Universal Peer to Peer

Community Creation Reference

Alexander Craig
July 15, 2011

Network Management and Artificial Intelligence Lab
ME4447 Carleton University

Contents

1 Introduction

Universal Peer to Peer (UP2P) is a web framework designed for peer to peer filesharing with well defined XML metadata. In order to share files in UP2P a "community" must be defined which details what type of data is being shared, and what resources the system should use to process and display the shared data. This guide is intended for UP2P users who wish to create and share their own community assets. This guide will detail the assets required for a UP2P community, as well as guidelines and special services provided by UP2P to aid asset creation.

2 Prerequisites

Users who wish to create their own communities should be familiar with HTML, CSS, and using XSLT 1.0 to transform XML files. The syntax and usage of these languages is outside the scope of this document. For information on general HTML and CSS see W3C HTML and W3C CSS tutorials. For information on XSLT see the W3C XSLT Tutorial and the XSLT 1.0 Specification. Note that UP2P does not currently support XSLT 2.0.

3 Community Assets Overview

A community in UP2P is a definition of a type of resource type to be shared, along with any attached assets that are needed for the processing or viewing of the shared resources. Community definitions are themselves resources of the "Root Community" and are stored as XML files just as any other resource in the UP2P system. Each community has its own set of attachments which can be accessed anytime an operation is performed on the community. This allows for complex and community specific behaviour. Communities can be generated through the "Root Community" create page, or they can be manually created and directly uploaded to the UP2P node.

3.1 Community Fields

The valid fields in the XML description of a community are listed below along with the XPath to the field. Note that those marked with * are required for all communities:

Title* (/community@title):
: The title of the community which will be displayed to the community users.

Short Name* (/community/name):
: An alternative name for the community with no spaces or special characters. This is used primarily in the internal representation of the community.

Category (/community/category):
A category of communities that the community should fall under (ex. Books, Biology, Sales Listings, etc.).

Description (/community/description):
A text description of the community. This should describe what type of files are shared by the community or what services the community provides.

Keywords (/community/keywords):
A space separated list of keywords that describe the community. This is used primarily to aid searches for new communities.

Title location* (/community/titleLocation):
The XPath that identifies that title of a resource in the community. This will be based on the schema of the files being shared.

3.2 Community Attachments

In addition to the above fields communities support a number of attachments which are listed below. In these cases the filename (prefixed by "file:") should be stored at the XPath listed below, and the attached file should be uploaded as an attachment when the community is created. The only required attachment is the XML schema for the community.

XML Schema* (/community/schemaLocation):
An XML schema (.xsd) file which determines what format the XML resources of the community must match. To generate simple schemas please see the SchemaTool application which is distributed and installed with UP2P (usually at "/SchemaTool//" on the same host as the UP2P node.

Community Display Stylesheet (/community/communityDisplayLocation):
An XSLT stylesheet which will be used to generate the HTML view of local resources.

Resource Display Stylesheet (/community/displayLocation):
An XSLT stylesheet which will be used to generate the HTML view of a specific resource.

Display CSS (/community/displayLocation@style):
A CSS stylesheet to be included when rendering the community and resource viewing pages.

Search Page (/community/searchLocation):
An HTML page which will be displayed to the user when the search function is used in the community.

Search CSS (/community/searchLocation@style):
A CSS stylesheet to be incuded when rendering the search page.

Search Result Stylesheet (/community/resultsLocation):
Advanced: An XSLT stylesheet which can be used to generate custom HTML or XML based on search results. Please see section 5.6 for details on how to use this feature.

Search Result CSS (/community/resultsLocation@style):
A CSS stylesheet to be included when rendering customized search results.

Search Result Javascript (/community/resultsLocation@jscript):
A javascript file that should be included when rendering customized search results. Please see section 5.6 for details on how to use this feature.

Create Page (/community/createLocation):
An HTML page which will be displayed to the user when the create function is used in the community.

Create CSS (/community/createLocation@style):
A CSS stylesheet to be incuded when rendering the create page.

