Object> map = new HashMap<MetricsFilter, Object>();
map.put(MetricsFilter.KEY_APPROVAL_STATUS,constants.getApprovalStatusA
pproved());
```

```
double
accepted=stubWrapper.getFlowCount(processId,processVersion,map);
map.put(MetricsFilter.KEY_APPROVAL_STATUS,constants.getApprovalStatusD
enied());
double
denied=stubWrapper.getFlowCount(processId,processVersion,map);map.put(
MetricsFilter.KEY_APPROVAL_STATUS,constants.getApprovalStatusError());
double error=stubWrapper.getFlowCount(processId,processVersion,map);
map.put(MetricsFilter.KEY_APPROVAL_STATUS,constants.getApprovalStatusR
etract());
double
retracted=stubWrapper.getFlowCount(processId,processVersion,map);
map.put(MetricsFilter.KEY_APPROVAL_STATUS,constants.getApprovalStatusR
efused());
double refused = stubWrapper.getFlowCount(processId,
processVersion, map);
```

また、適切なエントリをフィルタマップに追加することにより、他のフィルタを指定することもできます。以降の例は、さまざまなタイプのフィルタの追加方法を表しています。

開始日フィルタの追加

開始日フィルタを追加する (01/01/2006 < 日付 < 02/01/2006)

```
Calendar startDate=Calendar.getInstance();
startDate.set(2006,0,1);
Calendar endDate=Calendar.getInstance();
endDate.set(2006,1,1);
map.put(MetricsFilter.KEY_L_START_TIME,startDate);
map.put(MetricsFilter.KEY_S_START_TIME,endDate)
```

完了日フィルタの追加

完了日フィルタを追加する (02/01/2005 < 日付 < 03/01/2005)

```
Calendar startDate=Calendar.getInstance();
startDate.set(2006,0,1);
Calendar endDate=Calendar.getInstance();
endDate.set(2006,1,1);
map.put(MetricsFilter.KEY_L_COMPLETION_TIME,startDate);
map.put(MetricsFilter.KEY_S_COMPLETION_TIME,endDate)
```

特定のイニシエータへの要求への絞り込み

カウントするリクエストを特定のイニシエータに絞り込む

```
map.put(MetricsFilter.KEY_INITIATOR,"cn=admin,ou=idmsample,o=novell");
```

特定の受信者への要求への絞り込み

カウントするリクエストを特定の受信者に絞り込む

```
map.put(MetricsFilter.KEY_RECIPIENT,"cn=admin,ou=idmsample,o=novell");
```

21.3.2 その他の例

ワークフローカウントを取得するために、さまざまなメソッドを使用する例を次に示します。

チームの決定カウントの取得

この例は、チームのさまざまな決定成果の取得方法を表しています。チームの DN が必要です。これは、getTeams() メソッドを使って取得できます。

```
FilterConstants constants=new FilterConstants();
Map<MetricsFilter, Object> map = new HashMap<MetricsFilter, Object>();
map.put(MetricsFilter.KEY_ACTIVITY_END,
constants.getActivityApproved());
double accepted = stubWrapper.getTeamDecisionCount(processId,
processVersion, teamDN, map);
map.put(MetricsFilter.KEY_ACTIVITY_END,
constants.getActivityDenied());
double denied = stubWrapper.getTeamDecisionCount(processId,
processVersion, teamDN, map);
map.put(MetricsFilter.KEY_ACTIVITY_END,
         constants.getActivityReassigned());
double reassigned = stubWrapper.getTeamDecisionCount(processId,
         processVersion, teamDN, map);
map.put(MetricsFilter.KEY_ACTIVITY_END,
         constants.getActivityRefused());
double refused = stubWrapper.getTeamDecisionCount(processId,
         processVersion, teamDN, map);
```

チームメンバーが受信者の要求の決定カウントの取得

次の例は、チームのメンバーが受信者である要求の決定の成果を取得する方法を表しています。

```
FilterConstants constants = new FilterConstants();
Map<MetricsFilter, Object> map = new HashMap<MetricsFilter, Object>();
map.put(MetricsFilter.KEY_APPROVAL_STATUS,
constants.getActivityApproved());
double accepted = stubWrapper.getTeamRecipientCount(processId,
processVersion, teamDN, map);
map.put(MetricsFilter.KEY_APPROVAL_STATUS,
constants.getApprovalStatusDenied());
double denied = stubWrapper.getTeamRecipientCount(processId,
processVersion, teamDN, map);
map.put(MetricsFilter.KEY_APPROVAL_STATUS,
constants.getApprovalStatusError());
double error = stubWrapper.getTeamRecipientCount(processId,
processVersion, teamDN, map);
map.put(MetricsFilter.KEY_APPROVAL_STATUS,
constants.getApprovalStatusError());
double retracted = stubWrapper.getTeamRecipientCount(processId,
processVersion, teamDN, map);
map.put(MetricsFilter.KEY_APPROVAL_STATUS,
constants.getApprovalStatusRefused());
```

```
double refused = stubWrapper.getTeamRecipientCount(processId,
processVersion, teamDN, map);
```

引き受けられたけれどもまだ処理されていない要求の取得

この例は、2006 年 3 月 1 日以降に開始されて引き受けられたけれども、まだ処理されていない要求を取得する方法を表しています。

```
Map<MetricsFilter, Object> map = new HashMap<MetricsFilter, Object>();
Calendar startDate=Calendar.getInstance();
startDate.set(2006,2,1);
map.put(MetricsFilter.KEY_L_START_TIME,startDate);
MetricsResultset rset = stubWrapper.getLongestClaimed(processId,
processVersion, map);
```

特定のユーザが開始した、もっとも長く動作しているの要求の取得

この例は、イニシエータ「cn=admin,ou=idmsample,o=novell」が開始した、もっとも長く動作している要求の取得方法を表しています。

```
Map<MetricsFilter, Object> map = new HashMap<MetricsFilter, Object>();
map.put(MetricsFilter.KEY_INITIATOR,""cn=admin,ou=idmsample,o=novell")
;
MetricsResultset rset = stubWrapper.getLongestRunning(processId,
processVersion, map);
```

アクティビティインベントリの取得

この例は、アクティビティ ID「managerApproval」で、2006 年 1 月 1 日から 2006 年 2 月 1 日まで決定を処理したユーザの平均インベントリの取得方法を表しています。

```
Map<MetricsFilter, Object> map = new HashMap<MetricsFilter, Object>();
Calendar startDate=Calendar.getInstance();
startDate.set(2006,0,1);
Calendar endDate=Calendar.getInstance();
endDate.set(2006,1,1);
MetricsResultset rset = stubWrapper.getActivityInventory(processId,
processVersion,"managerApproval", startDate, endDate, map );
```

引き受けられたスループットとインベントリの取得

この例は、2006 年 1 月 1 日から 2006 年 2 月 1 日の時間間隔のチームのスループットとインベントリを取得する方法を表しています。

```
Map<MetricsFilter, Object> map = new HashMap<MetricsFilter, Object>();
Calendar startDate=Calendar.getInstance();
startDate.set(2006,0,1);
Calendar endDate=Calendar.getInstance();
endDate.set(2006,1,1);
double throughput =
stubWrapper.getClaimedThroughputCalendarDays(processId,
processVersion, startDate, endDate,teamDN, map);
double inventory = stubWrapper.getClaimedInventory(processId,
processVersion, startDate, endDate, teamDN, map)
```

通知 Web サービス

22

この節では、SOAP クライアントで電子メール通知機能を利用するための、通知 Web サービスについて説明します。主なトピックは次のとおりです。

22.1 通知 Web サービスについて

Identity Manager プロビジョニングモジュールには、電子メール通知機能が用意されています。これを使って、ユーザにプロビジョニングシステムのステータスの変更や、ユーザが実行する必要があるタスクを知らせるメッセージを電子メールで送信することができます。サードパーティのソフトウェアアプリケーションによるアクセスをサポートするために、通知機能には Web サービスエンドポイントが用意されています。エンドポイントにより、電子メールメッセージを 1 人または複数のユーザに送信することができます。電子メールを送信する場合、ターゲット電子メールアドレス、使用する電子メールテンプレート、および電子メールテンプレート内のトークンの置換値を指定するパラメータを入れます。

この付録では、通知 Web サービスのプログラミングインタフェースについて説明していきます。

22.1.1 テストページへのアクセス

通知 Web サービスエンドポイントへのアクセスには、次のような URL を使用します。

```
http://server:port/warcontext/notification/service?test
```

たとえば、サーバ名が「myserver」で、ユーザアプリケーションがポート 8080 で待機しており、ユーザアプリケーション WAR ファイル名が「IDMPROV」の場合、URL は次のようになります。

```
http://myserver:8080/IDMPROV/notification/service?test
```

22.1.2 WSDL へのアクセス

通知 Web サービスの WSDL へのアクセスには、次のような URL を使用します。

```
http://server:port/warcontext/notification/service?wsdl
```

たとえば、サーバ名が「myserver」で、ユーザアプリケーションがポート 8080 で待機しており、ユーザアプリケーション WAR ファイル名が「IDMPROV」の場合、URL は次のようになります。

```
http://myserver:8080/IDMPROV/notification/service?wsdl
```

22.1.3 スタブクラスの検索

通知 Web サービスに必要なスタブクラスは、製品に同梱されている wsclient.jar にあります。この JAR ファイルは、idmuserapp/lib フォルダにあります。作成したコードがこれらのスタブクラスを探せるようにするために、クラスパスにこの JAR ファイルを追加する必要があります。

22.2 通知 Web サービス API

この節では、通知 Web サービスで利用できるメソッドの詳細を説明します。この API は、WSSDK ツールキットで生成した Java コードを使用することを前提にしています。別の Web サービスツールキットを使用する場合、API は異なります。

すべてのメソッドが RemoteException をスローします。読みやすくするために、メソッドの署名の throws 節は省略されています。

22.2.1 iRemoteNotification

この節では、iRemoteNotification インタフェースに関連する各メソッドの参照情報を説明しています。

getVersion

実行中の通知機能のバージョン番号を返します。

構文 : メソッドの署名を次に示します。

```
VersionVO getVersion()
```

sendNotification

電子メール通知を送信します。

構文 : メソッドの署名を次に示します。

```
void sendNotification(NotificationMap arg0)
```

22.2.2 BuiltInTokens

この節では、BuiltInTokens クラスに関連する各メソッドの参照情報を説明しています。

BuiltInTokens コンストラクタ

BuiltInTokens クラスには、1 つのコンストラクタがあります。

構文 : BuiltInTokens クラスのコンストラクタを次に示します。

