OpenAM＆OpenIDM技術解説

株式会社野村総合研究所
オープンソースソリューション推進室
林田　敦

野村総合研究所のOpenStandia（オープンスタンディア）は、おかげさまで、2006年のサービス開始から2011年までの5年間で契約数累計が1,000件を突破いたしました！

株式会社 野村総合研究所 オープンソースソリューション推進室
Mail：ossc@nri.co.jp　Web：http://openstandia.jp/

はじめに

自己紹介

- 林田　敦

- 所属部署
  - オープンソースソリューション推進室
  - OSSを使ったシステム構築から運用までワンストップでサポート
  - 対象OSSは50種類以上

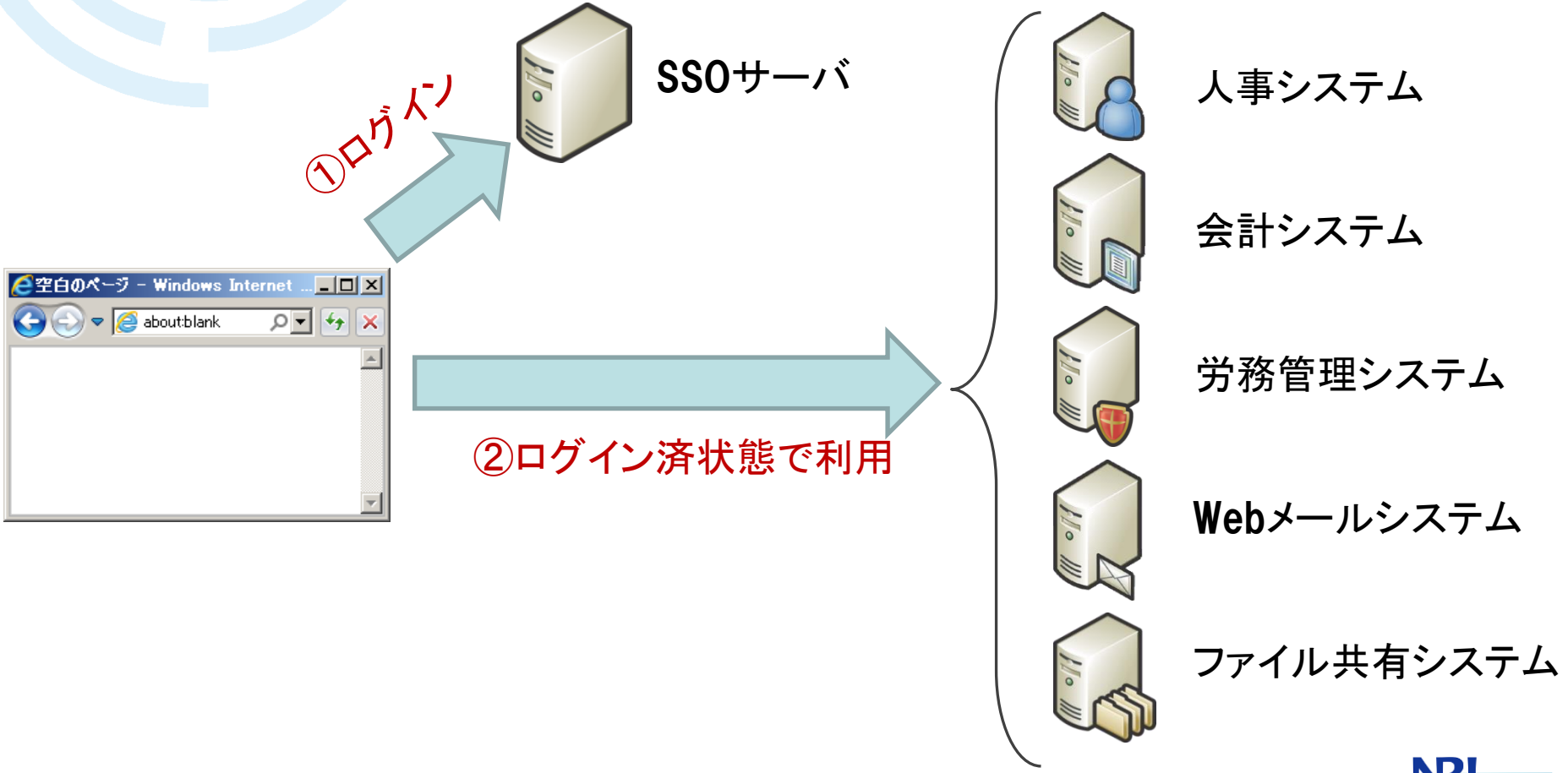
- 私の担当
  - OSSサポート
  - SI
  - 維持管理

SSOとは？IDMとは？

シングルサインオン(SSO)とは

オープンソースまるごと
OpenStandia™
Open Source Technology

- 一度のログイン(サインオン)で複数のアプリケーションがログイン済状態で利用できる仕組み

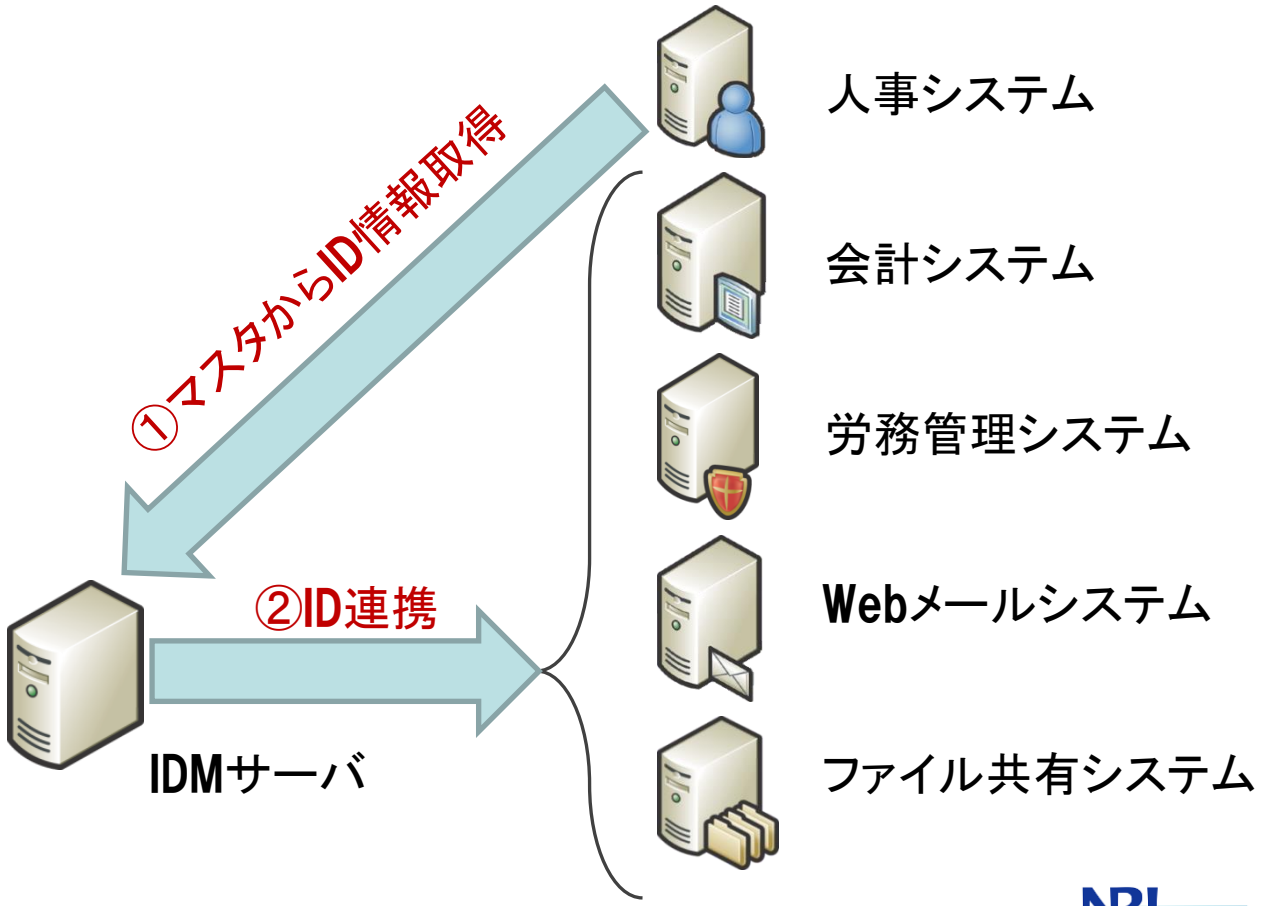


NRI 未来創発
Dream up the future.

IDマネジメント(IDM)とは

オープンソースまるごと
OpenStandia™
Open Source Technology

- IDのライフサイクルを管理すること



NRI 未来創発
Dream up the future.

