



White Paper

# Deploying Mobile Web Services using Liberty Alliance's Identity Web Services Framework (ID-WSF)

**NOKIA**  
CONNECTING PEOPLE

 **Sun**  
microsystems®

# Table of Contents

<b>Abstract</b>	3
<b>Identity, mobility and web services: the challenges at hand</b>	3
<b>Liberty Alliance – background and key terminology</b>	4
Liberty Alliance Background	4
Terminology	4
An Overview of Liberty's Identity Web Services Framework (ID-WSF)	5
<b>ID-WSF use case: Merchant application</b>	6
Simplified sequence flow	7
<b>An end-to-end view on architecture and its components</b>	8
<b>Developing applications for Liberty identity based web services – some service examples</b>	9
<b>The Nokia Web Services Framework</b>	10
<b>Sun's Liberty-ready identity solutions – a developers view</b>	11
Architecture	11
Discovery Service	12
SOAP Binding & Data Services template	12
Interaction Service	13
PAOS (reverse HTTP bindings)	13
Getting Started With Sun Java System Access Manager	13
<b>Why you should care</b>	14
The (Java) developer view	14
<b>Liberty Alliance participation</b>	14
<b>Useful web services links</b>	14
About Nokia	15
About Sun Microsystems	15

## Abstract

This white paper gives developers an overview of the Liberty Alliance's Identity Web Services Framework (ID-WSF) and discusses deployment strategies. The paper details solutions from Sun Microsystems and Nokia that will enable developers to architect, develop and deploy identity-based web services to fixed and mobile devices based on the ID-WSF protocols. In addition, this paper provides a sample web service use case as well as a real-life implementation implemented by AOL.

Knowledge of the basic Liberty architecture components is assumed. The Liberty Alliance has published white papers on its architecture, which can be downloaded at <http://www.projectliberty.org/resources/whitepapers>. In addition, the full set of Liberty Alliance specifications can be reviewed and downloaded at <http://www.projectliberty.org/specs>.

## Identity, mobility and web services: the challenges at hand

Identity is at the core of any high-value relationship between a company and its employees, its customers and its business partners. Secure management of these identities is a critical activity that can be used to increase efficiencies, enhanced security and provide for new revenue opportunities. Optimized identity management also allows for more effective delivery of web services.

Web services make it possible for software written in a variety of languages and deployed on a wide range of platforms, from mobile devices and laptops to corporate servers, to interact seamlessly. These services provide a vehicle for the discovery and exchange of key user attributes needed to perform a transaction, such as physical location, device type, access rights, or billing information. Standards-based web services protocols allow applications to concentrate on service value rather than on communications transport. This increased focus on "service" is also referred to as a Service Oriented Architecture (SOA).

As more web services are offered, the opportunities to create rich user experiences will grow. To fully realize this promise requires more than basic web service communications protocols. What is needed are standard specifications that tie together service discovery, invocation, authentication and other necessary components – thereby **adding context and value to web services**. Given a complete, standardized web services framework, companies can easily build end-to-end identity infrastructures that support interoperable web services applications. The Liberty Alliance's Identity Web Services Framework (ID-WSF), available today, is the industry's only set of open specifications meeting these requirements.

Liberty's specifications are uniquely effective for delivery of services to mobile users – where secure authentication, rapid deployment and interoperability are particularly important and where web services such as presence, payment and geolocation take on additional value. The lack of market-ready, open standards and infrastructure solutions has traditionally been a barrier for mobile web service deployment; Liberty's ID-WSF stands to remedy this problem.

# Liberty Alliance – background and key terminology

## Liberty Alliance Background

The Liberty Alliance Project was founded in the fall of 2001 by leading service providers and technology vendors with the stated goal of forming an open industry organization to focus on the technology, business and policy challenges associated with federated identity management and services. Liberty is the **only** open body focusing on these issues and currently consists of over 150 global organizations – ranging from leading technology and service providers to government and non-profit groups.