```
BuiltInTokens()
```

getTO

固定文字列 TO を返します。これをキーとして、TO システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getTO()
```

getCC

固定文字列 CC を返します。これをキーとして、CC システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getCC()
```

getBCC

固定文字列 BCC を返します。これをキーとして、BCC システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getBCC()
```

getTO_DN

固定文字列 TO_DN を返します。これをキーとして、TO_DN システムトークンの値を識別することができます。

構文 メソッドの署名を次に示します。

```
public java.lang.String getTO_DN()
```

getCC_DN

固定文字列 CC_DN を返します。これをキーとして、CC_DN システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getCC_DN()
```

getBCC_DN

固定文字列 BCC_DN を返します。これをキーとして、BCC_DN システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getBCC_DN()
```

getREPLYTO

固定文字列 REPLYTO を返します。これをキーとして、REPLYTO システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getREPLYTO()
```

getREPLYTO_DN

固定文字列 REPLYTO_DN を返します。これをキーとして、REPLYTO_DN システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getREPLYTO_DN()
```

getLOCALE

固定文字列 LOCALE を返します。これをキーとして、LOCALE システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getLOCALE()
```

getNOTIFICATION_TEMPLATE_DN

固定文字列 NOTIFICATION_TEMPLATE を返します。これをキーとして、NOTIFICATION_TEMPLATE システムトークンの値を識別することができます。

構文 : メソッドの署名を次に示します。

```
public java.lang.String getNOTIFICATION_TEMPLATE_DN()
```

22.2.3 エントリ

Entry クラスは、EntryArray オブジェクト内のエントリを表します。これは、電子メールテンプレートのトークンを指定する場合に使用します。

この節では、Entry クラスに関連する各メソッドの参照情報を説明しています。

Entry コンストラクタ

Entry クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
Entry()
```

構文 2: キーの値と値の配列の 2 つのパラメータを取得するコンストラクタの構文を次に示します。

```
Entry(java.lang.String KeyVal, StringArray ValuesVal)
```

getKey

Entry オブジェクトに定義されているキーを返します。このキーは、トークンを識別します。

構文 : メソッドの署名を次に示します。

```
java.lang.String getKey()
```

setKey

Entry オブジェクトにキーを設定します。このキーは、トークンを識別します。オブジェクトがビルトインのトークンを表している場合は、BuiltInTokens を使ってキーを設定できます。そうでない場合は、キーを指定する setKey メソッドに文字列を渡す必要があります。

構文 : メソッドの署名を次に示します。

```
void setKey(java.lang.String KeyVal)
```

getValues

Entry オブジェクトの値を表す StringArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
StringArray getValues()
```

setValues

Entry オブジェクトに値を設定します。

構文 : メソッドの署名を次に示します。

```
void setValues(StringArray ValuesVal)
```

22.2.4 EntryArray

EntryArray クラスは、Entry オブジェクトの配列のコンテナです。これは、NotificationMap オブジェクトにより保管されます。

この節では、EntryArray クラスに関連するメソッドの参照情報を説明しています。

EntryArray コンストラクタ

EntryArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
EntryArray()
```

構文 2: Entry オブジェクトの配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
EntryArray(Entry[] EntryVal)
```

getEntry

EntryArray オブジェクト内に保管されている Entry オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
Entry[] getEntry()
```

setEntry

この EntryArray オブジェクトに Entry オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setEntry(Entry[] EntryVal)
```

22.2.5 NotificationMap

NotificationMap オブジェクトは、EntryArray オブジェクトを保管しているマップです。これは、スタブの sendNotification メソッドに渡されます。

この節では、NotificationMap クラスに関連するメソッドの参照情報を説明しています。

NotificationMap コンストラクタ

NotificationMap クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
NotificationMap()
```

構文 2: EntryArray オブジェクトをパラメータとして取得するコンストラクタの構文を次に示します。

```
NotificationMap(EntryArray EntriesVal)
```

getEntries

この NotificationMap オブジェクトに保管されている EntryArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
EntryArray getEntries()
```

setEntries

この NotificationMap オブジェクトに EntryArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setEntries(EntryArray EntriesVal)
```

22.2.6 NotificationService

この節では、NotificationService インタフェースに関する参照情報を説明しています。

getIRemoteNotificationPort

リモートサービスのスタブを取得します。このスタブは、タイプ IRemoteNotification のポートです。

構文 : メソッドの署名を次に示します。

```
IRemoteNotification getIRemoteNotificationPort() throws
javax.xml.rpc.ServiceException
```

22.2.7 StringArray

この節では、StringArray クラスに関する参照情報を説明しています。

StringArray コンストラクタ

StringArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
StringArray()
```

構文 2: 文字列配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
StringArray(java.lang.String[] StringVal)
```

getString

この StringArray オブジェクトに定義されている文字列の配列を返します。

構文 : メソッドの署名を次に示します。

```
java.lang.String[] getString()
```

setString

この StringArray オブジェクトに文字列の配列を設定します。このメソッドは、文字列配列をパラメータとして取得する 2 番目のコンストラクタにより呼び出されます。

構文 : メソッドの署名を次に示します。

```
void setString(java.lang.String[] StringVal)
```

22.2.8 VersionVO

この節では、VersionVO クラスに関する参照情報を説明しています。

getValue

サービスのバージョン番号を返します。

構文 : メソッドの署名を次に示します。

```
java.lang.String getValue()
```

22.3 通知の例

次のコード例は、通知サービスを使って、事前定義されたシステムテンプレートで電子メールメッセージを送信する方法を表しています。通知サービスの SOAP エンドポイントの参照を取得するために、getNotificationStub() メソッドが呼び出されます。スタブインタフェースを取得したら、電子メール通知テンプレートとテンプレート内のビルトイントークンの値が設定されます。また、requestTitle と initiatorFullName の値も指定されます。このコードは、各トークンに対して、Entry オブジェクトを作成します。すべてのエントリが作成されたら、Entry 配列がタイプ NotificationMap のマップにパッケージ化されます。次に NotificationMap が、スタブの sendNotification メソッドに渡されます。

```
import java.util.Properties;
import javax.naming.Context;
import javax.naming.InitialContext;
import javax.xml.rpc.Stub;
import java.rmi.RemoteException;
//
// Notification imports
import com.novell.ws.client.notification.IRemoteNotification;
import com.novell.ws.client.notification.BuiltInTokens;
import com.novell.ws.client.notification.Entry;
import com.novell.ws.client.notification.EntryArray;
import com.novell.ws.client.notification.StringArray;
import com.novell.ws.client.notification.NotificationMap;
import com.novell.ws.client.notification.IRemoteNotification;
import com.novell.ws.client.notification.NotificationService;
```

```
public class NotificationTest
{
      private static final int LOCALHOST  = 0;  // localhost
      private static final int TESTSERVER  = 1;  // testserver
      private static final int SELECTED_URL     = TESTSERVER;

      private String [] SERVER_URLS = {
            "http://localhost:8080/IDMProv/notification/service",
            "http://testserver:8080/IDMProv/notification/service"
      };
      private String url = SERVER_URLS[SELECTED_URL];
      private String username = "cn=admin,ou=idmsample,o=novell";
      private String password = "test";

      public void emailNotificationTestCase()
      throws Exception
      {
        System.out.println("\nCalling emailNotificationTestCase() test
case");

        try
        {
            String targetEmailAddress = "jsmith@somewhere.com";
            //
            // Get the notification stub
            IRemoteNotification notificationStub =
getNotificationStub(url, username, password);

            BuiltInTokens builtInTokens = new BuiltInTokens();
            //
            // Set the To: entry
            Entry to = new Entry();
            to.setKey(builtInTokens.getTO());
            StringArray arr = new StringArray(new
String[]{targetEmailAddress} );
            to.setValues(arr);
            //
            // Set which email template to use : list in iManager
(Workflow Admin->Email Templates)
            Entry notificationTemplate = new Entry();

notificationTemplate.setKey(builtInTokens.getNOTIFICATION_TEMPLATE_DN(
));
            //
            // Use one of the email templates specifying DN
            String EMAIL_TEMPLATE_NAME = "Provisioning Notification";
            String templateDN = "cn=" + EMAIL_TEMPLATE_NAME +
",cn=Default Notification Collection,cn=Security";
            arr = new StringArray(new String[]{templateDN} );
            notificationTemplate.setValues(arr);
            //
            // Substitute key values defined in email templates
            Entry token1 = new Entry();
```

```
            token1.setKey("requestTitle"); // key is %requestTitle%
            arr = new StringArray(new String[]{"Sample Email using
Notification Web Service"} );
            token1.setValues(arr);
            Entry token2 = new Entry();
            token2.setKey("initiatorFullName");
            arr = new StringArray(new String[]{username} );
            token2.setValues(arr);
            //
            // Setup the notification map
            NotificationMap map = new NotificationMap();
            Entry[] entries = new
Entry[]{to,notificationTemplate,token1,token2};
            EntryArray entryArray = new EntryArray();
            entryArray.setEntry(entries);
            map.setEntries(entryArray);
            //
            // Make the notification endpoint call
            notificationStub.sendNotification(map);
        }
        catch(RemoteException error)
        {
            System.out.println(error.getMessage() );
            throw new Exception(error.getMessage() );
        }
     }


   /**
     * Method to obtain the remote interface to the Notification
endpoint
     * @param _url
     * @param _username
     * @param _password
     * @return IRemoteNotification interface
     * @throws Exception
     */
    private IRemoteNotification getNotificationStub(String _url,
String _username, String _password)
    throws Exception
    {
        Properties properties = new Properties();
        properties.put(Context.INITIAL_CONTEXT_FACTORY,
"org.jnp.interfaces.NamingContextFactory");

        String lookup =
"xmlrpc:soap:com.novell.ws.client.notification.NotificationService";

        InitialContext ctx = new InitialContext();
        NotificationService svc = (NotificationService)
ctx.lookup(lookup);

        Stub stub = (Stub)svc.getIRemoteNotificationPort();
```

```
        stub._setProperty(Stub.USERNAME_PROPERTY, _username);
        stub._setProperty(Stub.PASSWORD_PROPERTY, _password);
        stub._setProperty(Stub.SESSION_MAINTAIN_PROPERTY,
Boolean.TRUE);
        stub._setProperty(Stub.ENDPOINT_ADDRESS_PROPERTY, _url);

        return (IRemoteNotification) stub;
    }

}
```

ディレクトリ抽象化層 (VDX)Web サービス

23

この節では、SOAP クライアントがディレクトリ抽象化層にアクセスするための、VDX Web サービスについて説明します。主なトピックは次のとおりです。

23.1 ディレクトリ抽象化層 (VDX) Web サービスについて

ディレクトリ抽象化層は、アイデンティティボールトデータの論理的なビューを提供しています。サードパーティソフトウェアアプリケーションからのアクセスをサポートするために、ディレクトリ抽象化層サービスには、VDXWeb サービスと呼ばれる Web サービスエンドポイントが用意されています。このエンドポイントを利用すれば、ディレクトリ抽象化層に定義されているエンティティに関連する属性にアクセスすることができます。また、グローバル検索と呼ばれる事前定義された検索を実行したり、特定のエンティティを検索することができます。グローバルクエリを LDAP のプロシージャに保管することができます。