SSOとIDMを組み合わせるメリット

- **ユーザにとってのメリット**
  - システムごとの認証が無くなって手間が減る(SSO)
  - ID/PWの管理(記憶や更新)が楽になる(SSO)
- **システム管理者にとってのメリット**
  - アカウント情報の一元化により、“きちんと”運用できる(IDM)
    - ✓権限の管理(権限つけ忘れの回避など)
    - ✓インシデント発生時の適切な対処
  - アカウント情報の一元化により、セキュリティが向上する(SSO／IDM)
  - ユーザの認証方法を変更しやすい(指紋認証機能の追加など)(SSO)

OpenAMの紹介

OpenAMの紹介

1. OpenAMの概要
2. OpenAM 10.0 の新機能
3. OpenAM 10.1 (Xpress) の新機能
4. OpenAM 11.0 の新機能
5. NRI拡張機能について

1. OpenAMの概要

- SSOを実現するためのOSS
- 旧Sun Microsystems社の商用製品(OpenSSO)がベースであるため高品質かつ多機能
- ForgeRock社が継続開発中
- CDDL(Common Development and Distribution License)ライセンスで、ソースコードを無償で使用、改変、再配布可能
- 最新の安定バージョン(コミュニティ版)は11.0

SSO方式

OpenAMの代表的なSSO方式

SSO方式	説明
エージェント方式	アプリケーションが動作するサーバに直接エージェントを導入する方式。
リバースプロキシ方式	リバースプロキシサーバ(通常はApache)にエージェントを導入し、バックエンドにいる複数のアプリケーションサーバに対してリバースプロキシする方式。
代理認証方式	代理認証とは、ユーザからのログインリクエストをエミュレートし、認証を代行すること。OpenIGと連携することで、代理認証が可能となる。 連携先システムで、HTTPヘッダから認証情報を取得するカスタマイズが出来ない際に採用する方式。
SAML	SAMLとは認証情報を表現するためのXML仕様。主にSalesforce、GoogleAppsとSSOする際に採用する方式。
OpenID Connect	OAuth2.0をベースとするシンプルな新しいID連携プロトコル。OpenAM11.0から利用可能。主にクラウドサービスとのSSO方式として今後の主流になると思われる。

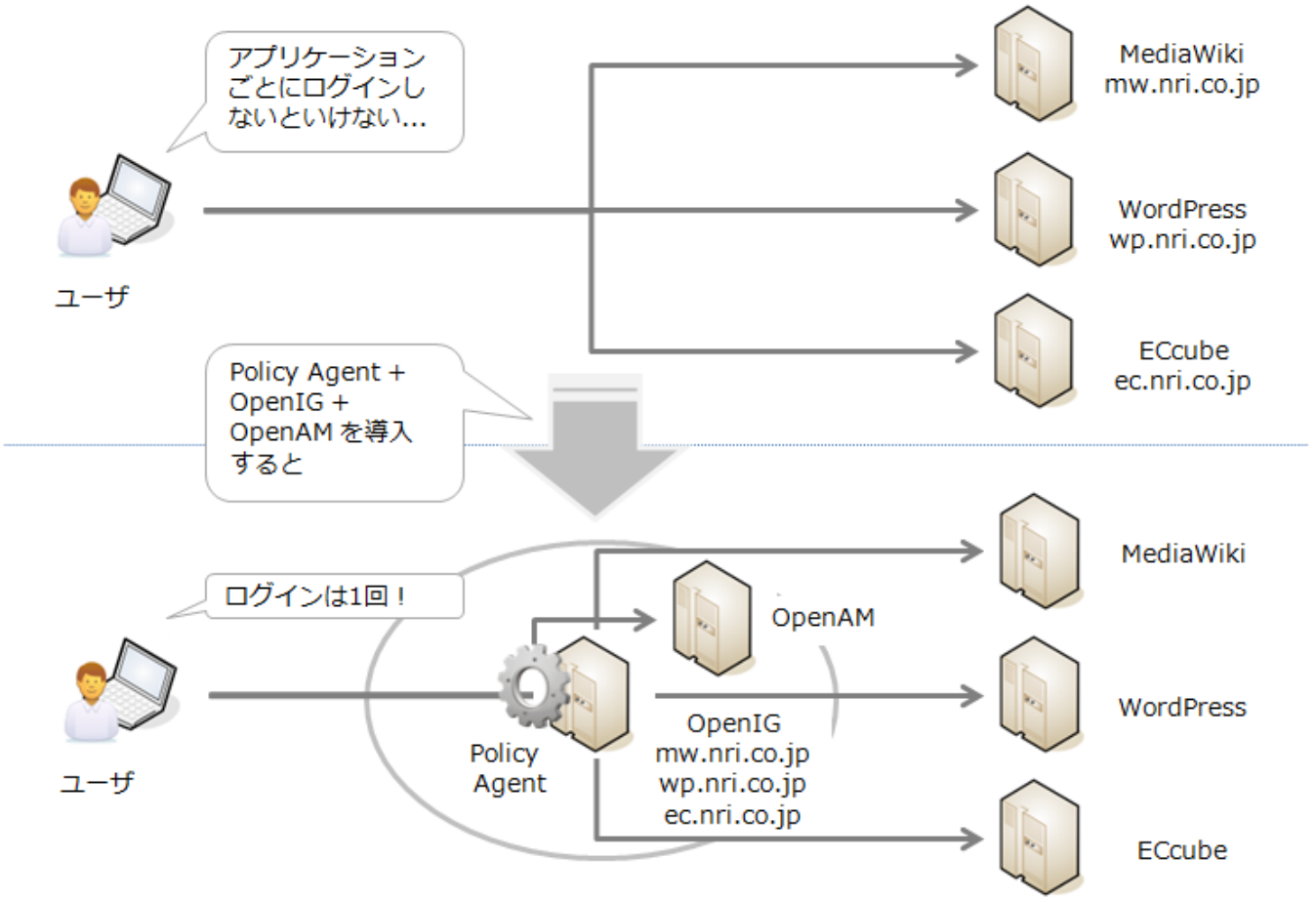
2. OpenAM 10.0 の新機能

- OpenIG
- リスクベース認証

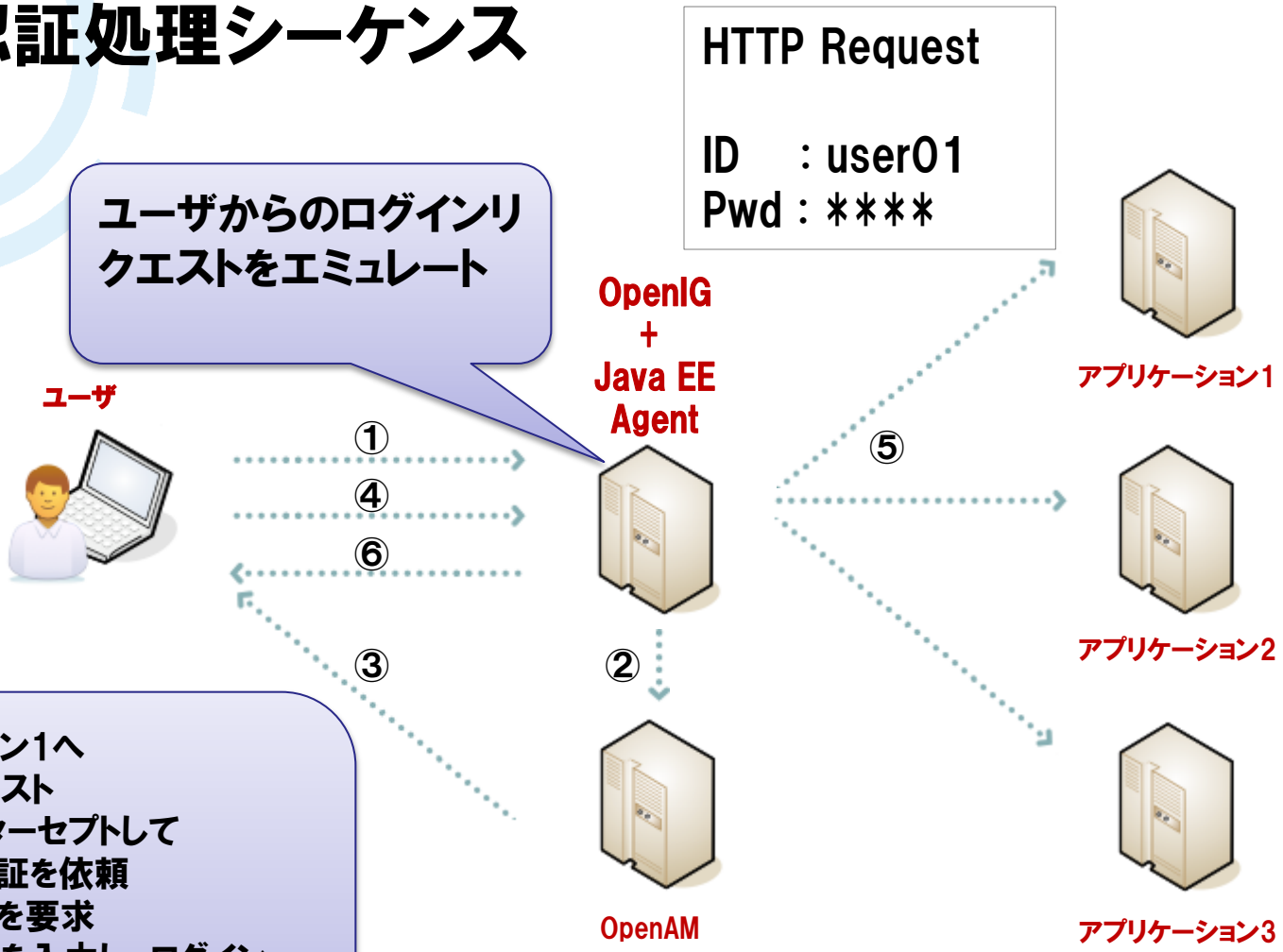
OpenIGとは

- 代理認証を実現するソフトウェア
- OpenAMとは独立した製品
- 基本的にはOpenAMと連携して動作させる
- リバースプロキシ型
- 単独でリバースプロキシサーバとして動作
- HTTPリクエストをエミュレートして認証を代行

