Draft 0.7, please comment through the wiki at www.yadis.org!

YADIS –
Yet Another Decentralized Identity Interoperability System

The YADIS Project
www.yadis.org

1. Background

In early 2005, NetMesh published the Light-Weight Digital Identity (LID) specification for decentralized, URL-based personal digital identifiers. Shortly thereafter, Six Apart published the OpenID specification for authenticated blog comments using blog URLs as identifiers. (see http://lid.netmesh.org/ and http://openid.net/) These two personal digital identity systems are currently being used by well over fifteen-million users world-wide.

The lead developers of these two initiatives (Brad Fitzpatrick and David Recordon of Livejournal/Six Apart, and Johannes Ernst of NetMesh) quickly realized the complementary nature of their technologies. Over a few weeks in summer 2005, they developed a design to make LID and OpenID interoperable, and to leverage each protocol's most compelling features with each other.

After the YADIS session at the October 2005 Internet Identity Workshop, the XRI folks working on i-Names joined the effort as well. YADIS is applicable to any URL-based identity system and by no means tied to OpenID, LID, or XRI.

Working on this, it became clear very quickly that the resulting interoperability architecture was much more broadly applicable. In our view, it promises to be a good foundation for decentralized, bottom-up interoperability of a whole range of personal digital identity and related technologies, without requiring complex technology, such as SOAP or WS-*. Due to its simplicity and openness, we hope that it will be useful for many projects who need identification, authentication, authorization and related capabilities.

This document describes the base YADIS protocol, and outlines how to use it together with LID and OpenID. For how to get involved, see the last section of this document. This document is largely still a work in progress, proposing how different existing identity systems can work together; feedback is welcomed. The codename, 'YADIS,' is not designed to be user facing and is expected to be changed as this project further progresses.

After this document is released in Version 1.0, formal specifications of the YADIS Capability Discovery Protocol and the YADIS Capability Document will be prepared and published.

2 Goals

The goals of YADIS are:

to further broaden the applicability and feature set of OpenID, LID, XRI/i-names and of other personal digital identity technologies: not by creating more fully-featured stovepipe technologies, not by expecting the world to conform to specs under control of a single vendor, but by creating an interoperable foundation around which many can innovate.

to reduce the fragmentation and to improve the interoperability of today's digital identity technologies; this includes to help reduce the number of passwords the typical Internet user has to manage today.

to provide a unified experience for users who wish to assert their personal digital identity on the Internet, regardless of the underlying technical plumbing.

to contribute a multi-vendor, multi-technology personal identity foundation for Web 2.0.

to make identification and authentication easier on the Internet, without compromising privacy.

to follow an open, meritocratic process for doing so, e.g. by following common open source development practices.

to allow and foster innovation and competition within the personal digital identity market.

to provide a foundation that current personal digital identity systems can build upon as to not discard their previous work or users.

YADIS' initial focus is to empower the individual user with user-centric personal digital identity, and not so much to serve the needs of enterprises for, say, enforcing compliance with government regulations. While there are successful uses of the described technologies in enterprises already, we realize that more work needs to be done to address additional enterprise requirements. If you have specific expertise in this area, we very much appreciate your input. We do however see the ability for corporations to integrate their existing authentication mechanisms with other YADIS enabled services, providing their users with SSO abilities outside of their own architecture.

3 Architectural Assumptions

We have found it is easiest to understand an architecture if it explicitly lists its assumptions; so here you are.

3.1 Fully decentralized, and no one point of control

The Internet is a big place, in which centralized control of any kind is very difficult or impossible. While the technology is quite simple in the case of ICANN, for example, the checkered history of ICANN can serve as the proverbial Exhibit A for this conjecture. On the reverse, where certain companies have been successful in establishing a technical or organizational choke hold on the Internet, innovation these days tends to route around that.

We believe any digital identity architecture must take these lessons learned and not introduce any additional centralized bottleneck, whether of a technical or of a governance nature, if there is a way of avoiding it. LID and OpenID have both demonstrated that this is possible.

3.2 Let many (interoperable) flowers bloom

We firmly believe that innovation is a good thing and want to enable people around the world to innovate upon this interoperable infrastructure, instead of declaring we have all the answers already and making ourselves the bottleneck for innovation.