この付録では、VDX Web サービスのプログラミングインタフェースについて説明していきます。

23.1.1 テストページへのアクセス

VDX Web サービスエンドポイントへのアクセスには、次のような URL を使用します。

```
http://server:port/warcontext/vdx/service?test
```

たとえば、サーバ名が「myserver」で、ユーザアプリケーションがポート 8080 で待機しており、ユーザアプリケーション WAR ファイル名が「IDMPROV」の場合、URL は次のようになります。

```
http://myserver:8080/IDMPROV/vdx/service?test
```

23.1.2 WSDL へのアクセス

VDX Web サービスの WSDL へのアクセスには、次のような URL を使用します。

```
http://server:port/warcontext/vdx/service?wsdl
```

たとえば、サーバ名が「myserver」で、ユーザアプリケーションがポート 8080 で待機しており、ユーザアプリケーション WAR ファイル名が「IDMPROV」の場合、URL は次のようになります。

```
http://myserver:8080/IDMPROV/vdx/service?wsdl
```

23.1.3 スタブクラスの検索

VDX Web サービスに必要なスタブクラスは、製品に同梱されている wsclient.jar にあります。この JAR ファイルは、idmuserapp/lib フォルダにあります。作成したコードがこれらのスタブクラスを探せるようにするために、クラスパスにこの JAR ファイルを追加する必要があります。

23.2 VDX Web サービス API

この節では、VDX Web サービスで利用できるメソッドの詳細を説明します。この API は、WSSDK ツールキットで生成した Java コードを使用することを前提にしています。別の Web サービスツールキットを使用する場合、API は異なります。

すべてのメソッドが VdxServiceException をスローします。読みやすくするために、メソッドの署名の throws 節は省略されています。

23.2.1 IRemoveVdx

この節では、IRemoveVdx インタフェースに関連する各メソッドの参照情報を説明しています。

getVersion

実行中の VDX サービスのバージョン番号を返します。

構文 : メソッドの署名を次に示します。

```
VersionVO getVersion() throws java.rmi.RemoteException;
```

globalQuery

あらかじめ定義された、グローバルクエリと呼ばれる検索を実行できます。グローバルクエリは、LDAP の検索に保存されます。これらは、一部のストアドプロシージャの機能を提供しています。

グローバルクエリを定義するには、ディレクトリ抽象化層エディタを使用する必要があります。詳細は、『*Identity Manager ユーザアプリケーション : 設計ガイド*』のディレクトリ抽象化層エディタに関する項目を参照してください。

構文 : メソッドの署名を次に示します。

```
java.lang.String[] globalQuery(java.lang.String queryDN, StringMap
queryParameterValues) throws VdxServiceException,
java.rmi.RemoteException;
```

クエリー

エンティティ、一連の属性、返されるデータをフィルタリングするクエリ式を指定したクエリを実行できます。

構文 : メソッドの署名を次に示します。

```
EntityAttributeMap query(java.lang.String entityDefinition,
java.lang.String[] attributeKeys, java.lang.String queryFilter) throws
VdxServiceException, java.rmi.RemoteException;
```

クエリの文法

query() メソッドの queryFilter パラメータを使って、返されるデータをフィルタリングする検索条件の式を渡すことができます。この節では、これらの式の文法について説明します。

クエリ構文 1: もっとも簡単な形式のクエリを次に示します。
RelationalExpression1

クエリ構文 2: クエリでは、論理演算子を使って複数の式を組み合わせることができます。
RelationalExpession1 logicalOperator RelationalExpression2

クエリ構文 3: 代わりに、括弧を使って式をセットオフすることができます。
(RelationalExpession1) logicalOperator (RelationalExpression2)

クエリ構文 4: 括弧を使って、サブクエリをセットオフすることもできます。
RelationalExpession1 logicalOperator (RelationalExpression2 logicalOperator1 RelationalExpression3)

関係式は、同じ論理演算子で区切る必要があります。有効なクエリの列を次に示します。
`expression1 AND expression2 AND expression3`

次のクエリは無効になります。
`expression1 AND expression2 OR expression3`

次の例のように、括弧を使って条件グループを作成することができます。
`expression1 AND (expression2 OR expression3)`

関係式の文法

関係式の構文 : 関係式は、この構文に準拠していなければなりません。
attribute relationalOperator value

演算子と値の文法

関係演算子 : 関係演算子は次のいずれかでなければなりません。
`> , < , >= , <= , = , != , !< , !> , !<= , !>= , STARTWITH, !STARTWITH, IN , !IN , PRESENT, !PRESENT`

論理演算子 : 論理演算子は次のいずれかでなければなりません。
`AND, OR`

値 : 値側の式は、次のいずれかでなければなりません。
`'foo',"foo", 1-9, true, false`

関係演算子 PRESENT と !PRESENT には、値は必要ありません。

getAttribute

単一の Attribute オブジェクトを返します。これを使って、ディレクトリ抽象化層の属性のデータを取得して、調べることができます。

構文 : メソッドの署名を次に示します。

```
Attribute getAttribute(java.lang.String objectDN, java.lang.String
entityDefinition, java.lang.String attributeKey) throws
VdxServiceException, java.rmi.RemoteException;
```

getAttributes

Attribute オブジェクトの配列を返します。これを使って、ディレクトリ抽象化層の属性のデータを取得して、調べることができます。

構文 : メソッドの署名を次に示します。

```
Attribute[] getAttributes(java.lang.String objectDN, java.lang.String
entityDefinition, java.lang.String[] attributeKeys) throws
VdxServiceException, java.rmi.RemoteException;
```

23.2.2 属性

Attribute クラスは、ディレクトリ抽象化層の属性を表しています。

この節では、Attribute クラスに関する参照情報を説明しています。

Attribute コンストラクタ

Attribute クラスには、2 つのコンストラクタがあります。

構文 1: 引数を取得しないコンストラクタの構文を次に示します。

```
Attribute()
```

構文 2: サポートするすべてのデータタイプの配列を引数として取得するコンストラクタの構文を次に示します。

```
Attribute(ByteArrayArray BinariesVal, BooleanArray BooleansVal,
DateArray DatesVal, IntegerArray IntegersVal, StringArray StringsVal,
AttributeType TypeVal)
```

getBinaries

属性の ByteArrayArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
ByteArrayArray getBinaries()
```

setBinaries

属性の ByteArrayArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setBinaries(ByteArrayArray BinariesVal)
```

getBooleans

属性の BooleanArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
BooleanArray getBooleans()
```

setBooleans

属性の BooleanArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setBooleans(BooleanArray BooleansVal)
```

getDates

属性の DateArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
DateArray getDates()
```

setDates

属性の DateArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setDates(DateArray DatesVal)
```

getIntegers

属性の IntegerArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
IntegerArray getIntegers()
```

setIntegers

属性の IntegerArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setIntegers(IntegerArray IntegersVal)
```

getStrings

属性の StringArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
StringArray getStrings()
```

setStrings

属性の StringArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setStrings(StringArray StringsVal)
```

getType

属性の AttributeType オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
AttributeType getType()
```

setType

属性の AttributeType オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setType(AttributeType TypeVal)
```

23.2.3 AttributeArray

この節では、AttributeArray クラスに関する参考情報を説明しています。

AttributeArray コンストラクタ

AttributeArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
AttributeArray()
```

構文 2: AttributeArray オブジェクトの配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
AttributeArray(Attribute[] AttributeVal)
```

getAttribute

Attribute オブジェクトの配列を返します。

構文 : メソッドの署名を次に示します。

```
Attribute[] getAttribute()
```

setAttribute

AttributeArray クラスに関連する Attribute オブジェクトの配列を設定します。

構文 : メソッドの署名を次に示します。

```
void setAttribute(Attribute[] AttributeVal)
```

23.2.4 AttributeType

この節では、AttributeType クラスに関する参考情報を説明しています。

AttributeType コンストラクタ

AttributeType クラスは、1 つのコンストラクタをサポートしています。

構文 : コンストラクタの構文を次に示します。

```
protected AttributeType(java.lang.String value)
```

getValue

属性タイプを示す文字列を返します。

構文 : メソッドの署名を次に示します。

```
java.lang.String getValue()
```

23.2.5 BooleanArray

この節では、BooleanArray クラスに関する参考情報を説明しています。

BooleanArray コンストラクタ

BooleanArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
BooleanArray()
```

構文 2: ブール値をパラメータとして取得するコンストラクタの構文を次に示します。

```
BooleanArray(boolean[] BooleanVal)
```

getBoolean

属性のブール値の配列を返します。

構文 : メソッドの署名を次に示します。

```
boolean[] getBoolean()
```

setBoolean

属性にブール値の配列を設定します。

構文 : メソッドの署名を次に示します。

```
void setBoolean(boolean[] BooleanVal)
```

23.2.6 ByteArrayArray

この節では、ByteArrayArray クラスに関する参考情報を説明しています。

ByteArrayArray コンストラクタ

ByteArrayArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
ByteArrayArray()
```

構文 2: Base 64 バイナリ値をパラメータとして取得するコンストラクタの構文を次に示します。

```
ByteArrayArray(byte[][] Base64BinaryVal)
```

getBase64Binary

属性のバイトの 2 次元配列を返します。

構文 : メソッドの署名を次に示します。

```
byte[][] getBase64Binary()
```

setBase64Binary

属性にバイトの 2 次元配列を設定します。

構文 : メソッドの署名を次に示します。

```
void setBase64Binary(byte[][] Base64BinaryVal)
```

23.2.7 DateArray

この節では、DateArray クラスに関する参考情報を説明しています。

DateArray コンストラクタ

DateArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
DateArray()
```

構文 2: カレンダ配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
DateArray(java.util.Calendar[] DatetimeVal)
```

getDatetime

属性のカレンダオブジェクトの配列を返します。

構文 : メソッドの署名を次に示します。

```
java.util.Calendar[] getDatetime()
```

setDatetime

属性にカレンダオブジェクトの配列を設定します。

構文 : メソッドの署名を次に示します。

```
void setDatetime(java.util.Calendar[] DatetimeVal)
```

23.2.8 EntryAttributeMap

EntryAttributeMap クラスは、EntryArray オブジェクトのコンテナです。これは、スタブのクエリメソッドにより返されます。

この節では、EntryAttributeMap クラスに関連するメソッドの参考情報を説明しています。

EntryAttributeMap コンストラクタ

EntryAttributeMap クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
EntryAttributeMap()
```

構文 2: EntryArray オブジェクトをパラメータとして取得するコンストラクタの構文を次に示します。