OpenIGによる代理認証



代理認証処理シーケンス

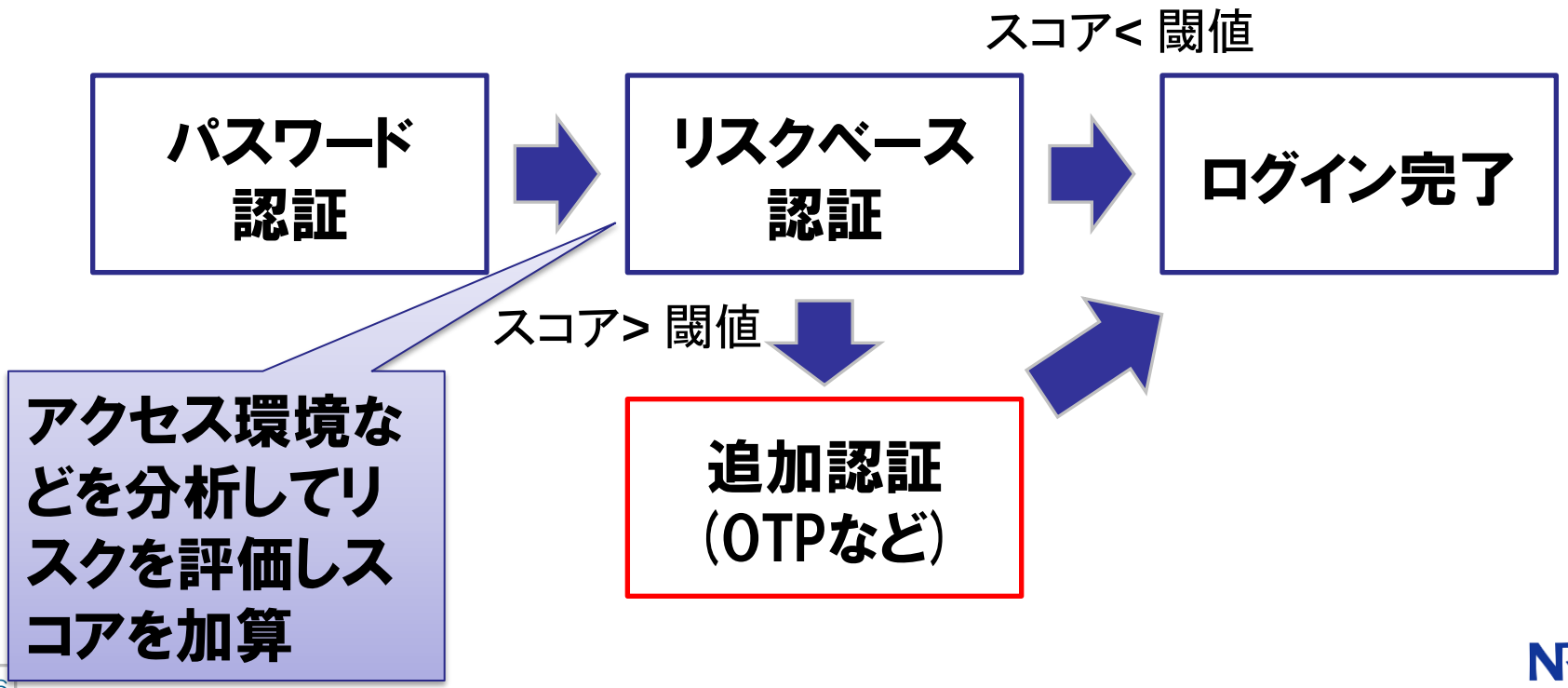


① アプリケーション1へ
ログインリクエスト
② Agentがインターセプトして
OpenAMへ認証を依頼
③ ユーザに認証を要求
④ ID、パスワードを入力し、ログイン
⑤ ユーザからのログインリクエストを
エミュレートし、認証を代行
⑥ ログインレスポンスを返す

SAML2.0 フェデレーションゲートウェイ機能

リスクベース認証

- 不正アクセスのリスクに配慮した認証方式
- ログイン時の地理的位置の評価、最終ログインからの経過時間や認証失敗回数のチェック、IPアドレスの履歴チェックを元に、追加の認証を要求する



リスク評価チェックロジック

チェック方法	概要
認証失敗チェック	ユーザーが過去に認証失敗をしているかをチェックする。 ※LDAPパスワードポリシーのアカウントロックと同時には使用不可。
IPレンジチェック	クライアントIPアドレスが指定した範囲内にあるかをチェックする。
IP履歴チェック	アクセスした際のIPアドレスがユーザープロファイルに記録されているIPアドレスの履歴リストに存在するかをチェックする。
既知のCookie値チェック	クライアントのリクエストに既知のCookieが存在し、正しい値を持っているかをチェックする。
最終ログインからの経過時間チェック	ユーザーのログインが、最後にログインした時刻から指定した経過時間内であるかをチェックする。
プロファイルのリスク属性チェック	ユーザープロファイルに指定した属性と値が含まれているかをチェックする。
デバイス登録Cookieチェック	クライアントリクエストに指定された名前のCookieが含まれているかをチェックする。
位置情報国コードチェック	位置情報データベースを利用してクライアントのIPアドレスをチェックする。 位置情報データベースはMaxMindのバイナリフォーマットが利用可能。
リクエストヘッダーチェック	クライアントリクエストが必須で指定されたヘッダーおよび値を含んでいるかをチェックする。

リスク評価

- 前述のチェックに設定したスコアの合計値が、リスク閾値に到達しなければ認証成功となる
  - 例）以下のように設定した場合
    a. 認証失敗のチェックのスコア = 1
    b. 位置情報国コードチェックのスコア = 2
    c. リクエストヘッダーチェックのスコア = 3
    d. リスク閾値 = 4

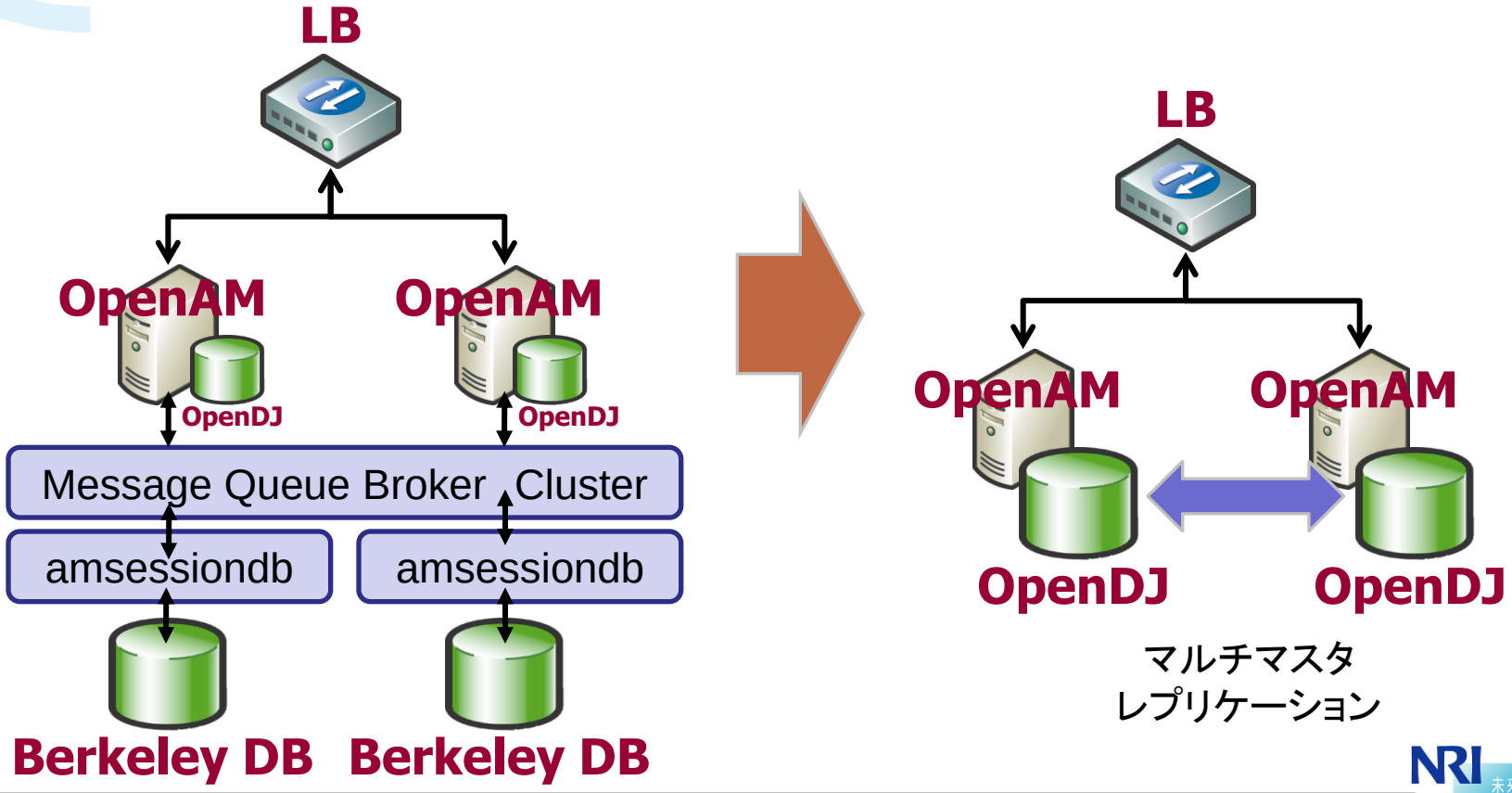
    ケース1: (a) + (b) < (d) ⇒ 認証成功
    ケース2: (a) + (c) = (d) ⇒ 認証失敗