We believe that digital identity technologies today are only the tip of the iceberg, and, while growing, the market is only embryonic today. For example, so far we have seen little public debate on the merits and issues of personal digital identity technologies; we can bet that such a debate will occur and that it will have substantial impact on what technologies will be broadly accepted and which won't. So we feel it is paramount to let people with good ideas plant new flowers and let those flowers bloom. While there may be a point that a single identity system reaches critical mass on its own, providing a foundation for interoperability will only decrease the amount of time before the general public understands and uses digital identity systems.

YADIS supports the introduction of new capabilities by anybody, while providing enough of a foundation to not break interoperability.

3.3 URLs as identifiers

It is a natural expectation of (non-technical) users that that they can employ search engines such as Google to find people, e.g. by searching for the first and last name of the person, company name etc.

Today, search engines most likely find a person's blog or home page (if they have one) first. Therefore, we believe using URLs (such as blog or homepage URLs) as identifiers for people is A Good Thing.

It is possible to extend the YADIS architecture to work with non-URL identifiers as well. This draft begins the work of integration with XRIs / i-names. We intend to integrate with other, non-URL-based personal digital identity technologies. We do however feel that reaching critical mass will be obtained first upon the assumption of the use of URLs as personal identifiers.

3.4 RESTful and easy to use for developers

Digital identity technologies can only live up to their full potential if it is really easy for developers of all kinds – from hobbyists running, say, a PTA's discussion board, through open source projects to large commercial firms – to identity-enable their projects. Thus it is paramount to make and keep YADIS as simple as possible.

LID and OpenID implementations exist already in many common programming environments (e.g. PHP, Perl, Java) and can be incorporated easily into existing applications. The same will shortly be true for YADIS. For example, YADIS does not require SOAP or a web services stack.

4 User scenarios

Currently, YADIS defines only one scenario performed by the end user. Identity Services that take advantage of YADIS support many additional scenarios. YADIS Users can take advantage of these other capabilities by virtue of the integration of the capabilities with YADIS. These other capabilities are specified by the Identity Services, not by YADIS.

4.1 Scenario: Authentication at website

The User encounters a website (called a 'Relying Party', see terminology section below) that allows or requires the user to present a YADIS identifier (for example, a LID or OpenID URL, an i-name or other XRI or some other identifier used by a YADIS compliant Identity Service). The user notices this because the website displays a text field titled 'My ID' instead of the classical login pair of text boxes.

The User enters their YADIS ID into the My ID text field and clicks Submit.

The Relying Party determines (using the YADIS Capability Discovery Protocol described below) which YADIS-compatible Identity Services are available for use with that YADIS ID.

The Relying Party chooses an available Identity Service and authenticates the User according to the protocol of that Identity Service.

5. The YADIS Capability Discovery Protocol

5.1 Overview of the YADIS Capability Discovery Protocol

The purpose of the YADIS capability discovery protocol is to enable a Relying Party, which has been proffered a YADIS ID, to obtain a YADIS capability document for that YADIS ID.

5.1.1 Obtaining the YADIS Capability Document

Once the User has entered their YADIS ID into the My ID text field at a website (called the Relying Party), the Relying Party must first discover which capabilities the entered YADIS ID supports, such as whether it is a LID or OpenID URL or an XRI, and which authentication methods it supports.

To do that, the Relying Party makes an HTTP request. This request may take any one of several forms, discussed later in this document.

In response to the request, the Relying Party obtains either:

1. A YADIS capability document.
2. A URL that locates the YADIS capability document.
3. Some other response; this indicates that the entered URL does not support any YADIS identity protocol and is thus invalid from a YADIS perspective.

If the response is 2, the Relying Party uses that URL to obtain the YADIS capability document.

The capability document lists the Identity Services that can authenticate the User.

For a description of the YADIS capability document, see below.

5.1.2 Authentication

YADIS delegates authentication to the Identity Services.

The Relying Party uses the information in the YADIS capability document to choose an Identity Service suitable to its purposes, and uses that Service to authenticate the user.

5.1.3 Other capabilities

Identity Services have other capabilities. These capabilities are identified in the YADIS capability document and each capability operates according to the specification of the particular Identity Service providing that capability.

For example, LID defines a RESTful protocol that allows the structured query of profile information independent of the schema in which that information is expressed.

5.2 Protocol Specification

5.2.1 YADIS ID

A YADIS ID MUST be resolvable to a URL usable with the HTTP protocol. In the remainder of this Section 5.2, this URL is called the 'YADIS URL.'