```
EntityAttributeMap(EntryArray EntriesVal)
```

getEntries

この EntryAttributeMap オブジェクト内に保管されている EntryArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。
```
EntryArray getEntries()
```

setEntries

この EntryAttributeMap オブジェクトに EntryArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。
```
void setEntry(EntryArray EntriesVal)
```

23.2.9 エントリ

Entry クラスは、EntryArray オブジェクト内のエントリを表します。

この節では、Entry クラスに関連する各メソッドの参照情報を説明しています。

Entry コンストラクタ

Entry クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。
```
Entry()
```

構文 2: キーの値と属性値の配列の 2 つのパラメータを取得するコンストラクタの構文を次に示します。
```
Entry(java.lang.String KeyVal, AttributeArray ValuesVal)
```

getKey

Entry オブジェクトに定義されているキーを返します。このキーは、属性を識別します。

構文 : メソッドの署名を次に示します。
```
java.lang.String getKey()
```

setKey

Entry オブジェクトにキーを設定します。このキーは、属性を識別します。

構文 : メソッドの署名を次に示します。
```
void setKey(java.lang.String KeyVal)
```

getValues

Entry オブジェクトの値を表す AttributeArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。
```
AttributeArray getValues()
```

setValues

Entry オブジェクトに値を設定します。

構文 : メソッドの署名を次に示します。
```
void setValues(AttributeArray ValuesVal)
```

23.2.10 EntryArray

EntryArray クラスは、Entry オブジェクトの配列のコンテナです。これは、EntryAttributeMap オブジェクトにより保管されます。

この節では、EntryArray クラスに関連するメソッドの参照情報を説明しています。

EntryArray コンストラクタ

EntryArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
EntryArray()
```

構文 2: Entry オブジェクトの配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
EntryArray(Entry[] EntryVal)
```

getEntry

EntryArray オブジェクト内に保管されている Entry オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
Entry[] getEntry()
```

setEntry

この EntryArray オブジェクトに Entry オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setEntry(Entry[] EntryVal)
```

23.2.11 IntegerArray

この節では、IntegerArray クラスに関する参考情報を説明しています。

IntegerArray コンストラクタ

IntegerArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
IntegerArray()
```

構文 2: int 配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
IntegerArray(int[] IntVal)
```

getInt

属性の整数配列を返します。

構文 : メソッドの署名を次に示します。

```
int[] getInt()
```

属性に整数配列を設定します。

構文 : メソッドの署名を次に示します。

```
void setInt(int[] IntVal)
```

23.2.12 StringArray

StringArray クラスは、String オブジェクトの配列のコンテナです。query() および getAttributes() メソッドを呼び出す場合、StringArray オブジェクトに値を取得する属性を指定して渡します。

この節では、StringArray クラスに関する参照情報を説明しています。

StringArray コンストラクタ

StringArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
StringArray()
```

構文 2: String 配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
StringArray(java.lang.String[] StringVal)
```

getString

StringArray オブジェクトに関連する String オブジェクトの配列を返します。

構文 : メソッドの署名を次に示します。

```
java.lang.String[] getString()
```

setString

StringArray オブジェクトに関連する String オブジェクトの配列を設定します。

構文 : メソッドの署名を次に示します。

```
void setString(java.lang.String[] StringVal)
```

23.2.13 StringEntry

StringEntry クラスは、StringEntryArray クラスにより保管されます。

この節では、StringEntry クラスに関する参考情報を説明しています。

StringEntry コンストラクタ

StringEntry クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
StringEntry()
```

構文 2: キーと String の値をパラメータとして取得するコンストラクタの構文を次に示します。

```
StringEntry(java.lang.String KeyVal, java.lang.String ValuesVal)
```

getKey

StringEntry オブジェクトに定義されているキーを返します。

構文 : メソッドの署名を次に示します。

```
java.lang.String getKey()
```

setKey

StringEntry オブジェクトにキーを設定します。

構文 : メソッドの署名を次に示します。

```
void setKey(java.lang.String KeyVal)
```

23.2.14 StringEntryArray

StringEntryArray クラスは、StringEntry オブジェクトの配列のコンテナです。これは、StringMap オブジェクトにより保管されます。

この節では、StringEntryArray クラスに関する参考情報を説明しています。

StringEntryArray コンストラクタ

StringEntryArray クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
StringEntryArray()
```

構文 2: StringEntry 配列をパラメータとして取得するコンストラクタの構文を次に示します。

```
StringEntryArray(StringEntry[] StringentryVal)
```

getStringentry

StringEntryArray オブジェクトのキーを返します。

構文 : メソッドの署名を次に示します。

```
StringEntry[] getStringentry()
```

setStringentry

StringEntryArray オブジェクトのキーを設定します。

構文 : メソッドの署名を次に示します。

```
void setStringentry(StringEntry[] StringentryVal)
```

23.2.15 StringMap

StringMap は、StringEntryArray オブジェクトのコンテナです。

この節では、StringMap クラスに関する参考情報を説明しています。

StringMap コンストラクタ

StringMap クラスには、2 つのコンストラクタがあります。

構文 1: パラメータを取得しないコンストラクタの構文を次に示します。

```
StringMap()
```

構文 2: StringEntryArray をパラメータとして取得するコンストラクタの構文を次に示します。

```
StringMap(StringEntryArray EntriesVal)
```

getEntries

この StringMap オブジェクトに保管されている StringEntryArray オブジェクトを返します。

構文 : メソッドの署名を次に示します。

```
StringEntryArray getEntries()
```

setEntries

この StringMap オブジェクトの StringEntryArray オブジェクトを設定します。

構文 : メソッドの署名を次に示します。

```
void setEntries(StringEntryArray EntriesVal)
```

23.2.16 VdxService

この節では、VdxService インタフェースに関する参考情報を説明しています。

getIRemoteVdxPort

リモートサービスのスタブを取得します。このスタブは、タイプ IRemoteVdx のポートです。

構文 : メソッドの署名を次に示します。

```
IRemoteVdx getIRemoteVdxPort() throws javax.xml.rpc.ServiceException;
```

23.2.17 VersionVO

この節では、VersionVO クラスに関する参照情報を説明しています。

getValue

サービスのバージョン番号を返します。

構文 : メソッドの署名を次に示します。

```
java.lang.String getValue()
```

23.3 VDX の例

次のコード例は、VDX サービスを使って、ディレクトリ抽象化層に定義されているエンティティに関連する属性にアクセスする方法を表しています。この例では、任意の検索、および事前定義されているグローバルクエリを実行しています。このコードには、サービスの getAttribute()、getAttributes()、query()、および globalQuery() メソッドの使用例が含まれています。

VDX サービスの SOAP エンドポイントへの参照を取得するために、getVdxStub() メソッドが呼び出されます。このメソッドの実装は、リストの最後に記載されています。

```
import java.util.Properties;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.xml.rpc.Stub;
import java.rmi.RemoteException;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.rmi.RemoteException;
import java.util.Calendar;
import java.util.Date;
import java.util.Hashtable;
import java.util.Map;
//
// Vdx imports
import com.novell.ws.client.vdx.IRemoteVdx;
import com.novell.ws.client.vdx.VdxService;
import com.novell.ws.client.vdx.VdxServiceException;
import com.novell.ws.client.vdx.VersionVO;
import com.novell.ws.client.vdx.Attribute;
import com.novell.ws.client.vdx.AttributeArray;
import com.novell.ws.client.vdx.AttributeType;
import com.novell.ws.client.vdx.ByteArrayArray;
import com.novell.ws.client.vdx.BooleanArray;
import com.novell.ws.client.vdx.DateArray;
import com.novell.ws.client.vdx.StringArray;
import com.novell.ws.client.vdx.IntegerArray;
import com.novell.ws.client.vdx.EntryArray;
import com.novell.ws.client.vdx.Entry;
import com.novell.ws.client.vdx.EntityAttributeMap;


public class ServiceTest
{
      public static final int VDX       = 0;
      public static final int NOTIFICATION      = 1;
      public static final int RESOURCE       = 2;
      public static final int ENDPOINT_SERVICE  = VDX;

      private static final int LOCALHOST  = 0;  // localhost
      private static final int TESTSERVER  = 1;  // testserver
```

```
    private static final int SELECTED_URL     = TESTSERVER;

    private String [] SERVER_URLS = {
          "http://localhost:8080/IDMProv/vdx/service",
          "http://testserver:8080/IDMProv/vdx/service"
    };

    private String url = SERVER_URLS[SELECTED_URL];
    private String username = "cn=admin,ou=idmsample,o=novell";
    private String password = "test";

    private String [] userAttributes = {
                  //"passwordAllowChange", // boolean
                  "UserPhoto",             // binary
                  //"loginTime",             // time
                  "Department",            // string
                  "Title",
                  "Email",
                  "manager",               // dn = string
                  "TelephoneNumber",
                  "directReports",
                  "FirstName",
                  //"surname",
                  "group",
                  "srvprvHideAttributes",
                  "NotificationPrefs",
                  "srvprvQueryList",
                  "Location",
                  };

    public ServiceTest() { };

    public static void main(String [] args)
    {
         ServiceTest serviceTest = new ServiceTest();
         //
         // Set default if no params are given
         int wService = ENDPOINT_SERVICE;
         if(args.length == 1)
            wService = Integer.parseInt(args[0]);

         try
         {
             serviceTest.run(wService);
         }
         catch(Exception e)
         {
             System.exit(-1);
         }
    }


    private void waitHere(long _time) { try { Thread.sleep(_time *
1000); } catch(InterruptedException ie) {} }
```

```
      public void run(int _service)
      throws Exception
      {
         if(_service == VDX)
         {
            System.out.println("Calling VDX endpoint");
            //
            // Get the version number
            getVersionTestCase();
            waitHere(2);
            //
            // Get attribute data for entity user
            getAttributeTestCase();
            waitHere(2);
            //
            // Get attributes
            getAttributesTestCase();
            waitHere(2);
            //
            // Query attributes
            queryAttributesTestCase();
            waitHere(2);
            //
            // Global query
           // Global query MUST be associated with a defined and
deployed query.
            // This can be done via the Designer.