3. OpenAM 10.1 Xpress の新機能

- セッションフェイルオーバの改良
- OATH対応

セッションフェイルオーバーの改良

- 設定が飛躍的に簡単に
  - 以前はOpen Message QueueとBerkeley DBの利用が必須で、構成が複雑
- OpenDJのレプリケーション機能を利用
  - 内蔵の設定データストア（OpenDJ）にセッションデータを書き込む

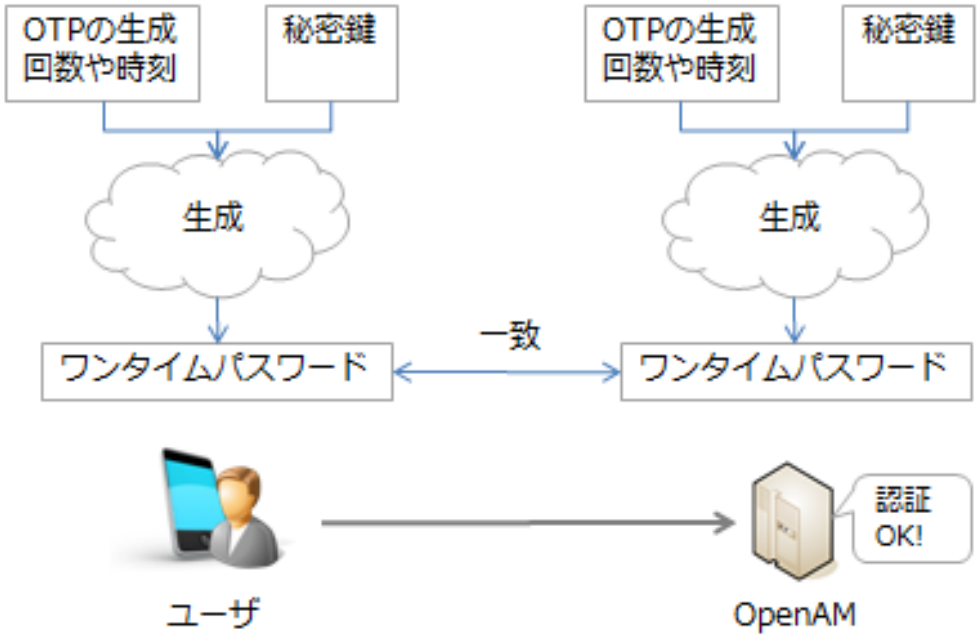


セッションフェイルオーバーの改良

- 構築時のサイト設定で「Enable Session HA Persistence and Failover」にチェックを入れる

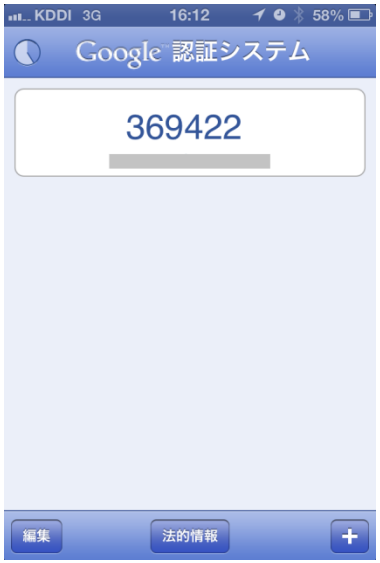
OATH対応

- オープンな認証仕様であるOATH(Initiative for Open AuTHentication)に準拠したワンタイムパスワード認証に対応
- 2種類のワンタイムパスワード方式
  - HOTP：カウンタベース
    - ✓OTPの生成回数(カウンタ)をもとにOTPを生成
  - TOTP：時刻ベース
    - ✓時刻をもとにOTPを生成



OATH対応

- オープンな標準仕様のため、OATH対応のトークン発行アプリ/デバイスをそのまま利用可能
  - Google Authenticator
  - Android Token
  - DS3 Oath
  - Yubikey



(出所) Android Token https://code.google.com/p/androidtoken/
DS3 Oath https://play.google.com/store/apps/details?id=uk.co.bitethebullet.android.token&hl=ja
Yubikey http://www.flickr.com/photos/yubikey/8357169183/

4. OpenAM 11.0 の新機能

- OpenID Connect 1.0のサポート
  - http://www.atmarkit.co.jp/ait/articles/1406/13/news004.html
- JDK7対応
- RESTful Webサービスの追加
  - JSON形式のいまどきのAPIになりました
- IPv6サポート

※リリースノート
http://docs.forgerock.org/en/openam/11.0.0/release-notes/index.html

5. NRI拡張機能

- セキュリティ強化
  - XSSチェック、リダイレクトURLチェック
- JDK7対応
- OpenStandia/Portal (Liferay) との連携機能
- 代理認証機能 (OpenIGとは異なる独自実装版)
- 各種バグ修正
  - メモリリーク
  - マルチスレッドでの問題 など

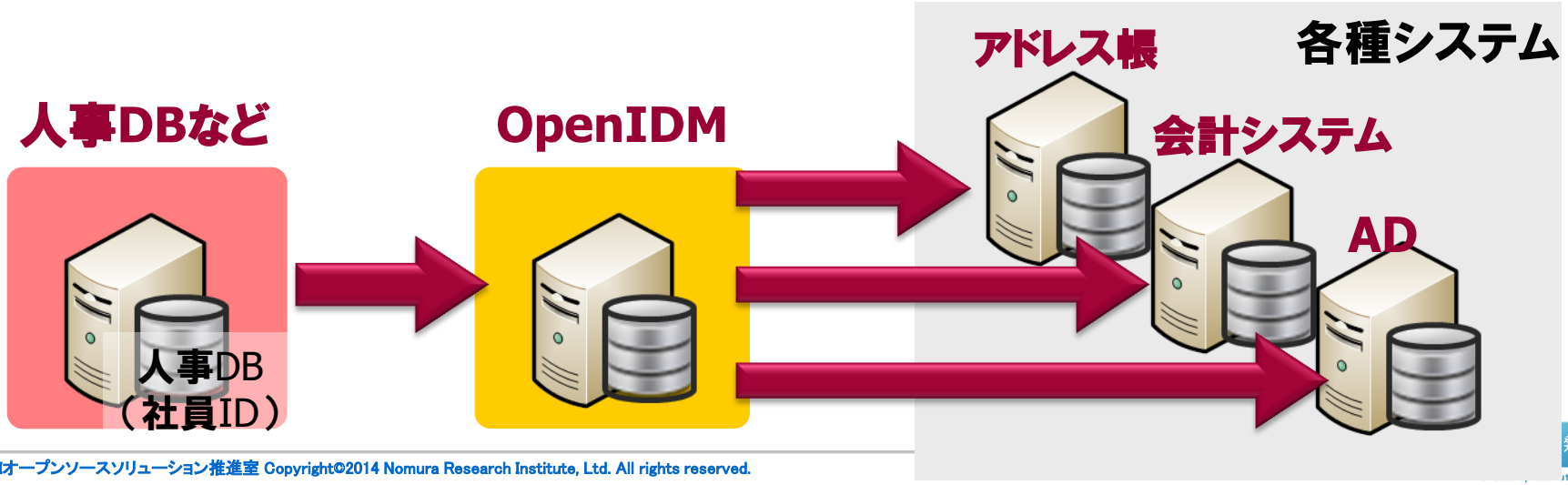
OpenIDMの紹介

OpenIDMの紹介

1. OpenIDMの概要
2. OpenIDMのアーキテクチャ
3. 同期機能
4. ワークフロー連携機能

1. OpenIDMの概要

- OSSのアイデンティティ管理(ID管理、アイデンティティーマネジャー)製品
- ForgeRock社により2010年からフルスクラッチで開発
- オープンスタンダードな技術の採用、モジュラー型アーキテクチャ、外部リソースとのコネクタにOpenICFを採用、REST APIの採用などによって、高い柔軟性と拡張性を備えたアイデンティティ管理製品



