Migration from Sun IdM

WORK IN PROGRESS

This page is work in progress. Some information presented on this page may not be correct or fully applicable yet.

Intro

In this article I try to compare Sun Identity Manager (now: Oracle Waveset) and Evolveum midPoint Identity Manager - both being user provisioning tools. What do they have in common and is there a way how to migrate Sun IDM deployments to open source midPoint solution?

IDM Product concept

Sun Identity Manager (IDM) was a provisioning product. Not a SSO server, not LDAP. Its primary purpose was to create and maintain accounts and access rights in target systems based on automatic or manual actions. There were multiple mechanisms how to manage the access rights or how to set the account attributes.

Very much of the previous definition applies also to Evolveum midPoint. It is an identity provisioning and synchronization tool but it is already evolving to more general synchronization (e.g. to provision and synchronize also other-than-account objects, like organizations or groups).

As both concepts are so similar, midPoint can replace the Sun IDM deployment after its end of life, and such replacement (or transformation) could reuse a lot of Sun IDM-based deployment concepts, which saves money, effort and time.

Software Dependencies

Sun IDM was relatively independent on the database and application server used, thanks to its predecessor "Waveset Lighthouse". You could run it on almost any common RDBMS and almost any J2EE application server.

Evolveum midPoint does not depend on any particular RDBMS or application server neither. For most deployments we recommend PostgreSQL RDBMS and Apache Tomcat (both being open source software as well).

GUI

Sun IDM user interface (Administration and End User) was JSP-based web interface which utilized reasonably recent browsers. Some Java applets were used in some versions, but generaly you could avoid them. There was no AJAX technology used.

Evolveum midPoint interface (common for Administrators and End Users) is based on Wicket web framework and also works in reasonably recent browsers. No Java applets are needed, Javascript/AJAX is used.

The midPoint user interface can be completely exchanged (e.g. standalone End User interface) by partners or customers, because midPoint provides webservice interface to access its data. This also means that the GUI may not be used at all and replaced by remote webservice calls. The REST implementation is currently in progress.

Adapters, Connectors and Agents

Sun IDM was well-known for its non-invasive approach whereever possible. Provisioning used Java-based adapters for remote communication with target systems. No agents were installed on target systems unless they were absolutely necessary and even then the agent ("Sun Identity Manager Gateway") could was often only a "protocol proxy", e.g. for accessing Microsoft Windows Active Directory interface from non-Windows operating system. In such case, adapters for provisioning were located on the Gateway machine. Adapter API was documented and new adapters could have been created by partners or customers.

The adapters eventually evolved into ICF Connectors. The ICF (Identity Connector Framework) project was open source, although only some connectors were open source. The later Sun IDM releases supported both adapters and connectors.

Evolveum midPoint uses the same non-invasive approach. Provisioning is using ICF connectors (OpenICF project driven by ForgeRock continues open source development of ICF framework), so customers can use OpenICF released connectors and reuse already existing custom connectors (if they have source code). The connectors are deployed on the midPoint machine. Remote interface for managing target system accounts are used. If they do not exist or cannot be used for all operations, .NET / Java Connector Server can be used to host the connectors for provisioning.

Although Sun adapters and ICF connectors have different API, our team is skilled enough to transform customer adapters to ICF connectors so they could be used in midPoint, or to provide assistance for such transformation.

XML Data Model

Sun IDM was well-known for representing the data in XML. This was great for exporting, transforming and importing data.

Evolveum midPoint also uses XML for representing data, but internally, unique Prism object concept is used, enabling more data representation formats in the future (e.g. JSON). Currently all objects can be exported and imported as XML objects.

The XML data representation is different in Sun IDM and midPoint, but as the product concepts are similar, once the user data is exported from Sun IDM, it can be transformed and imported to midPoint. Such tasks are typically done by partners and service integrators during the product deployment.

Expression Language

Sun IDM was using its own functional language "XPRESS" to create conditions, attribute values etc. during user provisioning. The language could also use Sun IDM API methods to access the data. As it was embedded in XML objects, it was interpreted on the machine(s) running Sun IDM. The disadvantage - XPRESS was proprietary, Sun IDM-only language.