            globalQueryTestCase();
         }
         else if(_service == NOTIFICATION)
         {
            System.out.println("Calling Notification endpoint");
            NotificationTest notificationTest = new
NotificationTest();
            //
            // Email Notification
            notificationTest.emailNotificationTestCase();
         }
         else if(_service == RESOURCE)
         {
            System.out.println("Calling Resource endpoint");
         }
         else
         {
            System.out.println("Unrecognized service selection");
         }
      }


public void globalQueryTestCase()
    throws Exception
```

```
    {

System.out.println("\n<=========queryAttributesTestCase=========>");
        try
        {
            //
            // Get the vdx stub
            IRemoteVdx vdxStub = getVdxStub(url, username, password);
            //
            // Create entry items corresponding to param key in DAL
            StringEntry [] entry = {
                    new StringEntry("titleattribute", "Chief Operating
Officer"),
                    new StringEntry("managerattribute",
"cn=jmiller,ou=users,ou=idmsample-pproto,o=novell")
            };
            //
            // Create and set the array of entries (key,value pairs)
            StringEntryArray entryArr = new StringEntryArray();
            entryArr.setStringentry(entry);
            //
            // Create and set the map using the entries
            StringMap map = new StringMap();
            map.setEntries(entryArr);
            //
            // Define and execute the global query
            int QUERY_KEY_INDEX = 0;
            String [] queryKeyName = {"TestVdxGlobalQuery2",
"TestVdxGlobalQuery"};
            //
            // Results from global query TestVdxGlobalQuery2 ----->
cn=apalani,ou=users,OU=idmsample-pproto,O=novell
            //
            // Make the vdx endpoint call
            StringArray array =
vdxStub.globalQuery(queryKeyName[QUERY_KEY_INDEX], map);
            String [] str = array.getString();
            if(str == null)
                throw new Exception("Global query returns null for key
name " + queryKeyName);
            else
            {
                System.out.println("Results for global query : " +
queryKeyName[QUERY_KEY_INDEX]);

System.out.println("================================================
===");
                for(int index = 0; index < str.length; index++)
                {
                    System.out.println(str[index]);
                }
            }
        }
        catch(VdxServiceException error)
```

```
        {
            System.out.println(error.getReason() );
            throw new Exception(error.getReason() );
        }
        catch(RemoteException error)
        {
            System.out.println(error.getMessage() );
            throw new Exception(error.getMessage() );
        }
    }


    public void queryAttributesTestCase()
    throws Exception
    {
        System.out.println("\nCalling queryAttributesTestCase() test
case");
        try
        {
            IRemoteVdx vdxStub = getVdxStub(url, username, password);

            StringArray attributes = new StringArray();
            attributes.setString(new String[]{"FirstName", "Title",
"UserPhoto", "Department"});
            String expression1 = "FirstName STARTWITH 'J'";
            String expression2 = "Title = 'Controller'";
            String expression3 = "vdxInteger > 0";
           String expression4 = "TelephoneNumber != '(555) 555-1201'";
            //
            // Test Cases
            // expression1 --> Should yield all users whose firstname
starts with J
           // expression1 AND expression2 --> Should yield jkelley who
is the Controller
          // expression1 AND expression3 --> Should yield only jmiller
            // expression1 AND expression4 --> Should yield all users
starting with J EXCEPT jmiller
            String finalExpression = expression1 + " AND " +
expression2;
            //
            // Make the vdx endpoint call
            EntityAttributeMap map = vdxStub.query("user", attributes,
finalExpression);
            EntryArray entryArray = map.getEntries();
            Entry [] entries = entryArray.getEntry();
            if(entries != null)
            {
                for(int index = 0; index < entries.length; index++)
                {
                    String dnKey = entries[index].getKey();
                    System.out.println("DN Key = " + dnKey);
                    AttributeArray attributeArray =
entries[index].getValues();
                    Attribute [] attributeData =
```

```
attributeArray.getAttribute();
                    for(int attrIndex = 0; attrIndex <
attributeData.length; attrIndex++)
                    {
                        //
                        // Determine how to handle the return data
                        examineAttributeData(attributeData[attrIndex],
" ");
                    }

                }
            }
        }
        catch(VdxServiceException error)
        {
            System.out.println(error.getReason() );
            throw new Exception(error.getReason() );
        }
        catch(RemoteException error)
        {
            System.out.println(error.getMessage() );
            throw new Exception(error.getMessage() );
        }
    }


    public void getVersionTestCase()
    throws Exception
    {
        System.out.println("\nCalling getVersionTestCase() test
case");

      try
      {
            IRemoteVdx vdxStub = getVdxStub(url, username, password);
            VersionVO version = vdxStub.getVersion();
            System.out.println("Version : " + version.getValue() );
      }
        catch(RemoteException error)
        {
                System.out.println(error.getMessage() );
                throw new Exception(error.getMessage() );
        }
    }


    public void getAttributeTestCase()
    throws Exception
    {
          System.out.println("\nCalling getAttributeTestCase() test
case");

          try
          {
```

```
            IRemoteVdx vdxStub = getVdxStub(url, username, password);

            String recipient =
"cn=jmiller,ou=users,ou=idmsample,o=novell";
            String entity = "user";
            for(int attributeIndex = 0; attributeIndex <
userAttributes.length; attributeIndex++)
            {
                //
                // Now, get the values for each attribute from the VDX
layer
                Attribute attributeData =
vdxStub.getAttribute(recipient,
                            entity, userAttributes[attributeIndex]);
                //
                // Determine how to handle the return data
                examineAttributeData(attributeData,
userAttributes[attributeIndex]);
            }
         }
        catch(VdxServiceException error)
        {
            System.out.println(error.getReason() );
            throw new Exception(error.getReason() );
        }
        catch(RemoteException error)
        {
            System.out.println(error.getMessage() );
            throw new Exception(error.getMessage() );
        }
   }

    public void getAttributesTestCase()
    throws Exception
    {
          System.out.println("\nCalling getAttributesTestCase() test
case");

          try
          {
            IRemoteVdx vdxStub = getVdxStub(url, username, password);

            String recipient =
"cn=jmiller,ou=users,ou=idmsample,o=novell";
            String entity = "user";
            StringArray userAttributesArray = new
StringArray(userAttributes);
            AttributeArray attributeArray =
vdxStub.getAttributes(recipient,
                  entity, userAttributesArray);
           Attribute [] attributeData = attributeArray.getAttribute();
            for(int index = 0; index < attributeData.length; index++)
            {
                //
```

```
                // Determine how to handle the return data
                examineAttributeData(attributeData[index],
userAttributes[index]);
            }
        }
        catch(VdxServiceException error)
        {
            System.out.println(error.getReason() );
            throw new Exception(error.getReason() );
        }
        catch(RemoteException error)
        {
            System.out.println(error.getMessage() );
            throw new Exception(error.getMessage() );
        }
   }

    private void examineAttributeData(Attribute _attribute, String
_attributeName)
    throws Exception
    {
        AttributeType type = _attribute.getType();
       System.out.println("Attribute type : " + type);
        //
        // What type are we dealing with?
        if(type.getValue().compareTo(AttributeType._Integer) == 0)
        {
            IntegerArray intArray = _attribute.getIntegers();
            int [] intData = intArray.getInt();
            if(intData == null)
                System.out.println(_attributeName + " attribute : " +
"null because no attribute value exists.");
            else
            {
                for(int intIndex = 0; intIndex < intData.length;
intIndex++)
                {
                   System.out.println(_attributeName + " attribute : "
+ intData[intIndex]);
                }
            }
        }
       else if(type.getValue().compareTo(AttributeType._Boolean) == 0)
        {
            BooleanArray boolArray = _attribute.getBooleans();
            boolean [] booleanData = boolArray.getBoolean();
            if(booleanData == null)
                System.out.println(_attributeName + " attribute : " +
"null because no attribute value exists.");
            else
            {
                for(int boolIndex = 0; boolIndex < booleanData.length;
boolIndex++)
                {
```

```
                    System.out.println(_attributeName + " attribute : "
+ booleanData[boolIndex]);
                }
            }
        }
        else if( (type.getValue().compareTo(AttributeType._String) ==
0) ||
                 (type.getValue().compareTo(AttributeType._DN) == 0) )
        {
            StringArray dataArray = _attribute.getStrings();
            String [] stringData = dataArray.getString();
            if(stringData == null)
                System.out.println(_attributeName + " attribute : " +
"null because no attribute value exists.");
            else
            {
                for(int strIndex = 0; strIndex < stringData.length;
strIndex++)
                {
                   System.out.println(_attributeName + " attribute : "
+ stringData[strIndex]);
                }
            }
        }
        else if(type.getValue().compareTo(AttributeType._Binary) == 0)
        {
            ByteArrayArray byteArray = _attribute.getBinaries();
            byte [][] byteData = byteArray.getBase64Binary();
            if(byteData == null)
                System.out.println(_attributeName + " attribute : " +
"null because no attribute value exists.");
            else
            {
                for(int byteIndex = 0; byteIndex < byteData.length;
byteIndex++)
                {
                    byte [] data = byteData[byteIndex];
                    //
                    // Save the data to a gif file and view it to
                    // make sure the binary return data is correct.
                    try
                    {
                       File fileObj = new File("C:\\temp\\photo.gif");
                        if(fileObj.exists())
                            fileObj.delete();
                        FileOutputStream fout = new
FileOutputStream(fileObj);
                        fout.write(data);
                        fout.flush();
                    }
                    catch(FileNotFoundException fne)
                    {
                        throw new Exception(fne.getMessage());
                    }
```

```
                    catch(IOException ioe)
                    {
                        throw new Exception(ioe.getMessage());
                    }
                }
            }
        }
        else if(type.getValue().compareTo(AttributeType._Time) == 0)
        {
            DateArray dateArray = _attribute.getDates();
            Calendar [] calendar = dateArray.getDatetime();
            if(calendar == null)
                System.out.println(_attributeName + " attribute : " +
"null because no attribute value exists.");
            else
            {
                for(int calIndex = 0; calIndex < calendar.length;
calIndex++)
                {
                   System.out.println(_attributeName + " attribute : "
+ calendar[calIndex].getTime().toString());
                }
            }
        }

    }


    /**
     * Method to obtain the remote interface to the Vdx endpoint
     * @param _url
     * @param _username
     * @param _password
     * @return IRemoteMetrics interface
     * @throws Exception
     */
   private IRemoteVdx getVdxStub(String _url, String _username, String
_password)
    throws Exception
    {
        Properties properties = new Properties();
        properties.put(Context.INITIAL_CONTEXT_FACTORY,
"org.jnp.interfaces.NamingContextFactory");

        String lookup =
"xmlrpc:soap:com.novell.ws.client.vdx.VdxService";

        InitialContext ctx = new InitialContext();
        VdxService svc = (VdxService) ctx.lookup(lookup);

        Stub stub = (Stub)svc.getIRemoteVdxPort();

        stub._setProperty(Stub.USERNAME_PROPERTY, _username);
        stub._setProperty(Stub.PASSWORD_PROPERTY, _password);
```