OpenIDMの特徴

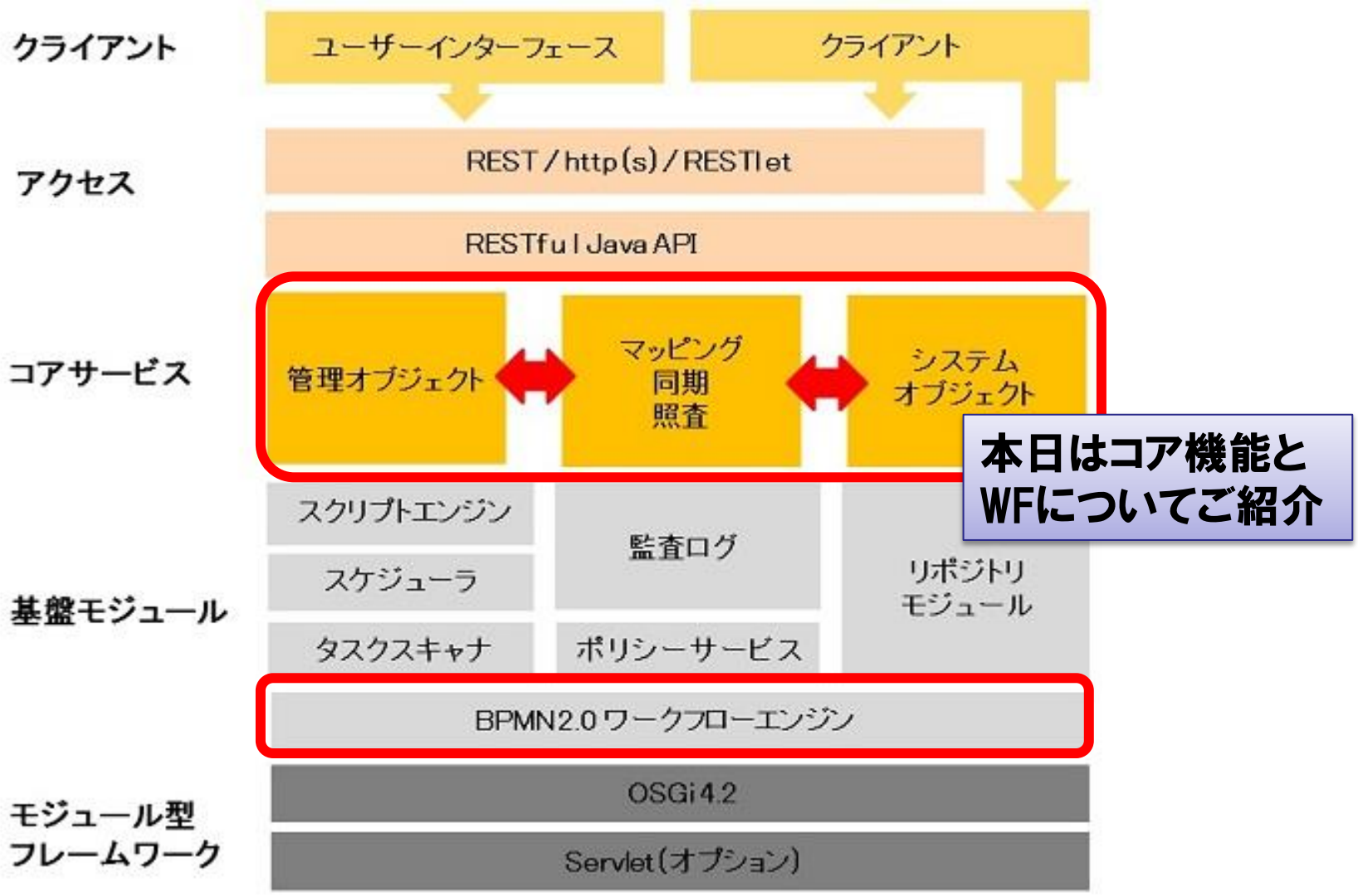
- REST APIの採用
  - OpenIDMに対するあらゆる操作をHTTPで行うことが可能であり、他システムとの連携が容易に可能
- サーバーサイドスクリプトエンジン
  - Java上で動作するJavaScriptエンジン（Rhino）を組み込んでおり、設定情報、マッピング情報、カスタムロジックを柔軟に定義可能
- 柔軟なデータモデル
  - ID情報のスキーマを要件に合わせて柔軟に定義可能
  - データはJSON形式で格納される

他製品との機能比較

- 基本的な機能は実装されつつある
- OpenIDMで不足している機能については拡張機能としてNRIで実装中

機能	OpenIDM	他OSS製品	商用製品A
Webブラウザによる設定画面	○	○	○
データ管理機能	○	○	○
データ検索機能	○	○	○
データ同期機能	○	○	○
IDの一括登録、一括変更	○(CSV)	○(CSV)	○
LDAPグループの作成、変更	○	○	○
IDのLDAPグループへの配属情報の一括登録	○	○	○
CSVアップロード機能	×	○	○
CSVダウンロード機能	×	○	○
パスワード自動生成機能理	△	×	○
メール通知機能	○	×	○
オンラインサインアップ機能	○	×	○
マルチバリューカラムの編集	○	×	○
プロビジョニング先としての任意テーブルの選択	○	○	○
アカウントロック解除機能	×	×	○
複数管理者によるユーザー管理機能	○	×	○
管理者の階層化機能	×	○	○
管理範囲の指定機能	○	○	○
エンドユーザへの情報公開機能(ユーザー自身のプロフィール画面)	○	×	○
プロビジョニング先としてOracle、LDAPおよびMySQLのサポート	○	○	○
ロールによる権限管理	○	×	○
認証DB更新履歴出力機能	○	×	○
ログ出力	○	○	○

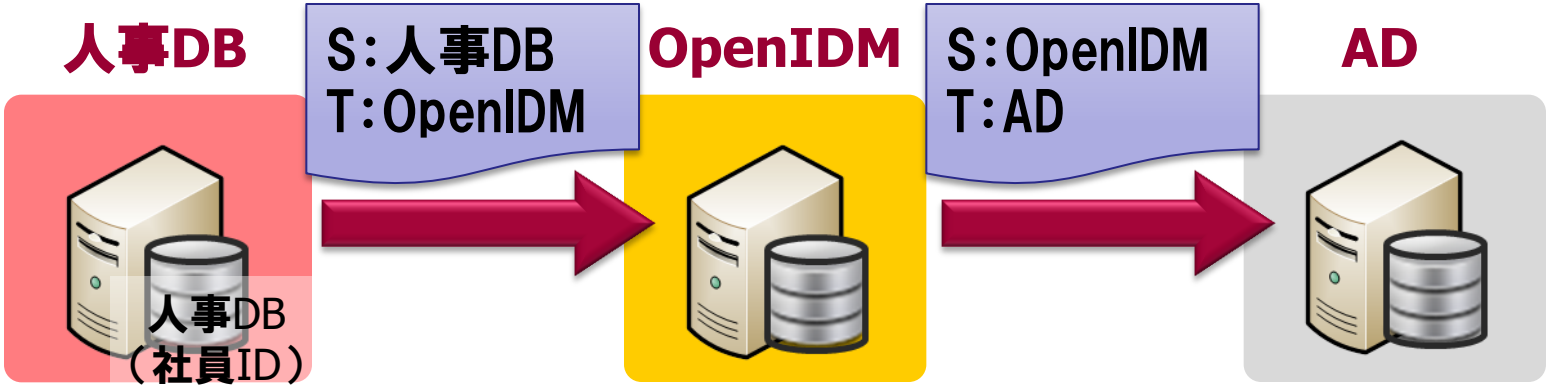
2. OpenIDMのアーキテクチャ



(出所)野村総合研究所 OpenStandia OSS紹介 OpenIDM最新情報 http://openstandia.jp/oss_info/openidm/

3. 同期機能

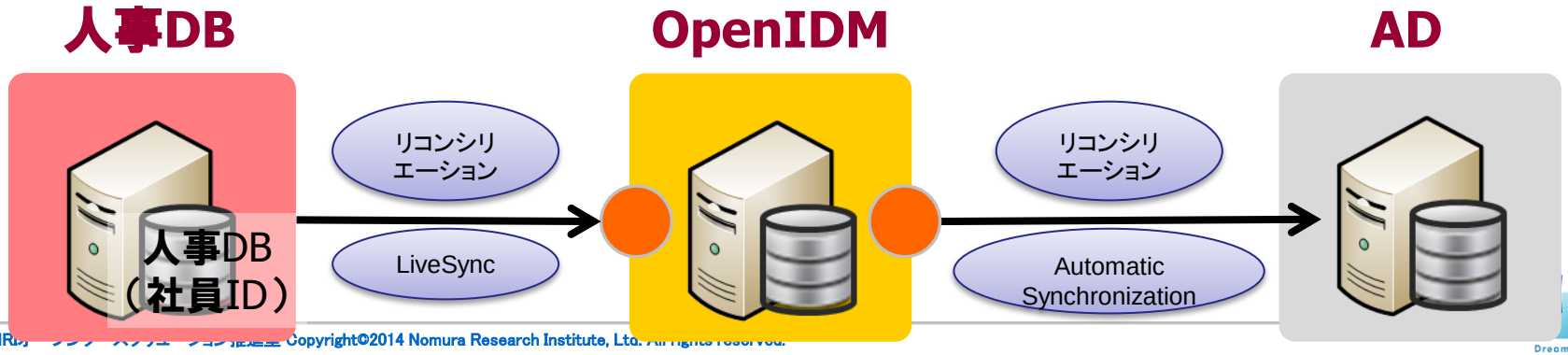
- OpenIDMでは双方向の同期をサポートしている
- ソースとターゲットを指定して同期の設定を行う
  - ソースを源泉データ、ターゲットをOpenIDMのリポジトリのデータと設定すると源泉データからのデータ取り込みとなる
  - 逆に、ソースをOpenIDMのリポジトリ、ターゲットを外部リソースに設定すれば、プロビジョニング処理となる