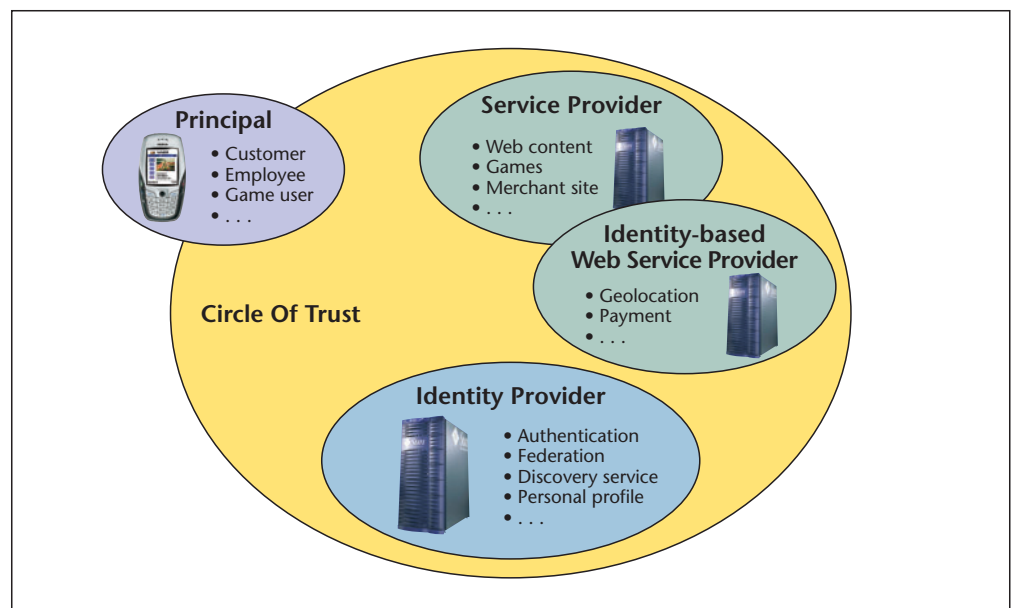
Liberty has emerged as the de facto standard for the mobile industry as its output reflects companies that represent over 200 million mobile subscribers, 50% of the device manufacturers, 80% of all SIM cards, and 55% of the mobile network infrastructure providers.

To date, the Alliance has released the Liberty Identity Federation Framework (ID-FF), facilitating web-based, cross-domain single sign-on (SSO) and account linkage, and also the Liberty Identity Web Services Framework (ID-WSF), which serves as the architectural foundation for current and emerging interoperable Liberty web services, as will be described further in this white paper.

## Terminology

Details on some of the most important Liberty and identity-based web services concepts are depicted in the following diagram and in the definitions below:

Figure 1.

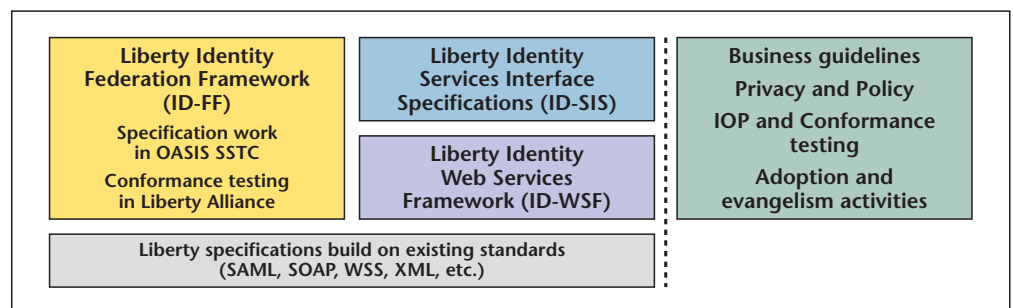


- **Principal** – any entity that can acquire a federated identity, for instance an individual, a legal entity, or a component of the Liberty Alliance architecture
- **IdP – Identity Provider** – a service that authenticates and asserts a Principal's identity. While any entity can become an IdP, it is most likely that organizations that already manage user identity information are most likely to encompass this function, such as mobile operators, banks, and one's employer. It is important to note that a principal can have several Identity Providers.

- **SP – Service Provider** (other than Web services) such as a website (i.e. www.aol.com), or an application running on a server that provides some kind of content or functionality to the users.
- **IS – Identity-based Web Service Provider** – a provider hosting identity-based web services that are invoked through or with a user’s identity.
- **DS – Discovery Service** – a service that provides identity-based discovery of Web Service Providers. A service requestor can use a standardized protocol to access the DS. The DS plays a different role than a generic search directory, or UDDI (specified by OASIS); DS is used to discover the services available to a certain user. In the context of this paper’s focus on identity-based web services, this functionality is particularly important. DS uses privacy-protected name and resource identifiers to provide that functionality. (UDDI can of course also be applied to discover “generic web services”.)
- **WSC – Web Services Consumer** (SOAP Client) can also be described as the requester of a web service.
- **WSP – Web Services Provider** – a SOAP service designed to answer to WSC requests, that fulfills Liberty ID-WSF requirements on schemas, protocols and profiles for providing a data service which leverages identity. Typically a WSP is registered into a discovery service. A WSP may in some cases be hosted by a mobile device, for example to provide a user profile or location service. In Liberty, such a device-hosted service can be accessed by the Reverse SOAP (PAOS) protocol.
- **Circle of Trust** – a group of service providers and identity providers that have business relationships based on Liberty architecture and operational agreements and with whom users can transact business in a secure and apparently seamless environment. Circles of Trust represent the second wave of identity federation, after SSO and federated account linking.