```
        stub._setProperty(Stub.SESSION_MAINTAIN_PROPERTY,
Boolean.TRUE);
        stub._setProperty(Stub.ENDPOINT_ADDRESS_PROPERTY, _url);

        return (IRemoteVdx) stub;
    }

}
```

付録

VII

次の節には、Identity Manager ユーザアプリケーションに関する参照情報と高度なトピックが記載されています。

ユーザアプリケーションのスキーマ拡張

A

この節では、ユーザアプリケーションが使用するスキーマ拡張について説明します。次の節から構成されています。

A.1 属性のスキーマ拡張

属性名	説明
srvprvAllowMgrInitiate	マネージャがプロビジョニング要求を開始できるかどうかを示すフラグです。
srvprvAllowMgrRetract	マネージャがプロビジョニング要求を撤回できるかどうかを示すフラグです。
srvprvAllowMgrSetAvailability	マネージャがチームの代理人を設定できるかどうかを示すフラグです。
srvprvAllowMgrSetDelegate	マネージャがプロビジョニング要求の委任を設定できるかどうかを示すフラグです。
srvprvAllowMgrSetProxy	マネージャがチーム代理人を設定できるかどうかを示すフラグです。
srvprvAllowMgrTaskClaim	マネージャがプロビジョニング承認タスクを引き受けできるかどうかを示すフラグです。
srvprvAllowMgrTaskReassign	マネージャがプロビジョニング承認タスクを再割り当てできるかどうかを示すフラグです。
srvprvAllRequests	割り当てが、チームのすべてのプロビジョニング要求定義をカバーしているかどうかを示すフラグです。
srvprvAOLIMAddress	AOL IM アドレス。
srvprvAssetRef	srvprvAssetRecipientAux クラスによってユーザに関連付けられている名前付き資産に対する集約資産プロパティを表したものです。
srvprvAssignExpiration	代理人または委任ユーザの割り当てが期限切れになる時間です。
srvprvAssignFromContainer	代理人または委任ユーザの割り当てのコンテナサブジェクトです。
srvprvAssignFromGroup	代理人または委任の割り当てのグループサブジェクトです。
srvprvAssignFromUser	代理人または委任ユーザの割り当てのユーザサブジェクトです。

属性名	説明
srvprvAssignStartTime	委任割り当てが有効になる時刻です。
srvprvAssignToRelationship	委任割り当てのターゲット関係です。
srvprvAssignToUser	代理人または委任ユーザの割り当てのユーザターゲットです。
srvprvAutoDisplayTeam	自動的にチームメンバーを表示します。
srvprvCapabilities1-5	ユーザのスキルを表します。
srvprvCategoryKey	特定のプロビジョニング要求定義をプロビジョニングカテゴリのセットに関連付けます。値は srvprvChoice インスタンスのキーです。'
srvprvCurrentDelegatees	ユーザに関連付けられた委任です。
srvprvCurrentDelegators	ユーザに関連付けられた委任です。
srvprvDefaultTheme	デフォルトのテーマです。
srvprvDelegateeDef	委任先定義 DN です。
srvprvDelegationDef	委任定義 DN です。
srvprvDelegators	この割り当てで委任者として定義されたユーザです。
srvprvEntitlementRef	DirXML エンタイトルメントへの参照です。
srvprvEntityType	ディレクトリ抽象化層エンティティの定義タイプを指定します。
srvprvFlowStrategy	プロビジョニング要求定義に対して使用するフロー起動方法を指定します。
srvprvGrant	true の場合、プロジェクトリクエスト定義が付与処理をサポートするよう指定するフラグです。
srvprvGroupwiseIMAddress	Groupwise IM アドレスです。
srvprvHideAttributes	特定の属性を非表示にするかどうかを示すフラグです。
srvprvHideUser	リスト検索クエリの実行時に、ユーザを非表示にするかどうかを示すフラグです。
srvprvIMAddress	インスタントメッセンジャーのアドレスです。
srvprvIsTaskManager	ユーザがタスクグループマネージャかどうかを示します。
srvprvLocalizedDescrs	プロビジョニング Web アプリケーション、Designer、および iManager に対し、ローカライズされた説明文字列のセットを提供します。
srvprvLocalizedNames	プロビジョニング Web アプリケーション、Designer、および iManager に対し、ローカライズされた表示名文字列のセットを提供します。
srvprvManager	マネージャであるユーザを示します。
srvprvManagerGroup	マネージャがあるグループを示します。
srvprvManagerNotMember	マネージャがチームのメンバーではないことを示します。
srvprvMember	ユーザがチームのメンバーであることを示します。

属性名	説明
srvprvMemberContainer	チームメンバーを含むコンテナの名前です。
srvprvMemberGroup	チームメンバーを含むグループの名前です。
srvprvMemberRelationship	マネージャオブジェクト内でメンバーベースの属性を判断する、ディレクトリ抽象化層リレーションシップ名です。
srvprvModified	ディレクトリモデルコンテナ内の定義オブジェクトインスタンスの変更を示すフラグです。
srvprvNotificationPrefs	ユーザが受け取る通知タイプセットを定義します。
srvprvPreferredLocale	ユーザの優先ロケールです。
srvprvProcessXML	ワークフローおよびプロビジョニングアクションを含むプロビジョニングプロセス定義を表す XML ドキュメント。
srvprvQueryList	保存されているクエリや検索条件のリストです。
srvprvRelationship	アイデンティティボールト内のオブジェクト間のリレーションシップを定義します。
srvprvRequest	許可または拒否する項目を 1 つ公開します。ワークフローおよびプロビジョニングターゲットのランタイムの側面を定義するワークフロープロセスが含まれます。
srvprvRequestDefName	委任定義に関連付けられているプロビジョニング要求定義名。
srvprvRequestScope	プロビジョニング要求のスコープです。
srvprvRequestXML	初期の要求フォームとそのデータバインドを表す XML ドキュメントです。
srvprvRevoke	True の場合、プロビジョニング要求定義が取り消し操作をサポートすることを示します。
srvprvStatus	プロビジョニングオブジェクトがサポートする値のステータスを指定します。
srvprvTaskGroups	ユーザがタスクマネージャであるグループです。
srvprvTaskManager	タスクグループのタスクマネージャです。
srvprvTaskScopeAddressee	宛先のタスクスコープです。
srvprvTaskScopeRecipient	受信者のタスクスコープです。
srvprvTeam	チーム定義のコンテナです。
srvprvUser	委任割り当てに関連するユーザです。
srvprvUUID	ポートレットの固有の識別子です。
srvprvYahooIMAddress	Yahoo* IM アドレスです。

A.2 Objectclass のスキーマ拡張

Objectclass 名	説明
srvprvAppConfig	DirXML ドライバの親が接続するプロビジョニングシステムのアプリケーション設定オブジェクトのコンテナです。
srvprvAppDefs	プロビジョニングランタイム環境 (Identity ポータルのテーマなど) を初期化するために使用される設定オブジェクトのコンテナです。
srvprvAssetRecipientAux	ユーザに対する非 IT 資産のプロビジョニングを記録します。
srvprvChoice	Identity ポートレットおよび他の Web アプリケーションコンポーネントで使用できるように、特定の属性に割り当てたり、クエリで使用したりできる値の列挙です。
srvprvChoiceDefs	Identity ポートレットおよび Web アプリケーションによって公開されるディレクトリ抽象化層選択肢の定義のコンテナです。
srvprvDelegateeAssignment	割り当て定義を委任します。
srvprvDelegateeDefs	委任定義のコンテナです。
srvprvDelegationAssignment	委任または可用性割り当て定義です。
srvprvDelegationDefs	委任および委任者定義のコンテナです。
srvprvDelegatorAssignment	委任または可用性割り当て定義です。
srvprvDirectoryModel	ディレクトリ抽象化層のメタレベルオブジェクトのコンテナで、Identity ポートレットおよび Web アプリケーションによって公開されるディレクトリの選択内容です。
srvprvDirectoryModelConfig	ランタイムディレクトリ抽象化層の環境設定パラメータです。
srvprvEntity	ディレクトリ内にある定義済みクラスの選択属性のビューを定義します。Identity ポートレットおよび他の Web アプリケーションコンポーネントによって使用されます。
srvprvEntityAux	標準の ObjectClass です。
srvprvEntityDefs	Identity ポートレットおよび Web アプリケーションによって公開されるディレクトリ抽象化層エンティティの定義のコンテナです。
srvprvProxyAssignment	代理人割り当て定義です。
srvprvProxyDefs	代理人定義のコンテナです。
srvprvQuery	ディレクトリ抽象化層クエリ定義です。
srvprvQueryDefs	ディレクトリ抽象化層クエリ定義のコンテナです。
srvprvRelationship	Identity ポートレットおよび他の Web アプリケーションコンポーネントで使用できるように、ディレクトリ内のオブジェクトの関係を定義します。
srvprvRelationshipDefs	Identity ポートレットおよび Web アプリケーションによって公開されるディレクトリ抽象化層関係の定義のコンテナです。

Objectclass 名	説明
srvprvRequest	許可または拒否する項目を 1 つ公開します。ワークフローおよびプロビジョニングターゲットのランタイムの側面を定義するワークフロープロセスが含まれます。
srvprvRequestDefs	Web アプリケーションランタイムに対して項目のセットであるプロビジョニング要求定義のコンテナ。
srvprvResource	プロビジョニング実行操作のために、実行するディレクトリ割り当てのセットを定義します (許可または拒否)。
srvprvResourceDefs	プロビジョニングターゲット定義のコンテナで、デザインタイムの記述のほかにテンプレートや未使用ターゲットも含まれます。
srvprvService	特定のワークフローから Web サービスを起動する方法を記述します。これには、入力値および戻り値の指定が含まれます。
srvprvServiceDefs	サービス定義オブジェクトのコンテナで、ワークフローによって呼び出される Web サービスをラップします。
srvprvTaskGroupAux	サービスプロビジョニングのタスクグループです。
srvprvTeam	プロビジョニング要求管理用チームです。
srvprvTeamDefs	チーム定義のコンテナです。
srvprvTeamRequest	チームプロビジョニング要求です。
srvprvTheme	テーマオブジェクトです。
srvprvUserAux	サービスプロビジョニングのユーザエンティティです。
srvprvWebAppConfig	Web アプリケーション環境設定オブジェクトです。
srvprvWorkflow	プロビジョニングアクションの許可を得るために実行される移動条件を含む動作のネットワークを定義します。
srvprvWorkflowDefs	: ワークフローオブジェクトのコンテナで、デザインタイムの説明に加えてテンプレートや未使用のフローが含まれます。

JavaScript 検索 API

B

Identity Manager ユーザアプリケーションのフレームワークは、ディレクトリ抽象化層にアクセスして検索を実行する JavaScript API をサポートしています。この API を使って、ユーザアプリケーション外で動作する JSP ページから、クエリを作成、保存、実行することができます。クエリを実行するために、SearchListPortlet のサービスを起動して、検索条件と書式設定オプションを指定したパラメータを渡すことができます。SearchListPortlet を使わず、直接 API を使って検索を実行することもできます。

このドキュメントでは次のトピックについて説明します。

B.1 SearchListPortlet を使った基本検索の実行

基本検索を実行するために、JSP ページから SearchListPortlet へのディープリンクを指定することができます。ポートレットの URL は、検索条件を指定した要求パラメータの単純セット、または JSON 形式のクエリ文字列を渡す必要があります。基本検索には、次のような単一の検索条件を定義します。