OpenIDMの同期機能

同期処理方式

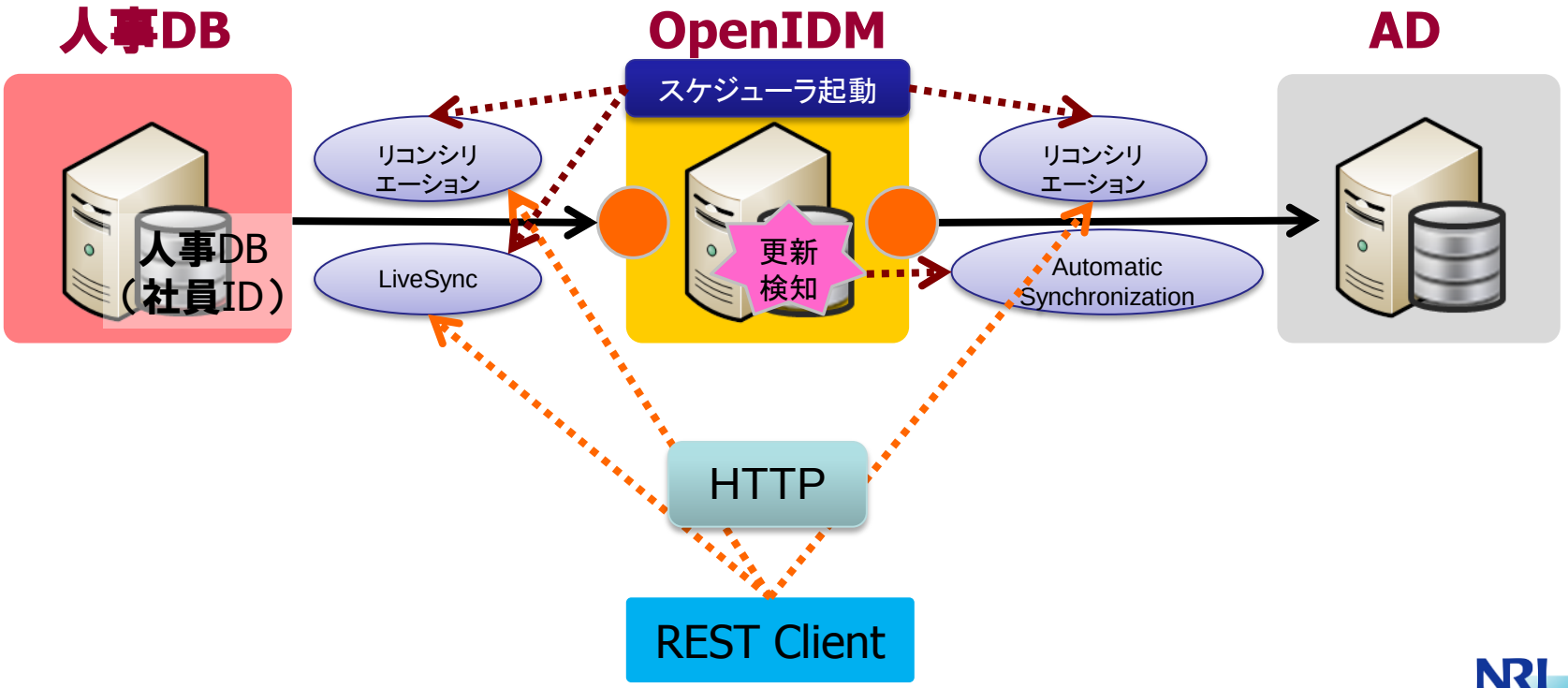
- Reconciliation（リコンシリエーション）
  - ✓いわゆる差分同期
  - ✓ソースとターゲットを比較し変更点を検知して同期を行う
- LiveSync
  - ✓外部リソースから変更点情報を取得して同期する方法
  - ✓不必要な差分チェックを行わないためリコンシリエーションと比べると軽いプロセス
  - ✓ただし、コネクタ・外部リソースがLiveSyncに対応している必要がある
- Automatic Synchronization
  - ✓リポジトリの更新を検知し、その更新内容をターゲットである外部リソースに反映する



OpenIDMの同期機能

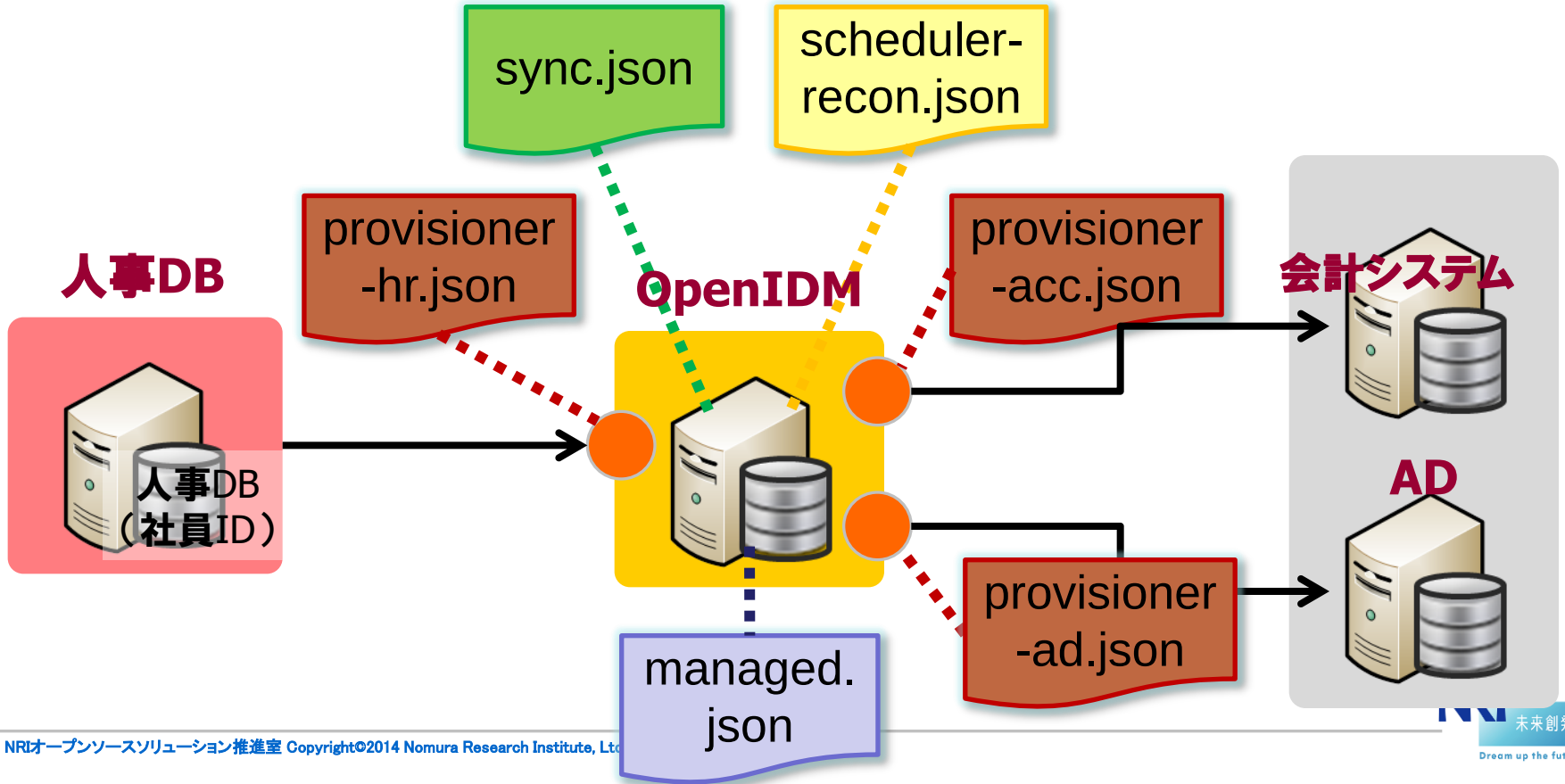
同期処理の実行トリガー

- REST APIによるマニュアル実行
- スケジューラによる自動実行
- リポジトリの更新検知による自動実行（Automatic Synchronization）



同期に関する設定 全体像

- OpenIDMのリポジトリに格納するデータ(Managed Objects)を定義(managed.json)
- 外部リソース(System Objects)を表すコネクタ定義を設定 (provisioner-*.json)
- System ObjectsとManaged Objectsをマッピング定義(sync.json)
- 実行スケジュールを定義(scheduler-*.json)



Managed Objects と System Objects

Managed Objects

- OpenIDMのリポジトリで管理されるオブジェクト
- JSON形式で格納される
- デフォルトではユーザーオブジェクトのみだが、任意のオブジェクトを定義可能
  - ✓managed.jsonで定義

```
{
   "objects": [
      { “name”: “user”    },
      { "name": "group” }
      ・・・
```