Evolveum midPoint uses standard high-level languages to access, transform and process data during provisioning and synchronization. The currently supported languages are Groovy, JavaScript and XPath 2.0. The expressions in these languages are embedded in objects in a way similar to Sun IDM.

As the languages are different, there is no automatic way of porting XPRESS-based expressions to Groovy (or JavaScript or XPath), but for simple expressions the process should be a relatively simple. For the rest, the expressions need to be reimplemented in midPoint-supported expression language.

Relative vs. Absolute Change Model

Sun IDM change model was based on absolute changes. Change of single user attribute was actually a "replace resource attribute value" operation. Thus parallel provisioning request might not work as supposed; for example, if the "Groups" attribute was initially set to "Group1", and if there were two parallel requests pending delays/approval, the first: "add Group2 - thus set Groups to Group1,Group2" and the second "add Group3 - thus set Groups to Group1, Group3", the result could actually vary (Group1,Group2 or Group1,Group3). The problem was that Sun IDM was computing the target attribute value in an absolute way before the request was completed.

Evolveum midPoint was designed with the absolute-value problem in mind. The changes of the attributes are provisioned, not absolute values. The above mentioned parallel request would be expressed as "add Group1 value to Groups attribute" and "add Group3 value to Groups attribute", respectively, much like LDAP. These operations can be completed in any order and no locking mechanisms are needed.

View, Forms, Workflow and Data Processing

Sun IDM view concept was quite unique. During processing of users (or other objects), special variables containing user (or other object) attributes were constructed. These variables were actually assembled tree of useful user/account information which could be used for computing values, decisions etc. Separate views existes for separate operations (Enable, Disable, Rename etc.), which could be extended to fulfill customer or partner needs.

The view concept was used in forms and workflows. Forms were used to display and edit the view using browsers or programatically, using input elements and XPRESS expressions. Workflows were used not only for actual request approvals; instead, Sun IDM would call a workflow (or process) after the form was submitted. The mapping of forms and workflows was fixed, but the forms and workflows could be updated or replaced providing maximal customization.

Evolveum midPoint has no concept of views. The configuration and expressions for provisioning is stored in the resource objects themselves - the place we believe is more logical and straightforward. The expressions are parts of "mappings". For any resource or user attribute there can be one or more (conditional) mappings which map source attribute (e.g. user's Fullname) to target attribute (e.g. Windows DisplayName) with optional expression. If there is an attribute change, the midPoint Model component will recompute the relevant mappings and apply them to other attributes. Mappings can be used also to compute user attributes whenever other user attributes change. In the mappings, special variables are exposed (user, account, shadow etc.) to access the midPoint user, target resource account etc. data.

Forms used in the current version of midPoint are automatically generated depending on schemas. We plan to further enhance the forms to allow also non-generated fields in the future. The workflows in midPoint are used only for approvals (using Activiti framework). No workflows are involved in provisioning if no approvals are configured. The midPoint Model component processes the changes and recomputes the attributes if needed, instead of workflow (as in Sun IDM). This simplifies the concept and the customization, and workflows are used in a more logical way than in Sun IDM.

Notifications

Sun IDM was using e-mail notifications for various events, such as new accounts created, passwords resets etc. The look&feel of the e-mail could be configured using email templates, which could also contain some XPRESS to customize the e-mail content. The notifications were sent from workflows.

Evolveum midPoint notifications can use e-mail or SMS notifications (other transports can be implemented and added). Additionally, the notifications can be stored in a file, which is very useful during testing and deployments (this feature is similar to Sun IDM "redirect to file"). The notification component is configured in global midPoint system configuration in a more programatically way using expressions. There are currently no email templates, but the concept is open to such changes to be added in the future releases.

Documentation

Sun IDM documentation was provided with the product (PDF) or online for free. There were also some attempts to provide wiki information (mainly for developers, such as NetBeans IDE plugin and ICF connectors).

Evolveum communication model is based on openness. All(!) midPoint documentation is public. All. Yes, including roadmap, various howtos, guides, tricks and tweaks.

Discussion/Forum

Sun provided forum where application users (e.g. Sun IDM) could discuss the problems. Anyone could have asked others for support and anyone could have helped by answering. Sun IDM developers could have been also in this list, but no direct communication was possible.