Support Files:
In addition to the above communites also support an arbitrary number of support files. These files can be of any type (ex. Javascript, Java applets, binary data... etc.). The community definition must include a single <supportFiles> element. Within this element, each support file is defined by a <file> element which has children <location> and <description> elements. The location element should contain the filename of the support file prefixed with "file:", and the <description> element should contain a textual description of the support file.

4 Special Variables / Services

UP2P provides a number of special variables and parameters which can be used in community assets to provide flexible processing of resources.

4.1 Accessing Community Assets

In many cases the assets of a community must be able to cross reference each other (ex. including Javascript files to be used in a creation page). To support this UP2P provides a special URL which always maps to the attachment directory of the current community. To access a community attachment with the name "filename" assets should link to **"/up2p/comm_attach/filename"**. Note that CSS stylesheets included in the community definition (not support files) are automatically included in their respective display pages.

4.2 XSLT parameters

UP2P provides three special XSLT parameters which can be used when viewing communites or resources. These parameters are:

$up2p-root-community-id
The community id of the UP2P root community.

$up2p-community-id
: The community id of the current community.

$up2p-resource-id
: The resource id of the resource currently being viewed (only available to the resource display XSLT).

Note that these parameters must be defined in the community XSLT files before the can be accessed. The parameter definition should be as follows:

```
<xsl:param name="up2p-root-community-id" />
<xsl:param name="up2p-community-id" />
<xsl:param name="up2p-resource-id" />
```

4.3 Javascript

To ease the creation of dynamic assets UP2P populates several special Javascript variables when pages are loaded:

root_community_id The community id of the root community.

current_community_id The community id of the current community.

current_resource_id The resource id of the resource currently being viewed (only available within the resource display stylesheet).

In addition to these features UP2P also includes JQuery v1.5 and JQuery UI in all generated HTML, so these libraries can be used freely by user communities.

5 Asset Creation Guidelines

5.1 XML Schema

The first step in defining a community is to create an XML schema for the community data type (i.e. the fields which will describe the data and how they will be organized). This schema must be described in an XML Schema Document format (xsd). Complex hierarchical schemas are not easily supported by the standard U-P2P stylesheets due to the way the resources are created (via HTML forms). For complex types, the construct <xs:all> must be used over the <xs:sequence> construct if the standard U-P2P stylesheets are to be used. Allowing multiple occurrences of an element may not be supported properly. Also note that the default create page does not support specifying attributes, and a custom create page is necessary if attributes should be specified by the user.