System Objects

- 連携先の外部システムのデータを表すオブジェクト
- コネクタ定義を行うことで表現
  - ✓provisioner-*.jsonで定義

```
{
  "name" : “HR",
  "connectorRef" : { ・・・}
  "poolConfigOption" : { ・・・}
  "configurationProperties " : { ・・・}
  "objectTypes" : {
      "account" : {
```

マッピング定義

- sync.jsonにソースとターゲット間でマッピングする項目名を定義する
- スクリプトで加工することも可能
  - 例えば、姓と名を結合して渡す場合など

```
{
   "mappings" : [
      {
         "name" : "systemHrAccounts_managedUser",
         "source" : "system/HR/account",
         "target" : "managed/user",
         "properties" : [
            { "source" : "_id",           "target" : "_id" },
            { "source" : "lastName",  "target" : "lastName" },
            { "source" : "firstName",  "target" : "firstName" },
            { "source" : "",              "target" : "displayName",
              "transform" : {
                 "type" : "text/javascript",
                 "source" : "source.lastName + ' ' + source.firstName"
              }
            }
            ...
```

ソース：System Objects
ターゲット：Managed Objects
を指定

姓・名を結合

スケジュール設定

- schedule-*.jsonに同期（リコンシリエーション）の実行タイミングを定義する

実行間隔を指定

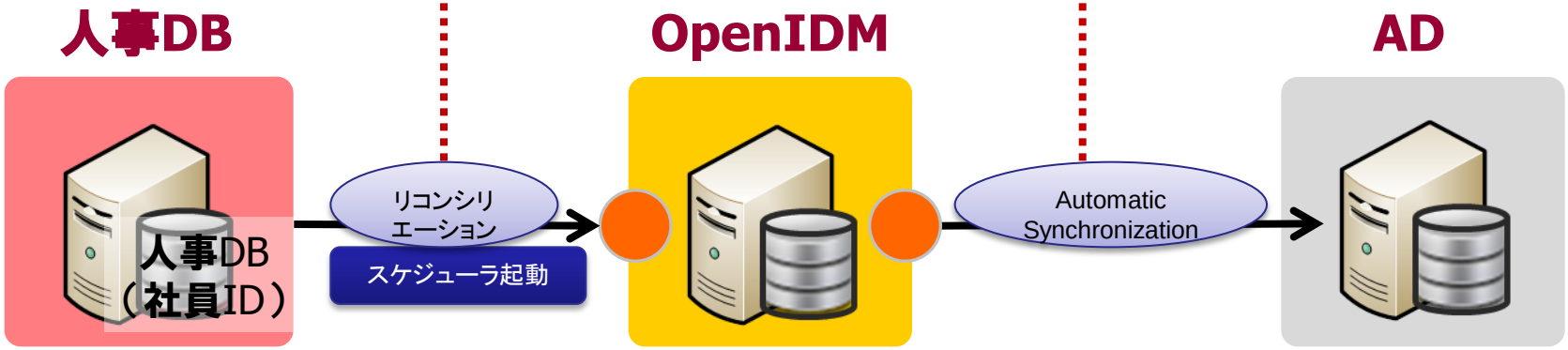
```
{
    "enabled" : true,
    "type": "cron",
    "schedule": "* * 2 * * ?",
    "concurrentExecution" : false,
    "invokeService": "sync",
    "invokeContext": {
        "action": "reconcile",
        "mapping": "systemHrAccounts_managedUser"
    }
}
```

実行する同期処理のマッピング定義名を指定

リコンシリエーションによる同期例

① スケジューラ機能で源泉データ取り込みのリコンシリエーションを実行指示
（毎日AM 2時)人事DBのデータとOpenIDMを比較して差分同期

② ①の同期完了後、ソース(この場合OpenIDM)の変更を検知し、Automatic Synchronizationにより差分同期される
（全件比較されるわけではない）

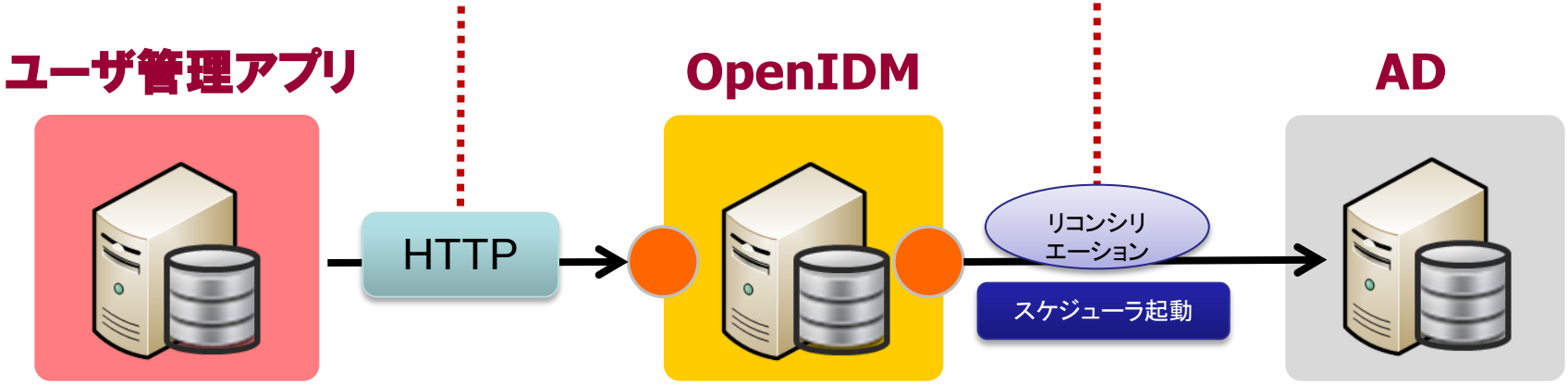


より高度なプロビジョニング

- 発令日ベースで連携先システムにIDをプロビジョニングするケース

① REST APIで直接データを登録・更新
発令日を登録

② スケジューラ機能でADへのプロビジョニングのリコンシリエーションを実行指示(毎日AM 4時)
OpenIDMとADを比較して差分同期を行うが、対象を 発令日 >= システム日付 のユーザとする。



（補足）OpenIDMで連携可能な外部リソース

- 標準では以下のコネクタが利用可能
  - CSV File
  - LDAP
  - Scripted SQL
  - XML File

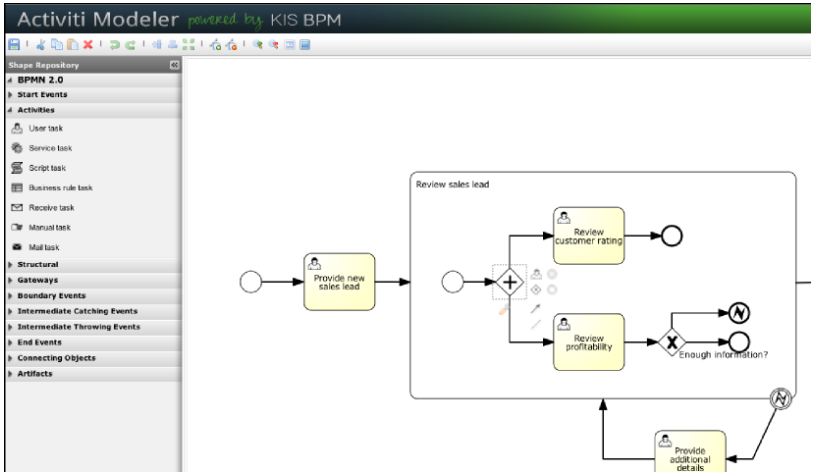
- OpenICFで提供される追加コネクタ
  - Database Table Connector
  - GoogleApps Connectior

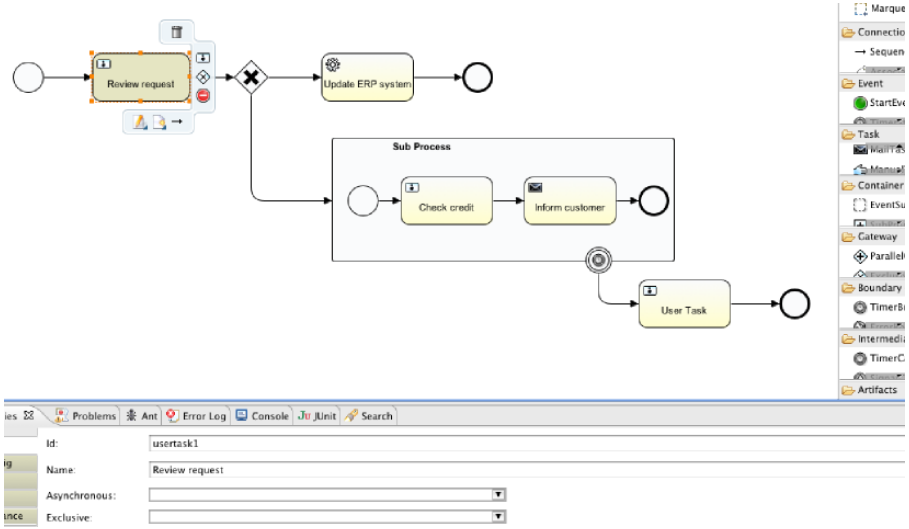
・・・などが提供されている

OpenICFの仕様に従って独自のコネクタを作ることも可能

4. ワークフロー連携

- ワークフローエンジンにActivitiを利用
- ワークフロー定義はActiviti Eclipse Designer, Activiti Explorerを利用する