## An Overview of Liberty’s Identity Web Services Framework (ID-WSF)

Figure 2.



The focus of Liberty’s ID-WSF is the definition of a simple, yet rich environment for both offering and using Identity-based web services. “Identity-based” in this context means that the service is associated with a Principal’s Identity. A service can be invoked using the Principal’s Identity while employing suitable privacy and security around information sharing and use. This is also referred to as **permission-based attribute sharing**.

The Liberty specifications fully support the addition of various service interfaces on top of the ID-WSF architecture, to address specific business needs and application functionality. These service interfaces are open to industry development. The Alliance has defined a set of interfaces commonly referred to as the Liberty Identity Service Interface Specifications (ID-SIS). The first interfaces to be implemented under ID-SIS will be Geolocation, Presence, and Contact Book services, followed by Gaming services.

Software tools make it quite simple to interact with the Liberty ID-WSF since the goal is to allow application developers to concentrate on services rather than the underlying communication and authentication protocols. ID-WSF provides for a rich set of interactions based on a comprehensive set of federated identity, permissions, service discovery and invocation services.

It is worth becoming familiar with the key ID-WSF features in order to understand what the architecture can offer to application developers. The following use case serves as an example.

## ID-WSF use case: Merchant application

While the “merchant”, or, more generally the selling/buying model is very well understood, the implementation of such applications requires many advanced features that the Liberty framework helps provide.

The basic logic of such a system is very simple: the customer is willing to buy, the merchant is willing to sell and operator is more than happy to guarantee the transaction for a small percentage of the value.

Nevertheless, the key stakeholders in this transaction have different interests:

- The customer is willing to have the option to buy from any merchant, but is reluctant to trust most of them.
- The merchant is willing to sell to anyone, allowing the customer to experience a seamless “one-click-buy”, while assuring that the merchant receives payment.
- The wireless operator is willing to leverage its direct businesses relationship with their customers, while assuring that opening their user base will not lead to privacy concerns around customer information leakage.

Furthermore, the business relationship between stakeholders is limited and not easy to change.

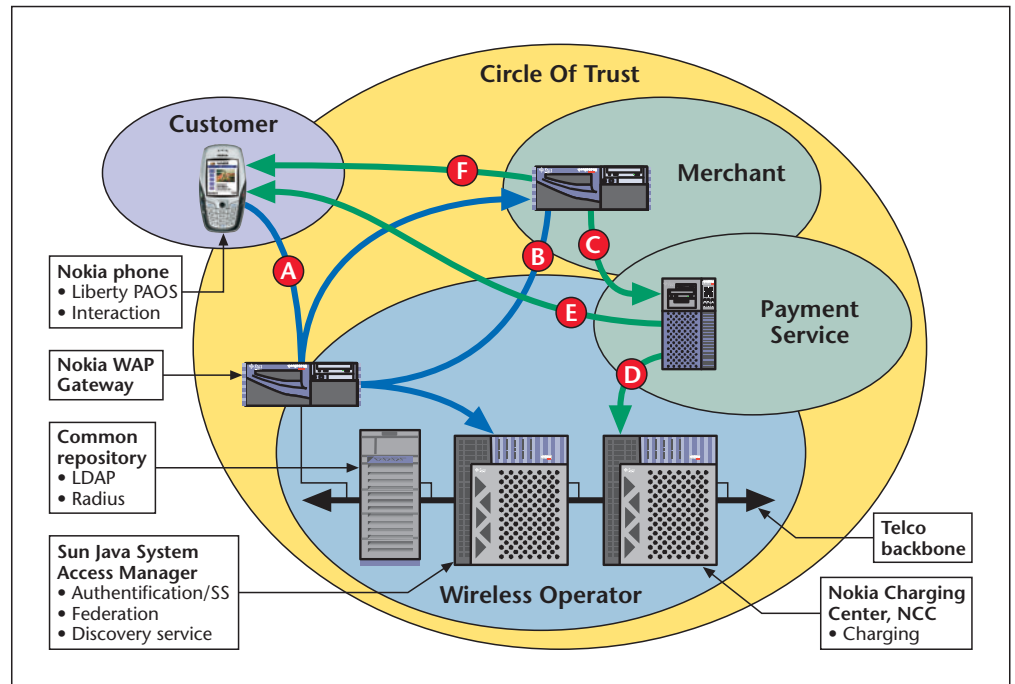
- Customers only have commercial contracts with their wireless operators
- Merchants may sign agreements with a few providers of payment services, but probably not with many. For example, merchants typically have agreements with only a few credit card providers.
- Payment services might – at least for micro payments – be handled directly by the operator, or be delegated to a trusted partner; for example, “Orange UK” trusting “Orange France” for roaming users.