```
First Name starts with A
```

検索を実行するには、SearchListPortlet のシングルポートレットレンダ URL を呼び出します。要求パラメータ MODE=MODE_RESULTS_LIST を渡す必要があります。

B.1.1 要求パラメータを渡す

SearchListPortlet に要求パラメータの単純セットを渡すことができます。これらのパラメータは、エンティティ、検索する属性、演算子、および検索文字列を指定します。ポートレットの URL および 4 つの要求パラメータを指定したスクリプトの例を次に示します。

```
<script type="text/javascript">function
openSearchResults(extraUrlParams) {
  var url = "/IDMProv/portal/portlet/SearchListPortlet?";
  url += "urlType=Render&novl-regid=SearchListPortlet";
  url += "&novl-inst=IDMProv.SearchListPortlet";
  url += "&wsrp-mode=view&wsrp-windowstate=normal";
  url += "&MODE=MODE_RESULTS_LIST&";
  url += extraUrlParams;
  var feat = "width=700,height=600";
  feat += ",menubar=no,resizable=yes,toolbar=no,scrollbars=yes";
  var win = window.open(url, "TestSearchPopup", feat);
```

```
  if (win) win.focus();
}

var search1a = "ENTITY_DEF=user";
search1a += "&COND_ROW_ATTR=FirstName";
search1a += "&COND_ROW_REL_OP=starts-with";
search1a += "&COND_ROW_VAL=A";
...
```

この関数を呼び出すために、onclick イベントを使って次のようにボタンを表示できます。

```
<input type="button" value="GO" onclick="openSearchResults(search1a)"/
>
```

要求パラメータの説明を次の表に示します。

表 B-1 *基本検索の要求パラメータ*

要求パラメータ	説明
ENTITY_DEF	ディレクトリ抽象化層のエンティティを指定します。
COND_ROW_ATTR	検索する属性を指定します。
COND_ROW_REL_OP	検索式で使用する演算子を指定します。属性タイプ string、boolean、integer、time, dn_lookup、dynamic_list、および static_list に対して、次の演算子がサポートされています。 equals 、 present、 not_equals、 not_present 属性タイプ string に対して、次の演算子がサポートされています。 starts_with、 ends_with、 contains、 not_starts_with、 not_ends_with、 not_contains 属性タイプ integer および time に対して、次の演算子がサポートされています。 greater、 greater_or_equal、 less、 less_or_equal、 not_greater、 not_greater_or_equal、 not_less、 not_less_or_equal

要求パラメータ	説明
COND_ROW_VAL	検索する値です。

B.1.2 JSON 形式の文字列を使ったクエリ

クエリに JSON 文字列形式を使用する場合、SearchListPortlet には前述の要求パラメータの代わりに、QUERY パラメータを渡します。QUERY パラメータの構成を示す JavaScript 変数を次に示します。

```
var search1b ='QUERY={"k":"Lastname starts with B","mxPg":"10",';
search1b +='"mxRes":"0","ptr":"1","grp":[{"map":{"row":[{"map":{';
search1b +='"rowRop":"starts-with","rowVal":"B","rowAttr":"LastName"';
search1b +='}}],"rowLop":"and"}}],';
search1b +='"orderBy":"LastName","entDef":"user",';
search1b +='"sScope":"","sRoot":"","grpLop":"and",';
search1b +='"selAttr":["FirstName","LastName",';
search1b +='"Title","Email","TelephoneNumber"]}';
```

JSON 構造では、SearchListPortlet に関連する大部分の設定／初期設定値を指定することができます。

SearchListPortlet に渡される QUERY パラメータを定義する、JSON 名／値のペアを次の表に示します。

***表 B-2** QUERY パラメータを定義する JSON 構造*

JSON 設定	説明
k	検索の名前を指定します。(オプション)
mxPg	ページ当たりの最大行数を指定します。(オプション)
mxRes	取得する行合計の最大値を指定します。(オプション)
ptr	ページ番号付けのオフセットを定義するスクロールポインタを設定します。(オプション)
grp	条件グループを定義します。1 つまたは複数の条件グループを定義することができます。条件グループの設定の詳細は、510 ページの 表 B-3 を参照してください。
orderBy	ソートする属性を指定します。(オプション)
entDef	ディレクトリ抽象化層のエンティティを指定します。
sScope	検索スコープを設定します。(オプション)
sRoot	検索ルートを設定します。(オプション)
grpLop	このクエリ内のグループに対する論理演算子 (and または or) を定義します。
selAttr	検索結果に含める属性を表示します。

条件グループを定義する JSON 構造を、次の表に示します。

表 B-3 *条件グループを定義する JSON 構造*

JSON 設定	説明
row	条件行を定義します。1 つまたは複数の条件行を定義することができます。条件行の設定の詳細は、510 ページの 表 B-4 を参照してください。
rowLop	このグループ内の行に対する論理演算子 (and または or) を定義します。

条件行を定義する JSON 構造を、次の表に示します。

表 B-4 *条件行のフィールドを定義する JSON 構造*

JSON 設定	説明
rowRop	関係演算子を定義します。JSON がサポートする関係演算子は、要求パラメータを使った基本検索の演算子と同じです。関係演算子の完全なリストについては、508 ページの 表 B-1 の COND_ROW_REL_OP の説明を参照してください。
rowVal	検索値を設定します。
rowAttr	検索する属性を指定します。

B.2 JavaScript API を使った新規クエリの作成

基本検索要求パラメータ、または JSON 構造を使用する代わりに、JavaScript API を呼び出してクエリを実行することができます。この節では、API を使用する際の簡単なテクニック、および API の参考ドキュメントについて説明します。

検索 API は、JUICE と言う名前のユーザアプリケーションコンポーネントに組み込まれた ajax フレームワークを使用します。JUICE(JavaScript UI Controls and Extensions) は、dojo ライブラリに準拠しており、このライブラリを使用します。JUICE は、ユーザアプリケーションで使用される dojo リリースに結合されます。

そのため、IDM ユーザアプリケーション WAR ファイル内のカスタムページで JUICE を使用するには、dojo.js(JUICE ではない) へのスクリプト参照が必要です。dojo.js への参照を追加したら、dojo に JUICE のダウンロードを指示する JavaScript 行を追加できます。

JavaScript API を使用する前に、dojo モジュールを利用できるようにするための設定作業を行う必要があります。

1 HTML ヘッダに dojo.js 用のスクリプトタグを追加します。dojo.js への参照は、次のようにヘッダ内 (本文ではない) になければなりません。

```
<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-
```

```
8">
<title>JavaScript Search</title>
<script type="text/javascript">
  if(typeof dojo=="undefined"){
    var djConfig={isDebug: false,
                  baseScriptUri: "/IDMProv/javascript/dojo/"};
    var buf="<script type='text\/javascript' ";
    buf+="src='/IDMProv/UIQuery?js=dojo\/dojo.js'><\/script>";
    document.writeln(buf);
  }
</script>
</head>
```

2 JUICE をブラウザのメモリに読み込むには、この JavaScript ステートメントを追加します。

```
<script type="text/javascript">
  //This line must precede any code using JUICE.
  dojo.require("JUICE.*");
</script>
```

3 エンティティ検索を含む JUICE.IDM サービスを活用するために、この JavaScript ステートメントも追加されます。

```
<script type="text/javascript">
  //This line must precede any code using JUICE.IDM services.
  dojo.require("JUICE.IDM.*");
</script>
```

クエリを作成するには、JUICE.IDM.Entities.Search オブジェクトで create() メソッドを呼び出し、クエリに与える名前を渡す必要があります。create() メソッドは静的メソッドです。起動方法を次に示します。

```
var newQuery = JUICE.IDM.Entities.Search.create("My New Search");
```

クエリオブジェクトを作成したら、このオブジェクトでメソッドを呼び出し、クエリの基本設定を定義したり、条件グループや条件行を定義することができます。JavaScript API を使って作成するクエリ構造は、JSON 表記モデルに従っています。クエリオブジェクトを作成したら、それを QUERY 要求パラメータに追加します。

次の JavaScript の例は、JavaScript API を使ったクエリの作成方法を表しています。