Evolveum communication model is based on openness. Mailing lists (midpoint-dev and midpoint) are of course public and support engineers and developers are members of the list. Anyone can become a member.

Please note that although support engineers and developers read (and often answer or contribute) the mails, the lists do not substitute support services.

For mailing list and other feedback information please visit this page.

Trainings

Sun provided several good trainings for Sun IDM administrators and deployers; we have lectured many of them in Slovakia and in Europe. Although the documentation was available, nothing compares to a good training led by a qualified instructor. Trainings can really open your eyes to see what was hidden (or better, not obvious) and understand the concept. With understanding, the ideas come more likely.

Evolveum currently provides two midPoint trainings. "MidPoint Identity Manager Essentials" is targeted to system administrators to understand major midPoint concepts when taking over the solution and starting maintaining it. "MidPoint Identity Manager Customization and Deployment I" is more challenging and is targeted to deployers - partners, system integrators or even customers. Its purpose is to understand how the product behaves and how it can be customized to fulfill customer's requirements. As we have deployed multiple Sun IDM installations and lectured many Sun IDM trainings, we can ease the knowledge transfer comparing midPoint features to Sun, highlighting major differences.

Summary

Sun IDM and Evolveum midPoint have a lot of similar features. When planning upgrade of your soon-to-be-EOLed Sun IDM, you should consider midPoint, because many of the concepts are similar, e.g. you could export your Sun IDM data in XML, transform and import to midPoint. Both solutions are non-invasive and can even share the same connectors (ICF connectors). Evolveum engineers and consultants have deployed multiple Sun IDM installations and have a deep knowledge of both solutions.

Upgrading to midPoint mitigates the vendor lock-in problem. The software is open-source, scriptable using standard high-level languages (Groovy, JavaScript, XPath) and the ICF connectors can be also used in other solutions.

Feature	Sun IDM	midPoint
License	Proprietary	Apache License 2.0
EOL Announced	Yes	No
Provisioning / synchronization Tool	Yes	Yes

SSO Tool	No	No
Software Dependencies	Lightweight	Lightweight
Provisioning Components	Java Connectors (ICF), Java Adapters	Java Connectors (ICF), Java Adapters (planned)
Invasive / Non-invasive	Non-invasive	Non-invasive
GUI	Web-based (JSP, JavaScript)	Web-based (Wicket, JavaScript, AJAX)
Data Interface	Web services	Web services
Data Representation	XML	Prism objects, XML, JSON (implementation in progress), (more formats planned)
Data Change Model	Absolute	Relative
Forms	Generated ("MissingFields"), Customizable	Generated (based on schema) (Customizable forms implementation in progress)
Roles	Static, Dynamic (rules), Hierarchical	Static, Dynamic (expressions), Hierarchical
Workflows / Approvals	Yes	Yes
Expression Language	XPRESS (Proprietary)	Groovy, JavaScript, XPath2
Notifications	E-mail, File redirection	E-mail, SMS, File redirection (extensible for more transports)
Communication	Discussion forum (public and restricted to partners)	Mailing lists (public)
Documentation	Online (PDF)	Online (Wiki)
Product Trainings	Yes	Yes
Upgradable from Sun IDM	N/A	Yes (with limitations and concept issues)

Implementation Tips

This section contains tips for Sun IDM engineers that helps then to use midPoint efficiently. It describes especially the "hacks" that were often used in Sun IDM and the correct equivalent used in midPoint deployments.

Identity Template

MidPoint does not have a special identity template. Account identifier is considered to be very like an ordinary account attribute. Use outbound mapping to set the value of account identifier instead of identity template.

Login Roles

Sun IDM deployments often used "login roles" or "default roles" to set resource-global policies. Such roles had only one resource and used the ability of Sun IDM role to set account attributes. Other roles then haven't included the resource directly but included the "login role" instead.

Do not use this approach in midPoint. MidPoint has an elegant mechanism of outbound mappings that can be used to set resource-global attribute values. The ability of a login role to "hold" the account in a disabled state can be done in a much easier way by using activation existence mapping.