The following image describes one possible browsing use case for payment using the Liberty framework built on Sun Java System Access Manager and the Nokia WAP Gateway. In this example, the Liberty value propositions are:

- The customer does not have to trust the merchant.
- The operator does not have to reveal the customer name or phone number.
- The payment service can either be an embedded service from the operator or an external trusted partner. In the latter case, knowledge of the customer will be limited to a pseudonym.

Non-browsing use cases are explained in the chapters “Developing applications for Liberty identity based web services – some service examples” and “The Nokia Web Services Framework”, later in this White Paper.

Figure 3.



### Simplified sequence flow

- A The customer is browsing the merchant site and initiates a purchase of some item using their wireless handset. Most wireless operators will handle this request through a gateway though this is not required. The Liberty conformant Nokia WAP Gateway adds LECP (Liberty Enabled Client Profile), optimizes the wireless protocols, handles white/black list and provides other necessary support features. At this point the merchant site may create a browsing session for the user, but the user is still not authenticated at this point.
- B The merchant, who needs to authenticate the customer, can request authentication via Sun Java System Access Manager server through the Nokia gateway. In return it receives an SSO authentication token. Note that this is an ID-FF SSO token and thus can mask real customer identity, thus helping to preserve customer privacy.
- C The merchant requests the payment service to guarantee the transaction. First the merchant site discovers the customer payment service, via the Discovery Service. It then requests the payment services on behalf of the principal. This step leverages the fact that the Liberty discovery mechanism is **per principal** oriented, allowing merchants to request payment while not knowing the real customer identity.
- D The payment service connects to the operator charging service to record this transaction. The payment service discovers the operator charging service through the Liberty Discovery Service in order to get a token that will allow it to request charging on behalf of the Principal.
- E Before charging the customer, the payment service needs to get formal user consent. In the simplest use case, consent might be given globally for a set of services within customer profile. In a real deployment, a buying action it is most likely to require interactive consent. Fortunately Liberty ID-WSF supports three models for obtaining user consent. In this scenario any one of the three could be valid, and the choice will depend on business agreement, and customer preferences. Depending on the chosen model, consent can either:

- Go through the merchant site, using it as a proxy for consent request. This model requires trust of the merchant, or forces customer to sign the answer.
  - Redirecting the user to the payment service. This is probably the most typical option for web applications. It is especially valid if the payment service is handled by a well-known and trusted entity [for example: a well known bank, a national wireless operator, etc.].
  - Use an independent **interaction service**. An interaction service allows for out-of-band consent, which is mandatory in the case that an individual is trying to carry out a transaction on behalf of another individual. In this scenario, the payment service discovers the Principal's interaction service from the wireless operator's DS. This can be done via any technique to get user consent such as sending an SMS or a WAP push. (For example, "will you accept recharge to your pre-paid account?"). The consent request is a key element of Liberty ID-WSF. It is important to understand that while consent is requested directly by the identity service that needs it, the actual dialogue with the principal may be delegated to another entity of that has the user's trust.
- F Deliver the service, return the confirmation of purchase, etc.