The following box shows an example schema for a community sharing snippets of program code:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<xsd:element name="CodeSnippet">
  <xsd:complexType>
    <xsd:all>
    <xsd:element name="Title" type="xsd:string"/>
    <xsd:element name="Language" type="xsd:string"/>
    <xsd:element name="Author" type="xsd:string"/>
    <xsd:element name="Snippet" type="xsd:string"/>
    <xsd:element name="Tags" type="xsd:string"/>
    <xsd:element name="Description" type="xsd:string"/>
    <xsd:element name="License" type="xsd:string"/>
    </xsd:all>
  </xsd:complexType>
</xsd:element>
</xsd:schema>
```

U-P2P provides a SchemaTool wizard which allows for the easy creation of simple XML schemas. See the U-P2P User Guide for further documentation, or access SchemaTool at the URL "/SchemaTool" on any U-P2P node.

5.2 Create Page (HTML)

The create page is responsible for gathering user input to generate a new resource. This input is transmitted in the form of XPath/value pairs to the UP2P node, and the resource file is generated in the server repository. The create CSS stylesheet is automatically included when the create page is displayed. The HTML will be inserted between the <body> tag of an existing HTML page and should therefore not contain <head>, <html> or any other markup not permitted in the <body>. The create page should adhere to the following guidelines:

All the user input should be in a single HTML form element with the following attributes:
action="create"
method="post"
enctype="multipart/form-data"
onsubmit="copyValue()"

The "name" attribute of input elements are passed to the server as XPath parameters. Therefore, each input element in the create page should have a valid XPath as its "name" attribute. When submitted, the value of the input element will be added to the resulting resource XML file at the XPath specified in the "name" parameter.

An input element with the name "up2p:filename" must appear in the form. The value of this input element specifies the filename of the generated resource in the server's repository. If a filename is submitted which does not end in .xml any existing file extension is stripped and replaced with .xml.

Attachments are specified by adding an input element with type="file" to the form. This element must have the name "up2p:filename" to be processed correctly on the server side. **WARNING:** Any attachment elements must appear **after** the first "up2p:filename" input element which determines the filename of the generated resource.

Note: A special parameter name of "up2p:rawxml" is also supported. If a value is submitted with this parameter name the value will be parsed as a complete XML document and any supplied XPath parameters will be ignored. This can be useful for resources which require more specialized input processing than simple XPath/value pairs (ex. in the sample UP2Pedia community resource XML is generated client side with Javascript before being submitted). The "up2p:filename" parameter is still required.

5.3 Search Page (HTML)

The search page is displayed when a resource search is initiated and is responsible for collecting search terms from the user. The search CSS stylesheet is automatically included when the search page is displayed. The HTML will be inserted between the <body> tag of an existing HTML page and should therefore not contain <head>, <html> or any other markup not permitted in the <body>. The search page should adhere to the following guidelines:

All the user input should be in a single HTML form element with the following attributes:
action="search"
method="post"

The search terms are transmitted as XPath/value pairs which will be matched against all resources (local and network) in the community. XPath values should be included in the "name" attribute of any input fields, and the value of the input fields represents the term to search for. If multiple XPath/value pairs are submitted than any returned documents must match all the search terms specified (terms are logically AND'ed together). Blank search terms are not submitted to the server. Only plain text strings are supported (no wildcards), although a single asterisk character '*' is recognized as a "locate all available resources" query.

5.4 Community Display Stylesheet (XSLT)

The community display stylesheet is invoked whenever a user attempts to view the contents of thier local repository. When this occurs an XML string is generated by UP2P which describes the contents of the current community. This string is transformed by the community display stylesheet and the resulting HTML is displayed to the user. The display CSS stylesheet is automatically included in the resulting HTML. The generated HTML will be inserted between the <body> tag of an existing HTML page and should therefore not contain <head>, <html> or any other markup not permitted in the <body>.

The community XML to be transformed has the following format:

The single element called "community" forms the root of the XML tree. The community's title and community id are stored in the attributes "title" and "id" respectively. The root element has a child called "resource" for each resource in the community's local repository. The resource's title and resource id are stored in the attributes "title" and "id" respectively. The XML of the stored resource is included as the child of the "resource" element.

It therefore follows that the community id can always be found at the XPath "/community@id", and the community title at "/community@title". Individual resource ids and titles can be found at "/community/resource@id" and "/community/resource@title". The special parameters described in section 4.2 can also be used for this purpose.

To include a link to view a resource, the link href must be of the format:
view.jsp?up2p:resource=<desired_resource_id>

To include a link to delete a resource the link href must be of the format:
javascript: confirmDelete('view.jsp?up2p:delete=<desired_resource_id>')
An onclick attribute can also be used, with the value of:
confirmDelete('view.jsp?up2p:delete=<desired_resource_id>');

Please see "/up2p/default-community-display.xsl" on any UP2P node for an example of how to generate these links through XSLT.

5.5 Resource Display Stylesheet (XSLT)

The resource display stylesheet is invoked whenever a user attempts to view a specific XML resource. Unlike the community display stylesheet the XML of the resource is passed directly to the XSLT transformation without any manipulation by UP2P. The display CSS stylesheet is automatically included in the resulting HTML. The generated HTML will be inserted between the <body> tag of an existing HTML page and should therefore not contain <head>, <html> or any other markup not permitted in the <body>.

To access the community and resource id from this stylesheet please see the special XSLT parameters described in section 4.2.

5.6 Customized Search Results (XSLT / Javascript)

] Search results can be customized through a combination of XSLT and Javascript. By default, U-P2P provides a dynamic search result viewer which will automatically detect and display metadata fields for most community schemas. If this functionality is insufficient a custom XSLT can be specified which will be used to transform the search result XML before it is read by the client. The input XML format for this stylesheet is identical to the format used by the standard dynamic search result viewer, and can be examined by running a search and manually visiting "/up2p/search". An example is provided below:

```
<results up2pDefault="true">
        <queryParams>
                <queryTerm value="*" xpath="community/@title"/>
        </queryParams>
        <resource>
                <title>Root Community</title>
                <resId>d9643ccbea26fc2545acc76b14190c91</resId>
                <comId>d9643ccbea26fc2545acc76b14190c91</comId>
                <filename>community.xml</filename>
                <sources local="true" number="3">
                        <source peer="134.117.60.69:8080"/>
                        <source peer="134.117.60.64:8080"/>
                        <source peer="134.117.60.83:8080"/>
                </sources>
                <resourceDOM>
                        <field name="name">RootCommunity</field>
                        <field name="category">U-P2P</field>
                        <field name="titleLocation">
                                /community/@title
                        </field>
                </resourceDOM>
        </resource>
        <resource>
                <title>Linked Actors (Images)</title>
                <resId>d789148a26797297107fd9ed55f88ac5</resId>
                <comId>d9643ccbea26fc2545acc76b14190c91</comId>
                <filename>linked_actors_images.xml</filename>
                <sources local="false" number="1">
                        <source peer="134.117.60.64:8080"
                                jaccard="75" netHops="2" neighbours="3"/>
                </sources>
                <resourceDOM>
                        <field name="category">Film</field>
                        <field name="name">LinkedActorsImages</field>
                        <field name="titleLocation">/actor/name</field>
                </resourceDOM>
        </resource>
</results>
```

When a customized search XSL is used UP2P still handles the periodic refreshing of search results through client side javascript. The server is polled every two seconds through AJAX requests, and responds to the client with the results of the custom XSL transform. Upon recieving the data the client side javascript executes an optional callback which can be used to further modify the data before it is displayed to the user. The callback function is passed the complete document object model (either XML or HTML) received from the server, and must return the HTML that should

be displayed to the user. If this callback functionality is desired, the required javascript should be attached to the community in the "Search Result Javascript" field, and must explicitly specify which function should recieve the callback. If no callback function is specified, the received data will be displayed to the user without modification. An example of the required code to setup a callback is provided below:

```
function callbackFunction(data) {
        // ... data manipulation ...
        return data;
}

up2pSearchRefresh.setRefreshCallback(callbackFunction);
```

U-P2P also provides the means to simply add asynchronous download button to search results. To generate a download button the custom XSL/Javascript must generate a button with the class "up2p_search_download" and two attributes called "comId" and "resId" which contain the community and resource ID of the resource to download. Any button which fulfills these requirements will automatically have an "onclick" handler generated which asynchronously launches the download process. An example of the XSL required to generate these buttons is provided below:

```
<!-- Note: Example assumes that the root element for the template is
     a "resource" tag from the standard U-P2P search result XML format -->
<button type="button" class="up2p_search_download">
        <xsl:attribute name="comId">
                <xsl:value-of select="./comId" />
        </xsl:attribute>
        <xsl:attribute name="resId">
                <xsl:value-of select="./resId" />
        </xsl:attribute>
        Download
</button>
```

6 Resource Attachments

Resources in UP2P can have an arbitrary number of attachments. The direct upload form provides fields for uploading resource attachment files. Please see section 5.2 for a description of how attachments should be handled in the create page. To have a resource link to or access its attachments, simply include the filename of the attachment prefixed with "file:" in the XML source of the resource. When this resource is viewed UP2P will parse the "file:" link into the actual URL of the attachment on the UP2P node.