5.2.3 Alternatives

A Relying Party MAY use the YADIS ID to make an HTTP GET request. This request MAY return an HTML document. If it does, either that document itself or the HTTP response-headers will contain a URL giving the location of the YADIS capability document. The Relying Party then obtains the YADIS capability document using that URL.

The Protocol also includes two alternatives:

The Relying Party MAY first issue an HTTP HEAD request. In that case, the URL MAY be returned in an HTTP header.

The Relying Party MAY include in the HTTP GET request an Accept request-header asking for the YADIS capability document to be returned. In that case the YADIS capability document MAY be returned in response to that request, instead of an HTML document.

The following sections specify the steps of the protocol.

[I will later provide an illustration for each of the alternatives in the form of UML sequence diagrams. –jm]

5.1.2 Initiation

The YADIS protocol is initiated by the Relying Party with an initial HTTP request using the YADIS URL.

This request MAY be either an HTTP GET or an HTTP HEAD request.

A GET request MAY include an HTTP Accept request-header [HTTP 14.1] specifying MIME type, application/xrds+xml.

5.2.3 Response

The response to the initial request MUST comply with the HTTP protocol.
The response MUST one of:

1. An HTML document with a <head> element which includes a <meta> element with http-equiv attribute, X-YADIS-Location

2. An HTML document with HTTP headers which include an X-YADIS-Location response-header

3. HTTP headers only, which MAY include an X-YADIS-Location header

4. A document of MIME type, application/xrds+xml

[Why not require that, if there is an X-YADIS-Location in the HTTP headers, that it also be in the HTML document. That will enable would-be weak Relying Parties to get the X-YADIS-Location. –jm]

5.2.4 Document

If the response is a document of MIME type, application/xrds+xml, that document MUST be a YADIS capability document.

5.2.5 URL

If the response includes an X-YADIS-Location header or an HTML <meta> element with attribute, X-YADIS-Location, the value MUST be an absolute URL. That URL is a YADIS capability document locator; it MUST locate a YADIS capability document. The Relying Party MUST issue an HTTP GET to retrieve that YADIS capability document.

5.2.6 Termination

Unless the Relying Party obtains the YADIS capability document using an HTTP Accept request-header or obtains a YADIS capability document locator using an HTTP HEAD request, the Relying Party must issue an HTTP GET request and examine the response to determine if it contains a YADIS capability document locator in either the HTTP headers or the HTML <head> element and request the document as specified in Section 5.2.4.

If none of the requests are successful and the HTML document does not contain a YADIS capability document locator, then the URL used in the initial request is not a YADIS URL or there has been a failure. [The previous sentence needs to be rewritten to tighten it up.]

6. YADIS Capability Document

6.1 Overview of the YADIS capability document

Note: We considered providing both a text and an XML-based format, in order to make it as easy as possible for clients to use capability information available through YADIS IDs. However, we came to the conclusion that requiring both a text and an XML-based format provides only small additional value, and using an XML-based format enables use of standard parsing tools.

Draft 0.7, please comment through the wiki at www.yadis.org!
We have chosen a simple, extensible XML document called an Extensible Resource Descriptor (abbreviated 'XRD'). The format of XRD documents is being specified by the XRI Technical Committee of OASIS.

The YADIS capability document provides a list of identifiers of Identity Services. These are the Identity Services that know the User identified by the YADIS ID used to obtain the YADIS capability document. In the case of some Identity Services, additional data is included for use by the Relying Party in making a request to the Identity Service.

The capability document also enables the User to specify the Identity Services it prefers be used.

6.2 A simple YADIS capability document

Here is an example of a small YADIS capability document:

```
<?xml version="1.0" encoding="UTF-8"?>
<xrds:XRDS
    xmlns:xrds="xri://$xrds"
    xmlns="xri://$xrd*($v*2.0)">
  <XRD>

    <Service>
      <Type>http://lid.netmesh.org/sso/2.0b5</Type>
    </Service>

    <Service>
      <Type>http://lid.netmesh.org/sso/1.0</Type>
    </Service>

  </XRD>
</xrds:XRDS>
```

This capability document specifies two capabilities.

6.2.1 XRD

The YADIS capability document consists of an XRD container (the XRDS element) with a resource descriptor (the XRD element). The resource descriptor contains a sequence of capability descriptions. Each capability description is a Service element. The sequence of these capability descriptions is not significant.

A YADIS capability document MUST include at least one XRD element.

6.2.2 Service

A YADIS capability document XRD element MAY contain one or more Service elements describing a YADIS capability.