## An end-to-end view on architecture and its components

There are some general aspects to consider when setting up an end-to-end architecture, such as our described use case:

- **Interoperability:** The implementation has to be interoperable. This can be ensured through participation in the Liberty Alliance conformance program, where successful participants earn the right to use the "Liberty Interoperable" logo. Alternatively, an implementer can use one of several "Liberty Interoperable" products, such as Sun's Java System Access Manager and the Nokia WAP Gateway.
- **Development Tools:** An SDK should be provided so that the developers can focus on the application development and service delivery.
- **Usability:** Ease of use must be a primary driver when designing the services. Identity-based web services are user-focused by nature. Authentication, Discovery Service and other functions are largely automated. Users should experience a "backward-leaning" approach, where the service setup process can be performed with very few clicks. This naturally is especially useful for mobile users where input limitations otherwise would be a showstopper for service consumption.
- **Scalability:** It is important to consider an open and extensible implementation architecture. For example, planning the end-to-end architecture to support different access networks and devices – even if multiple access methods are not initially provided – in order to increase addressable market and to further improve the chances for wide adoption.
- Another approach is to start small – and **then** broaden the scope. For example, an implementation could start with basic ID-FF and a small Circle of Trust and include ID-WSF functionality at a later stage. The opposite approach is also possible: start with the ID-WSF, and then expand the system by including ID-FF functionality. Both approaches are valid and supported in the ID-FF and ID-WSF specifications.

## Developing applications for Liberty identity based web services – some service examples

A good example of a web service implementation based on ID-WSF is the Radio@AOL service. The service has been launched for AOL's broadband customers and a pilot has also been implemented where the user can access the service with a Nokia 6600 or 6620 smartphone. The picture illustrates this mobile implementation, and suggests how the Nokia web services architecture can help to ease the development of services.

Figure 4.

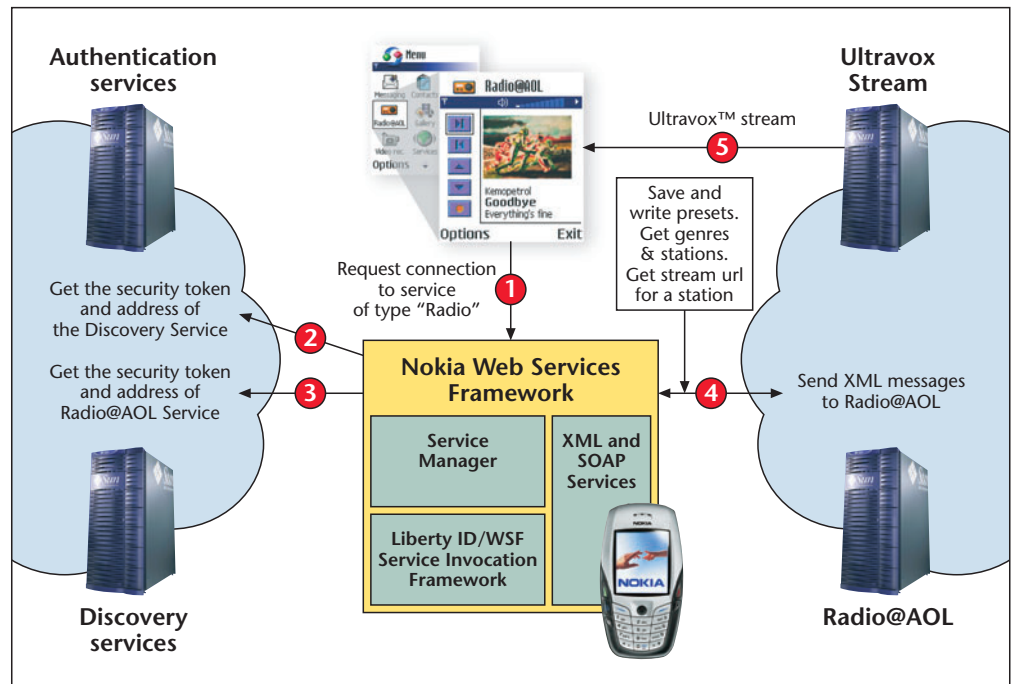


Figure 5. User view

The actual service content can be proprietary, but the setup procedure (steps 1–3) is open and based on ID-WSF. The Radio@AOL service is only one example of what can be done on top of a product such as Sun Java System Access Manager, which supports ID-WSF. Similar service implementations can easily be made, for instance for picture sharing, location-based applications, or other services. The first service implementation will typically prove to be the most challenging. The learning curve is fast and additional services can easily be added. The mobile pilot of the Radio@AOL service took just 6 weeks to develop and deploy – a good proof-point for the Liberty specifications.