```
function buildQuery3() {
  var newQuery = JUICE.IDM.Entities.Search.create("My New Search");
  newQuery.setFrom("user");
  var selAttrs = ["FirstName","LastName"];
  newQuery.setSelects(selAttrs);
  var newCondGrp1 = newQuery.addConditionGroup();
  var newCondRow1_1 = newCondGrp1.addConditionRow();
  newCondRow1_1.setRowAttr("FirstName");
  newCondRow1_1.setRowRop("contains");
  newCondRow1_1.setRowVal("C");
  openSearchResults("QUERY=" + newQuery);
}
```

B.2.1 JavaScript API

この節では、JavaScript API を使ったディレクトリ抽象化層内のエンティティの検索方法に関する参考資料を取り上げています。

JUICE.IDM.Entities.Search オブジェクトの静的メソッドを次の表に示します。

表 B-5 *JUICE.IDM.Entities.Search の静的メソッド*

方法	説明
<Query> create(searchName)	searchName の名前で新しいクエリを作成します。
<void> load(uuid)	ユーザが保存した指定した uuid の検索をロードします。
<Query> get(uuid)	ユーザが保存した指定した uuid の検索をクエリとして返します。
<String[]> getNames()	ログインしたユーザが保存したすべての検索の名前を返します。
<String> getUUID(searchName)	searchName の名前を持つ保存済み検索の uuid を返します。

Query オブジェクトのメソッドを次の表に示します。

表 B-6 *Query オブジェクトのメソッド*

方法	説明
<void> setKey(searchName)	searchName を設定します。
<void> setFrom(defKey)	エンティティ定義から設定します。
<void> setSelects(attrKey[])	選択項目を設定します (オプション、SearchListPortlet を使用する場合)。
<void> setSearchScope(scp)	検索スコープを設定します (オプション)。
<void> setSearchRoot(rt)	検索ルートを設定します (オプション)。
<void> setMaxPage(int)	ページ当たりの最大行数を設定します (オプション)。
<void> setMaxResults(int)	合計最大行数を設定します (オプション)。
<void> setOrderBy(attrKey)	ソートを設定します (オプション)。
<void> setPointer(int)	ページ番号付けのオフセットを設定します (オプション)。
<void> setGroupLop(lop)	グループ間論理演算子を設定します。
<String> getKey()	searchName を取得します。
<String> getFrom()	エンティティ定義から取得します。
<String> getSelects()	選択項目を取得します。

方法	説明
<String> getSearchScope()	検索スコープを取得します。
<String> getSearchRoot()	検索ルートを取得します。
<int> getMaxPage()	ページ当たりの最大行数を取得します。
<int> getMaxResults()	合計最大行数を取得します。
<String> getOrderBy()	ソートを取得します。
<int> getPointer()	ページ番号付けのオフセットを取得します。
<String> getGroupLop()	グループ間論理演算子を取得します。
<int> nbConditionGroups	条件グループ数を返します。
<CondGroup> addConditionGroup	クエリに追加された新しい条件グループ (CondGroup) を作成し、返します。
<void> removeConditonGroup(i)	i の条件グループを削除します。
<CondGroup> getConditonGroup(i)	i の条件グループを返します。

CondGroup オブジェクトのメソッドを次の表に示します。

表 B-7 *CondGroup オブジェクトのメソッド*

方法	説明
<void> setRowLop(lop)	グループ内論理演算子を設定します。
<String> getRowLop()	グループ内論理演算子を取得します。
<int> nbConditionRows()	条件行数を返します。
<CondRow> addConditionRow()	条件グループに追加された新しい条件行を作成し、返します。
<void> removeConditionRow(i)	i の条件行を削除します。
<CondRow> getConditionRow(i)	i の条件行を返します。

CondRow オブジェクトのメソッドを次の表に示します。

表 B-8 *CondRow オブジェクトのメソッド*

方法	説明
<void> setRowAttr(attrKey)	属性を設定します。
<void> setRowRop(rop)	関係演算子を設定します。
<void> setRowVal(val)	検索値を設定します。
<String> getRowAttr()	属性を取得します。
<String> getRowRop()	関係演算子を取得します。

方法	説明
<String> getRowVal()	検索値を取得します。

B.3 JSON 形式クエリによる高度な検索の実行

JSON と QUERY パラメータを使って、高度な検索を実行することができます。JSON 構文の規則は、基本検索と同じです。ただし、一般的に高度な検索では、複数の条件グループや条件行が定義されるという違いがあります。次の JavaScript 変数は、複数の条件グループと条件行を使った検索を行うための、QUERY パラメータの設定方法を表しています。

```
var search2 = 'QUERY={"k":"Complicated Search All
OK","mxPg":"10","mxRes":"0","ptr":"1","grp":[{"map":{"row":[{"map":{"r
owRop":"equals","rowVal":"cn=bg1,ou=groups,ou=idmsample,o=novell","row
Attr":"group"}},{"map":{"rowRop":"contains","rowVal":"0","rowAttr":"Fi
rstName"}}],"rowLop":"and"}},{"map":{"row":[{"map":{"rowRop":"not-
present","rowVal":"","rowAttr":"TelephoneNumber"}},{"map":{"rowRop":"e
quals","rowVal":"cn=ablake,ou=users,ou=idmsample,o=novell","rowAttr":"
directReports"}},{"map":{"rowRop":"equals","rowVal":"cn=cnano,ou=users
,ou=idmsample,o=novell","rowAttr":"manager"}}],"rowLop":"and"}},{"map"
:{"row":[{"map":{"rowRop":"not-
present","rowVal":"","rowAttr":"TelephoneNumber"}},{"map":{"rowRop":"e
quals","rowVal":"cn=ablake,ou=users,ou=idmsample,o=novell","rowAttr":"
directReports"}},{"map":{"rowRop":"equals","rowVal":"cn=cnano,ou=users
,ou=idmsample,o=novell","rowAttr":"manager"}}],"rowLop":"and"}}],"orde
rBy":"LastName","entDef":"user","sScope":"","sRoot":"","grpLop":"or","
selAttr":["FirstName","Title","Email","TelephoneNumber"]}';
```

JSON 設定の詳細は、509 ページのセクション B.1.2「JSON 形式の文字列を使ったクエリ」を参照してください。

B.4 現在のユーザのすべての保存済みクエリの取得

JavaScript API を使って、現在ログオンしているユーザのすべての保存済みクエリを取得することができます。そのためには、JUICE.IDM.Enities.Search オブジェクトで getNames() 静的メソッドを呼び出す必要があります。

次の JavaScript の例は、現在のユーザのすべての保存済みクエリを取得するプロシージャを表しています。

```
function query4GetSavedQueries() {
  var searchNames = JUICE.IDM.Entities.Search.getNames();
  var replaceDiv = document.getElementById("savedQueryNames");
  replaceDiv.innerHTML = searchNames;
}
```

B.5 既存の保存済みクエリの実行

JavaScript API を使って、保存済みクエリを実行することができます。保存済みクエリを実行する前に、次の JavaScript ステートメントを実行して保存済みクエリを取得してください (前の節で説明)。

```
JUICE.IDM.Entities.Search.getNames();
```

実行する保存済み検索名が分かっている場合でも、まず getNames() を先に呼び出す必要があります。

getNames() 関数を呼び出したら、次の手順で保存済み検索を実行します。

1. getUUID() メソッドを呼び出して、検索名に対応する UUID にアクセスします。
2. JUICE.IDM.Entities.Search オブジェクトで load() メソッドを呼び出して、UUID を持つ保存済みクエリをロードします。
3. get() メソッドを呼び出して、保存済みクエリ構造を取得します。

これらのメソッドはすべて静的メソッドです。

クエリ構造を取得したら、それを使って QUERY 要求パラメータを作成することができます。

次の JavaScript の例は、保存済みクエリの起動手順を表しています。

```
function runQuery4() {
  var textField = document.getElementById("savedQueryToRun");
  var queryName = textField.value;
  var queryUUID = JUICE.IDM.Entities.Search.getUUID(queryName);
  JUICE.IDM.Entities.Search.load(queryUUID);
  var myQuery = JUICE.IDM.Entities.Search.get(queryUUID);

  openSearchResults("QUERY=" + myQuery);
}
```

B.6 検索可能なすべての属性の検索

JavaScript API を使って、エンティティのすべての検索可能な属性を検索することができます。このタイプの検索は、string(文字列) タイプの属性にだけ適用されます。そのため、DN、date(日付)、integer(整数)、boolean(ブール) など、他のタイプでこのような検索を行うことはできません。

検索可能なすべての属性に対して検索を実行するには、まず他の検索方法と同じようにクエリオブジェクトを作成します (前述)。つぎに 、JUICE.IDM.Definition.load() を呼び出して、エンティティ定義の属性リストを取得する必要があります。属性リストを取得したら、各属性が文字列で検索可能であることを確認してください。検索可能な各文字列属性に対して、条件グループオブジェクトで addConditionRow() メソッドを呼び出して、条件行を追加します。すべての条件行を追加したら、検索を実行することができます。

次の JavaScript の例は、すべての検索可能属性の検索方法を表しています。

```
function buildQuery5() {
  var searchStr = document.getElementById("query5Text").value;
  if (searchStr == "") {
    alert("Enter a search string in the text field.");
    return;
  }
  var newQuery = JUICE.IDM.Entities.Search.create("My New Search");
  var entDef = "user";
  newQuery.setFrom(entDef);
  var selAttrs = new Array();
  selAttrs.push("FirstName");
  selAttrs.push("LastName");
```

```
  newQuery.setSelects(selAttrs);
  var newCondGrp1 = newQuery.addConditionGroup();
  newCondGrp1.setRowLop("or");

  //get all the searchable attributes of entity-definition user that
are type string (excludes DN, date, integer, boolean, etc)
  JUICE.IDM.Definitions.load(entDef);
  var attrKeys = JUICE.IDM.Definitions.getAttributeKeys(entDef);
  for (var i = 0; i < attrKeys.length; i++) {
    var attrDef = JUICE.IDM.Definitions.getAttribute(entDef,
attrKeys[i]);
    var attrType = attrDef.getType();
    var searchable = attrDef.isSearchable();

    if (attrType == "String" && searchable ) {
      var newCondRow = newCondGrp1.addConditionRow();
      newCondRow.setRowAttr(attrKeys[i]);
      newCondRow.setRowRop("contains");
      newCondRow.setRowVal(searchStr);
    }
  }
  openSearchResults("QUERY=" + newQuery);
}
```

トラブルシューティング

C

この節では、一般的なエラーへの対処方法を説明しています。これには、次の項目が含まれています。

C.1 PermGen スペースエラー

ユーザアプリケーションの再展開時に、次のエラーが発生することがあります。

```
11:32:20,194 ERROR [[PortalAggregator]] Servlet.service() for
servletPortalAggregator threw exception java.lang.OutOfMemoryError:
PermGen space
```

このエラーを回避するには、次のどちらかの作業を行います。

- JBoss サーバを再起動する .

または

- または、start-jboss スクリプトの JAVA_OPTS を使って、JVM に -XX:MaxPermSize を渡して PermSpace の値を増やします。次に例を示します。

  ```
  -XX:MaxpermSize=128m
  ```

C.2 電子メール通知テンプレート

電子メール通知テンプレートがある言語でのみ表示され、ユーザが指定したデフォルトロケールで表示されない場合は、どの通知テンプレートを選択しているかを確認してください。デフォルトのテンプレート、またはローカライズ版のテンプレートを選択できます。ローカライズ版のテンプレートを選択した場合は、ユーザのデフォルト言語に関係なく、そのローカライズ版のテンプレートに対応する言語で表示されます。デフォルトのテンプレート (ロケールコードがないテンプレート) を選択した場合、電子メールはユーザのデフォルト言語で表示されます (サポートしているデフォルト言語の場合)。

C.3 組織図とゲストアクセス

ランタイム時にこのようなエラーが発生した場合、ユーザアプリケーション WAR のサービス定義を変更する必要があります。

```
error: "an error occurred Control instantiation of JUICE.OrgChartCtrl
failed (Object doesn't support this property or method). Please
contact your system administrator. Detailed information can be found
in the console." when accessing the portlet in a browser.
```

このメッセージの修正方法の詳細は、265 ページのセクション 13.3「ゲストアクセス用の組織図の環境設定」を参照してください。

C.4 プロビジョニング通知

次の［要求と承認］ページで、*［他のユーザにこれらの変更について通知します］*チェックボックスが表示されない場合 :

- 可用性の設定
- マイ代理人割り当て
- マイ委任先割り当て
- チームの代理人割り当て
- チームの委任割り当て
- チームの可用性

電子メール通知テンプレートが定義されていることを確認してください。これらを定義するには、*［管理］*>*［プロビジョニング］*>*［委任、代理人、およびタスク］*を選択します。