Each YADIS Service element consists of a Type element plus optional elements (we encourage identity protocol designers to be brief).

Note: A YADIS capability document that has no Service elements describing a YADIS capability is permitted.

6.2.3 Type

Each YADIS Service element MUST contain at least one Type element.

Each Type element MUST contain an identifier of some version of some capability of some Identity Service. This capability identifier MUST be a URI or XRI. It MUST identify a single version of that capability.

{If the capability identifier is a URL, the authority part of the URL plus all but the last element of the path part SHOULD identify the service. The last element of the path part SHOULD identify a version of that service. For example, the following element identifies version 1.0 of the LID single sign-on service:

```
<Type>http://lid.netmesh.org/sso/1.0</Type>
```

| <not this paragraph> }

For the purposes of YADIS, version identifiers are not interpreted. There is no assumption whatsoever that support of one version implies support of another. For example, if a version 2.2.1 is supported, there is no implication that version 2.2 is also supported. All versions of the capability that are supported MUST be identified separately.

For each capability identified by a Type element there SHOULD be a capability specification document. It is RECOMMENDED that the capability identifier be resolvable to obtain that capability specification document.

6.2.4 XRDS

The beginning of a YADIS capability document is always the same

```
<?xml version="1.0" encoding="UTF-8"?>
<xrds:XRDS
    xmlns:xrds="xri://$xrds"
    xmlns="xri://$xrd*($v*2.0)"
```

This is followed by:

declarations of any other XML namespaces that will be used

a '>' to close the XRDS element

an XRD element with Service elements

and

```
</xrds:XRDS>
```

to end the XRDS.

6.2 Other parts of a YADIS capability document

Here is an example of a larger YADIS capability document:

```
<?xml version="1.0" encoding="UTF-8"?>
<xrds:XRDS
    xmlns:xrds="xri://$xrds"
    xmlns="xri://$xrd*($v*2.0)">
    xmlns:openid=http://openid.net/xmlns/1.0
    xmlns:typekey=http://www.sixapart.com/typekey/xmlns/1.0
  <XRD>

    <Service priority="10">
      <Type>http://openid.net/signon/1.0</Type>
      <URI>http://www.myopenid.com/server</URI>
      <openid:Delegate>http://jimmy.myopenid.com/</openid:Delegate>
    </Service>

    <Service priority="30">
      <Type>http://openid.net/signon/1.0</Type>
      <URI>http://www.example.com/openid</URI>
      <openid:Delegate>
        http://users.example.com/james.s
      </openid:Delegate>
    </Service>

    <Service priority="50">
      <Type>http://openid.net/signon/1.0</Type>
      <URI>http://www.livejournal.com/openid/server.bml</URI>
      <openid:Delegate>
        http://www.livejournal.com/users/jimmy/
      </openid:Delegate>
    </Service>

    <Service priority="40">
      <Type>http://www.sixapart.com/typekey/sso/1.0</Type>
      <typekey:MemberName>jimmy</typekey:MemberName>
    </Service>

    <Service priority="20">
      <Type>http://lid.netmesh.org/sso/2.0b5</Type>
    </Service>

    <Service>
      <Type>http://lid.netmesh.org/sso/1.0</Type>
    </Service>

  </XRD>
</xrds:XRDS>
```

6.2.1 URI element

A YADIS Service element MAY contain one URI element. The URI element specifies a resource that will provide the service. The URI element is OPTIONAL. A YADIS Service element MUST NOT contain more than one URI element.

The URI element MUST be used when that resource is not the same as the resource identified by the User's YADIS ID. It MUST contain a URI that resolves to a resource providing the service (or services) specified by the Type element(s) of that Service element.

If no URI element is provided, the service(s) identified by the Type element(s) MUST be provided by the resource identified by the YADIS ID.

The URI MUST be an absolute URL, not a relative URL, URL reference, or other URI.

Note: It is our intention that this element may contain any URI or may contain an IRI, as specified by RFC 3987 http://www.ietf.org/rfc/rfc3987.txt . This will be specified in a later version of this specification.

If a Relying Party chooses a Service element, that Relying Party MUST recognize and use the URI element of that Service element.

6.2.2 Priority attribute

The OPTIONAL priority attribute of the Service element allows the User to specify preferences for the Identity Service to be used. The example above indicates that the User prefers the OpenID protocol using the server, `http://www.myopenid.com/server`, and that the last choice is LID version 1.0.