## The Nokia Web Services Framework

ID-WSF support is built into the Nokia terminal web services programming platform. The platform hides all of the complex details of ID-WSF protocol interaction from the application so the developer can concentrate on the actual service client logic. Typically, the client application interacts with the platform as follows:

```
serviceHandle = getServiceSession("ServiceURI");  
submit(serviceHandle, "XML Request", response);
```

As shown in the pseudo-code the client program asks for a service handle based on the Uniform Resource Identifier (URI) of the service. The WS platform then carries out the required ID-WSF protocol interactions with authentication and discovery services to create a valid application session with the requested service. Once the application has a valid service handle, it can use this to send and receive service-specific XML messages.

The WS platform caches service handles so that it can shortcut framework protocol interactions as much as possible. For example, the platform may have cached a service handle that only requires re-authentication in order to be valid.

A key principal of the Nokia WS platform design is that the application should only have to handle the XML messages needed to interact with the service itself since these are what embody the service logic. The WS platform takes care of the authentication and discovery steps and wraps the service-specific application messages as necessary for communication with the service.

One of the major benefits of separating the application service logic from the underlying ID-WSF framework protocols is that the application becomes framework- and protocol-independent. The Nokia WS architecture is designed to allow service frameworks other than ID-WSF to plug in. So, for example, the same application logic can be used both for an ID-WSF implementation and for a "vanilla" basic web service implementation.

Developers will also be able to use upcoming web services development tools that allow them to interact with the Nokia WS platform in a more traditional way: by compiling a WSDL service interface description to automatically generate application stub code. In this case, the application logic deals with the service interface as a set of Java classes and methods rather than via the underlying XML service messages. The automatically generated stubs take care of generating and parsing XML messages and communicating them with the service. This style of programming can make it easier to create RPC-style web services where the application sees the service interface as a set of operations with parameters and return types rather than as an XML document.

# Sun's Liberty-ready identity solutions – a developers view

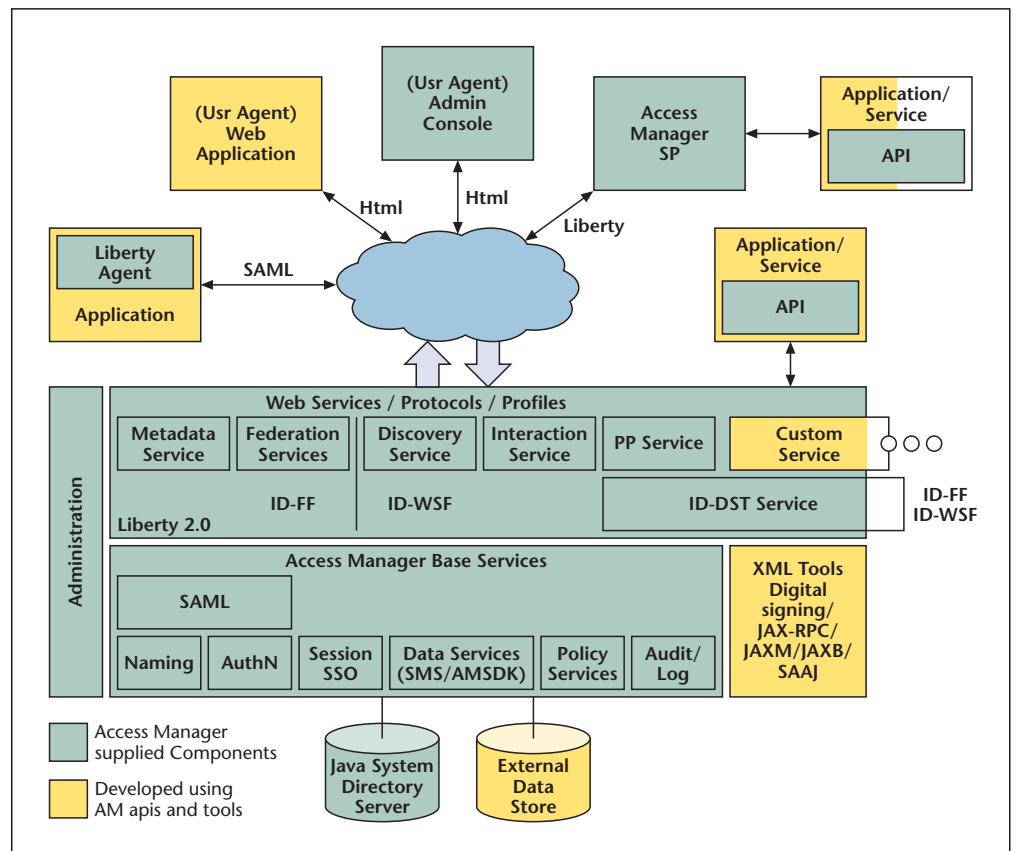
Sun Java System Access Manager is a standards-based product designed to help manage secure access to a company's web applications – both within the enterprise and across business networks. Access Manager is a security foundation that provides web single sign-on, role- and rule-based access control for centralized authentication and policy-based authorization with a single, unified framework.