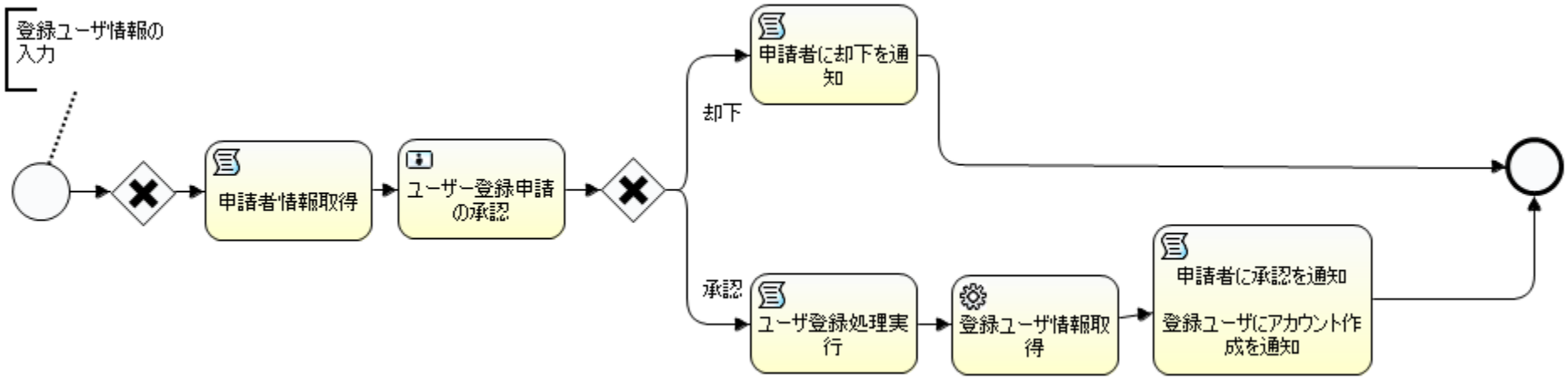


(出所) http://www.activiti.org/screenshots.html

ワークフロー連携

例：ユーザー登録のワークフロー

- 申請者は、登録アカウントの情報を入力する
- 承認者が承認を行うと、ユーザー登録を行い、申請者と登録アカウントに通知を行う
- 承認者が却下した場合は、ユーザー登録は行われず、申請者に却下を通知する



ワークフロー連携：動作イメージ

オープンソースまるごと
OpenStandia™
Open Source Technology

- 申請者（user1）でログイン
- 申請可能なワークフロー一覧から「ユーザー登録申請」を選択する

NRI 未来創発
Dream up the future.

ワークフロー連携：動作イメージ

オープンソースまるごと
OpenStandia™
Open Source Technology

登録ユーザ情報を入力し、申請を行う

NRI 未来創発
Dream up the future.

ワークフロー連携：動作イメージ

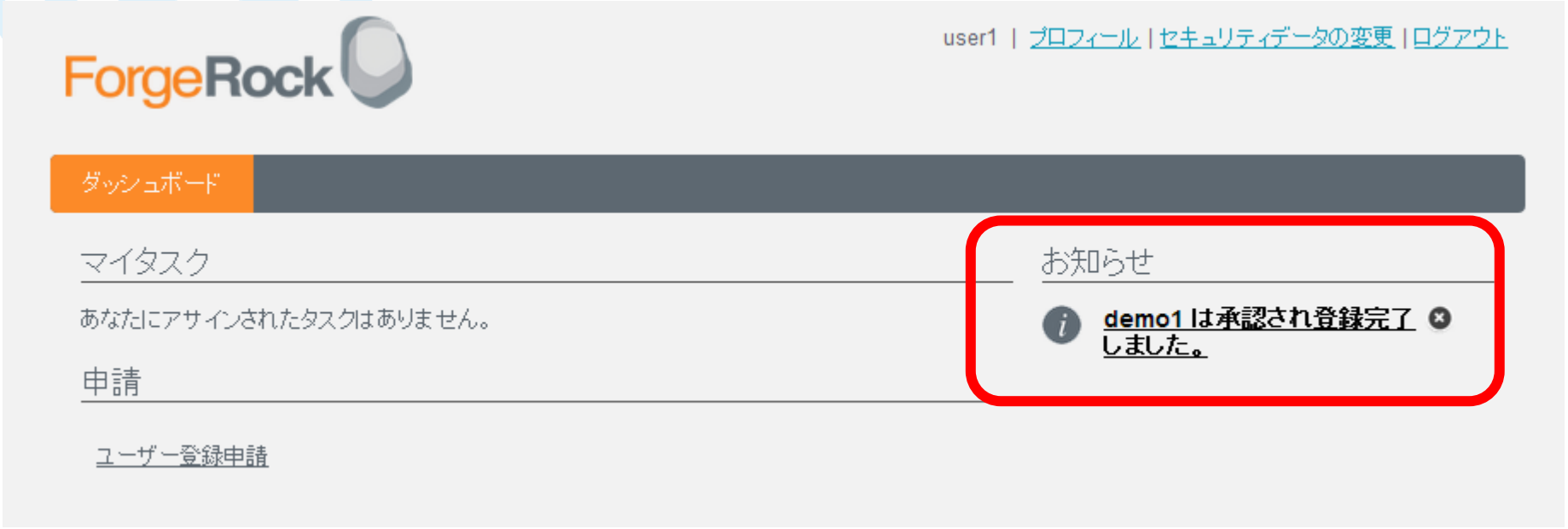
- 承認者（manager1）でログイン
- ユーザー登録申請が表示されるので、自分にアサインする

ワークフロー連携：動作イメージ

- アサインすると、マイタスクに表示されるようになる
- 登録情報を確認し、承認を行う
- 承認が行われると、内部でアカウント登録処理が行われる

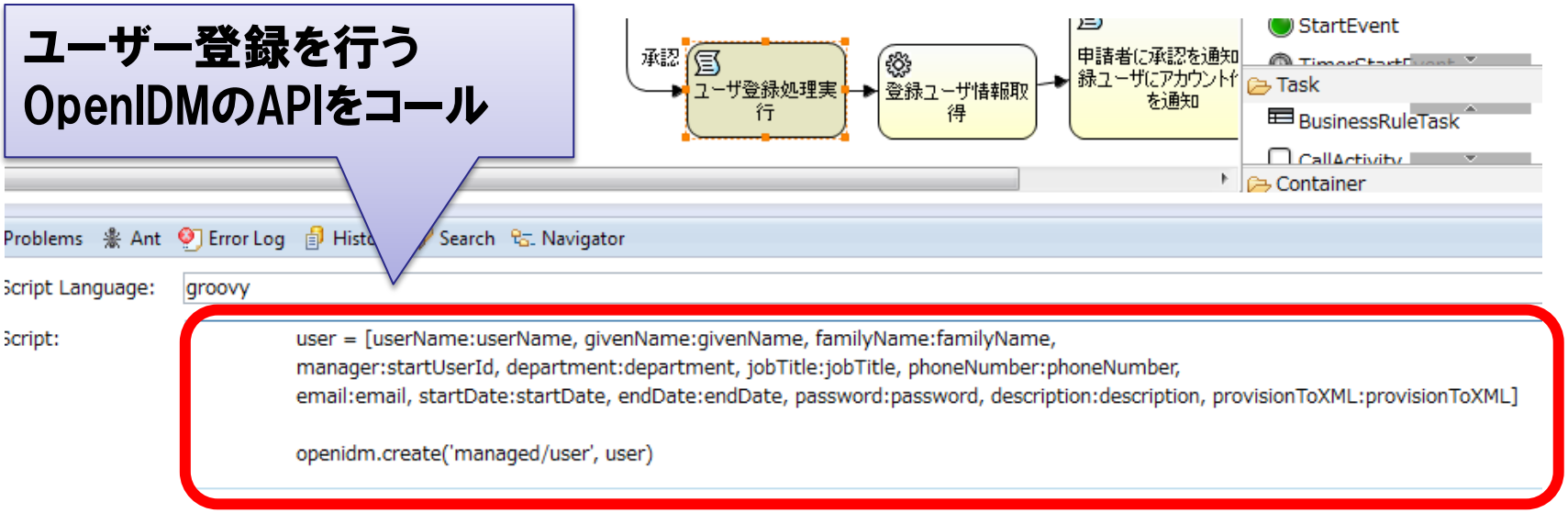
ワークフロー連携：動作イメージ

- 登録完了後、申請者（user1）に登録完了が通知される



ワークフロー連携の注意点

- ワークフローの各タスクの内容はOpenIDMのAPIをコールするように実装する必要がある
- お絵描きするだけで簡単に動くものではない
- 例）ユーザー登録タスク



- OpenStandiaは、「攻めのIT」を支援します。
- オープンソースのことなら、なんでもご相談ください！

オープンソースまるごと

OpenStandia™

Open Source Technology

お問い合わせは、NRIオープンソースソリューションセンターへ

ossc@nri.co.jp

http://openstandia.jp/

NRI 未来創発 Dream up the future.