In keeping with the goal of ease of implementation, a Relying Party MAY ignore the priority attribute. A Relying Party that recognizes and uses the priority attribute in one or more Service elements MUST follow the specification of priority attributes of [reference here].

6.2.3 Other elements in a Service element

YADIS does not specify any other elements for a Service element. An Identity Service can define and use other elements as they wish. This allows a Service element to indicate things that are specific to that service. As examples, the OpenID specification defines the optional Delegate element, which specifies a URL by which the OpenID server knows the User; the TypeKey specification defines the MemberName element.

If a Relying Party intends to chose a particular Identity Service, that Relying Party MUST recognize and use all elements specified by that Identity Service for inclusion in a Service element for that Identity Service.

A Relying Party MAY ignore all other elements in a Service element. A Relying Party MAY recognize and use any elements in a Service element, which are specified by the XRD schema.

A Relying Party using a YADIS capability document MAY ignore any element of that document except for the elements specified here and any elements specified by a capability specification. Any Relying Party not using a particular capability MAY ignore any element of YADIS capability document that is specified by the capability specification of that capability. The use of elements specified by a capability specification is determined by that capability specification.

6.2.4 Other elements in an XRD

An XRD in a YADIS capability document may contain Service elements that do not identify YADIS Identity Services. A Relying Party MAY ignore all such Service elements.

A YADIS Identity Service MUST NOT require the use of Service elements that do not identify YADIS Identity Services.

A Relying Party MAY ignore all other elements in an XRD element. A Relying Party MAY recognize and use any elements in an XRD element, which are specified by the XRD schema.

A YADIS Identity Service MUST NOT require the use of any elements of an XRD other than the Service elements that identify the capabilities of that YADIS Identity Service.

6.2.5 Other services

The examples so far have been for single sign-on services, the scenario described in Section 4.1. An Identity Service can have other capabilities, too. For example, the following capability document specifies a number of LID services available at the resource identified by the users YADIS ID:

```
<?xml version="1.0" encoding="UTF-8"?>
<xrds:XRDS
    xmlns:xrds="xri://$xrds"
    xmlns="xri://$xrd*($v*2.0)">
  <XRD>

    <Service priority="10">
      <Type>http://lid.netmesh.org/sso/2.0b5</Type>
    </Service>

    <Service priority="20">
      <Type>http://lid.netmesh.org/sso/1.0</Type>
    </Service>

    <Service>
      <Type>http://lid.netmesh.org/post/sender/2.0b5</Type>
    </Service>

    <Service>
      <Type>http://lid.netmesh.org/post/receiver/2.0b5</Type>
    </Service>

    <Service>
      <Type>http://lid.netmesh.org/relying-party/2.0b5</Type>
    </Service>

    <Service>
      <Type>http://lid.netmesh.org/traversal/2.0b5</Type>
    </Service>

    <Service>
      <Type>http://lid.netmesh.org/format-negotiation/2.0b5</Type>
    </Service>

  </XRD>
</xrds:XRDS>
```

6.2.6 Other XRDs and other elements in the XRDS

The XRDS schema permits more than one XRD in an XRDS and other elements in an XRDS, in addition to XRD elements. Such documents are valid YADIS capability documents.

A Relying Party MUST examine {the first | all} of the XRD elements in an XRDS returned in response to a request for a YADIS capability document.

A Relying Party MAY ignore all other elements in an XRDS.

6.3 Schema for the YADIS capability document

The schemas of the YADIS capability document are the XRDS and XRD schemas contained in working draft 9 of the XRI Resolution 2.0 specification.
[http://www.oasis-open.org/committees/download.php/15310/xri-resolution-V2.0-wd-09.pdf]

Until such time as those schemas may be adopted by OASIS, we copy them here:

[which I will do in due time. – jm]

Should XRDS and XRD schemas not be adopted, we intend to include a smaller schema in the YADIS specification, limited to a subset of the elements specified in working draft 9. We intend to modify the YADIS specification to reference subsequent working drafts and committee drafts of the XRI Resolution 2.0 specification as they are published.

7 Impact on LID and OpenID

OpenID MUST support the YADIS Capability Discovery Protocol.

LID MUST support the YADIS Capability Discovery Protocol as part of MinimumLID.

An OASIS technical committee will consider interoperation of XRI Resolution with sites supporting the YADIS Capability Discovery Protocol.

Going forward, OpenID implementations SHOULD recognize when non-OpenID authentication is requested and respond appropriately. They are encouraged to support LID authentication as well.