Access Manager leads the industry in providing the first fully productized support for the latest federation standards including the Liberty ID-FF and ID-WSF specifications, as well as the SAML 1.1 protocols. Based on the Java 2 Platform, Enterprise Edition (J2EE) architecture, Access Manager has been certified as "Liberty Interoperable" and provides a secure, open and scalable foundation for federated identity and web service implementations.

As it pertains to the context of this white paper, Access Manager allows for simplified implementation of ID-WSF through bundled functionality including a Liberty Discovery Service, SOAP bindings, a Data Services Template, a Liberty Interaction Service, and PAOS (reverse HTTP binding for SOAP messages). The product provides tools and APIs for two distinct classes of developers: (1) Identity Web Service implementors (WSPs), and (2) application developers consuming these web services (WSCs). Furthermore, Sun recognizes that most Liberty deployments will need to be done on top of existing systems and legacy services. As a result, Access Manager provides the necessary hooks to quickly integrate these services into a Liberty-enabled infrastructure.

## Architecture

Figure 6.



## Discovery Service

Sun Java System Access Manager's Discovery Service comes integrated with a policy framework for access authorization and features a remote Client API, a pluggable Authorizer SPI as well as a pluggable SPI to retrieve users' resource offerings. Access Manager enables configuration of users' resource offerings at the User, Role or Organization level and provides console support to create and modify the resource offerings.

Common URIs handled by Access Manager's Discovery Service, include a ServiceType element and a ResourceID element. The ServiceType element is used to identify a service type. This URI needs be constant across all implementations of a service to enable interoperability.

The ResourceID element contains a URI used to identify a particular resource. Resource identifiers can be registered with the discovery service; queries on the discovery service return the resource identifiers of matching resources, along with a service instance description that describes how to access the resource. The format of resource IDs will vary from service instance to service instance. Clients do not have to – and should never – construct resource IDs since clients obtain them from the discovery service.

Sun's Access Manager implementation enables users to create specific ResourceID at the ResourceID mapper. This mechanism supports the implementation of generic services that are mapped, not on a per principal basis, but on a group of principals basis. This is key to quickly Liberty-enabling large-scale legacy systems that would otherwise take significantly longer while Discovery is populated for each user.

## SOAP Binding & Data Services template

As part of Sun's goal of making Access Manager as powerful as possible for developers, the product provides APIs that abstract developers from SOAP message parsing and insulates them from dealing directly with the security headers that accompany all requests – thereby allowing the use of a high level interface to handle requests. The core framework handles all the heavy lifting in parsing SOAP messages and verifying and generating security header information. Key related features include a set of Java APIs to send and receive ID-\* messages using SOAP and XML, as well as Java APIs for Messages processing. Access Manager also provides a Request Handler SPI as an extension point for customer-facing web services.

A Data Services Template is the preferred interface to implement new services as it provides developers with a direct interface to the Liberty protocols, allowing an interaction with the discovery service, ResourceID mapper, and other functionality. Access Manager's Data Services Template provides common utilities for message error checking and verification, a remote client API to support customer data service instance, and an abstract handler that can be used as extension point for any data service instance.

APIs are provided for WSC developers to authenticate and to securely invoke liberty web services in a circle of trust.

## Interaction Service

In order to facilitate interactivity obtaining Principal consent (as described in this white paper's merchant use case) Sun provides an Interaction Service in Access Manager. Key features of this Interaction Service include a protocol for simple interaction of an ID-WSF participant with a Principal, an interaction profile based on Support Redirect Request, and embedded support for HTML and WML.

Integration with the policy framework allows for fine-grained control on conditions for interacting with a user, such as attributes of a given service for specific clients.

## PAOS (reverse HTTP bindings)

Access Manager also supports reversed HTTP binding for SOAP messages, also known as 'PAOS'. The primary difference from the normal HTTP binding for SOAP is that here a SOAP request is bound to a HTTP response and vice versa. Access Manager provides PAOS APIs to process PAOS messages.