Going forward, LID implementations SHOULD recognize when a non-LID authentication is requested and respond appropriately. They are encouraged to support OpenID authentication as well.

8 Examples

[Still to be revised. –jm]

Log on at an OpenID site (non-delegated case)

Action: User enters URL of Relying Party into browser, browser requests page from Relying Party
Response: Relying Party's page is displayed in browser

Action: User enters their YADIS ID (**MYID**, which in this example is an OpenID URL), into the login field, clicks "Submit"
Response: Control gets handed to the Relying Party

Action: Relying Party performs a YADIS capabilities request on **MYID**.
Response: HTML page is returned, which contains **identity.server** field. (This is an OpenID URL)

Action: Relying Party performs a "meta=capabilities" request on the URL obtained from the **IDENTITY.SERVER** field.
Response: Document of type "application/x-meta-identity" is returned, which lists **http://openid.net/** as one of the supported capabilities.

Action: Relying Party interacts with the **IDENTITY.SERVER** (which now has been identified as an OpenID server) as per the OpenID protocol.
Response: The browser session has been authenticated, the Relying Party shows the logged-in page, and control returns to the user.

Log on at a LID site (non-delegated case)

Action: User enters URL of Relying Party into browser, browser requests page from Relying Party
Response: Relying Party's page is displayed in browser

Action: User enters their YADIS ID (**MYID**, which in this case is a LID URL), into the login field, clicks "submit"
Response: Control gets handed to the Relying Party

Action: Relying Party performs a "meta=capabilities" request on **MYID**.
Response: Document of type "application/x-meta-identity" is returned, which lists **http://lid.netmesh.org/** as one of the supported capabilities.

Action: Relying Party interacts with the **MYID** (which now has been identified as an LID URL) as per the LID protocol.
Response: The browser session has been authenticated, the Relying Party shows the logged-in page, and control returns to the user

Terminology

We distinguish between end-user and developer terminology. Developer terminology is intended to be technically precise and unambiguous, while end-user terminology is intended to be easy to understand and to use by non-technical users.

Developer Terminology

Term	***Meaning***
YADIS ID	A URL that is used to identify an entity. A YADIS ID may or may not directly support YADIS Capability Discovery (see Capability Discovery protocol described above). See RESTful YADIS ID, Hosted YADIS ID, and Delegating YADIS ID.
Relying Party	A server, website, or application that can take advantage of YADIS IDs (and/or information accessible through or with them) provided by users. The Relying Party discovers the capabilities of any provided YADIS ID according to the YADIS Capability Discovery protocol defined above, and modifies its own behavior accordingly.
Identity Server	A server that hosts one or more Hosted YADIS IDs. The Identity Server may or may not be located at the same URL as the YADIS ID.
RESTful YADIS ID	A YADIS ID that supports the `meta=capabilities` query for capability discovery. LID URLs are RESTfule YADIS ID.
Hosted YADIS ID	A YADIS ID that does not return the YADIS capabilities document itself, but which instead specifies an Identity Server, but no delegate YADIS ID in the HTML HEAD field of a returned HTML document or in an HTTP header. OpenID URLs are either Hosted YADIS IDs, or Delegating YADIS IDs.
Delegating YADIS ID	A YADIS ID that does not return the YADIS capabilities document itself, but which instead specifies an Identity Server and a delegate YADIS ID in the HTML HEAD field of a returned HTML document or in an HTTP header. OpenID URLs are either Hosted YADIS IDs, or Delegating YADIS IDs.

End-user terminology

Term	*Meaning*
My ID	A URL the user uses to identify an individual, such as themselves or somebody else. Same as YADIS ID.

[Still working on this:
Entity:
entity proffering a YADIS identifier: citizen
URLs
the URL or other identifier: YADIS identifier ; YADIS ID ; (SXIP: persona URL XRI: XRI)
of the capability document: YADIS resource descriptor locator
of the service provider: YADIS service provider locator

Documents
The capability document: YADIS resource descriptor

Roles
requesting capability document: relying party ; requester
responding to initial request: ? ; YADIS capability server ; responder
providing capability document: YADIS resource descriptor server
providing service described in capability document: YADIS service provider ; identity service
]

For more information

Please visit http://yadis.org/ for additional information, including contact information. You can also sign up to the YADIS e-mail list there.

p.s.

[Yup! You are right about the last paragraph of 6.2.4. We can't have that. Just checking to see if folks are reading this. – jm]