Access Manager's PAOS functionality enables deployment of ID-WSF services to a client, such as a mobile phone, without having to run a web server on the client

## Getting Started With Sun Java System Access Manager

To facilitate the development and implementation of identity-based solutions and services, Access Manager provides several examples shipped in source code, including implementations of the Employee Profile Service, PAOS and Web Services Consumer.

## Why you should care

### The (Java) developer view

There are several reasons why it makes sense to develop and deploy services on top of Liberty's ID-WSF.

1. **Make an early market footprint:** By making early services available for early adopters you get a time advantage in your learning curve. Your business potential increases through early positioning.
2. **The Liberty Alliance specifications reflect high quality and broad industry support:** The Liberty Alliance develops its specifications in a flexible and efficient manner. They are written by world-class collective intelligence in the Identity Management and web services arena. Open service interfaces (ID-SIS) will be added to the specifications to help developers and vendors offer interoperable services. Leading vendors and key end-user companies from several industries support and participate in Liberty Alliance activities which helps the development cycle to be highly accurate from requirements to market-ready specifications. Liberty also provides conformance testing of products, which helps to ensure interoperability.
3. **Security, integrity and usability:** Liberty's approach is to keep the user's privacy in focus for all its specifications, thus providing user trust in services. This helps to speed up the adoption, along with the easy access and usage of the services that Liberty-enabled solutions offer.

## Liberty Alliance participation

Should your company be interested in directly impacting the future of the Liberty specifications, the Alliance has multiple levels of membership available. Liberty members not only have the opportunity to define and maintain the core Liberty specifications, but also can participate in valuable marketing, business guideline and public policy activities.

## Useful web services links

- Liberty Alliance website: [www.projectliberty.org](http://www.projectliberty.org)
- Nokia Developer organization: [www.forum.nokia.com](http://www.forum.nokia.com)
- SUN developer organization: [www.developers.sun.com](http://www.developers.sun.com)
- SAML (SSTC) and UDDI specifications: [www.oasis-open.org](http://www.oasis-open.org)
- SOAP and XML specifications: [www.w3c.org](http://www.w3c.org)

## About Nokia

Nokia is the world leader in mobile communications, driving the growth and sustainability of the broader mobility industry. Nokia is dedicated to enhancing people's lives and productivity by providing easy-to-use and secure products like mobile phones, and solutions for imaging, games, media, mobile network operators and businesses. Nokia is a broadly held company with listings on five major exchanges.

## About Sun Microsystems

Since its inception in 1982, a singular vision – “The Network Is The Computer” – has propelled Sun Microsystems, Inc. (Nasdaq: SUNW) to its position as a leading provider of industrial-strength hardware, software and services that make the Net work. Sun can be found in more than 100 countries and on the World Wide Web at <http://sun.com>.

The contents of this document are copyright © 2004 Nokia and copyright © 2004 Sun Microsystems, Inc. All rights reserved. A license is hereby granted to download and print a copy of this document for personal use only. No other license to any other intellectual property rights is granted herein. Unless expressly permitted herein, reproduction, transfer, distribution or storage of part or all of the contents in any form without the respective prior written permission of Nokia or Sun Microsystems is prohibited.

The content of this document is provided “as is”, without warranties of any kind with regards its accuracy or reliability, and specifically excluding all implied warranties, for example of merchantability, fitness for purpose, title and noninfringement. In no event shall Nokia or Sun Microsystems be liable for any special, indirect or consequential damages, or any damages whatsoever resulting from loss of use, data or profits, arising out of or in connection with the use of the document. Nokia and Sun Microsystems reserve the right to revise the document or withdraw it at any time without prior notice.

This distribution may include materials developed by third parties.

Nokia and Nokia Connecting People are registered trademarks of Nokia Corporation. Nokia product names are either trademarks or registered trademarks of Nokia. Other product and company names mentioned herein may be trademarks or trade names of their respective owners.

Sun, Sun Microsystems, the Sun logo, Java, SunTone, Sun™ ONE, The Network is the Computer, We're the dot in .com, iForce, Java System Access Manager, Java System Directory Server and Powered by Sun are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

NOKIA CORPORATION  
Technology Platforms  
P.O. Box 300  
FIN-00045 NOKIA GROUP, Finland  
Phone: +358 (0) 7180 08000  
[www.nokia.com](http://www.nokia.com)

SUN MICROSYSTEMS, INC.  
4150 Network Circle  
Santa Clara, CA 95054 USA  
Phone: 1-650-960-1300  
or 1-800-555-9SUN  
Web: <http://sun.com>

**NOKIA**  
CONNECTING PEOPLE

