Semantic Mediacasting and Collaborative Feed Sharing

Narendra Kumar Shukla
Dhirubhai Ambani Institute of Information and Communication Technology, India
Tel: +91 79 30510 540
Fax:+91 79 30520 010
narendra_shukla@daiict.ac.in

Anand Agrawal
Dhirubhai Ambani Institute of Information and Communication Technology, India
anand_agrawal@daiict.ac.in

Abstract

This paper presents a novel architecture of a Semantic Mediacasting System describing Mediacast feeds in RDF instead of currently used RSS2.0. We have built a centralized semantic repository which can be queried for subscription and sharing of feeds and also collaborative feed sharing mechanism where a user fetches subscribed feeds and associated media content and shares it with his neighbors, acting as a Pseudo-server.

Keywords:
Mediacsting, Podcasting, Photocasting, Screencasting, RDF, OCS, Semantic Web.

1. Introduction

In the past couple of years, the exchange of information and knowledge sharing over the web in the form of digital multimedia content has seen a lot of development with the arrival of syndication technologies like RSS. These technologies have enabled new means of publishing information on the web such as multimedia blogging, Photocasts and Podcasts, which can be collectively referred as Mediacasts. Mediacasts enable individuals to publish their multimedia content over the web and others to subscribe to content of their interest and access them at a time that is convenient to them in a simple and easy manner. The very simplicity of Mediacasting has resulted in many casual users generating a lot of multimedia content making it available to a wide audience. Also, through mediacasts, users are in effect annotating a lot of content available on the web for others.

There remains no doubt that these mediacasts are revolutionizing the way information is made available on the web. However, this information remains largely unstructured and loosely organized and a lot of effort is being made by people to make the information contained in these mediacasts *structured* where different types of content can be easily recognized and processed by applications and automated search services. One such attempt is to describe mediacasts in RSS 1.0 (RDF Site Summary)[1] based on RDF/XML instead of otherwise popular RSS 2.0 (Really Simple Syndication) which does not support RDF (Resource Description Format) and hence in some ways, does not completely exploit the opportunities being explored by the Semantic Web endeavour.

Another thing that is being observed with mediacasts becoming ever so ubiquitous is a lot of server side congestion with a large number of users wishing to access and download a lot of multimedia content in a short duration of time.

Also, we are yet to see a mass adoption of syndication technologies, which in some ways are a hindrance to the massive rate at which mediacasts are proliferating on the World Wide Web today and a lot of it could be attributed to the lack of a centralized search-and-subscribe mechanism where people can find and subscribe to their feeds of interest and be able to share this with their peers.

Hence, in this paper we propose:

1. The use of metadata to make the mediacasts more structured and organize the information available in a more usable manner.
2. A collaborative feed sharing mechanism where a user fetches subscribed feed and associated media content and shares it with his neighbours, acting as a pseudo server.
3. The use of a semantic repository and a structured feedlist with suitable semantics like OCS (Open Content Syndications) making the whole process of feeds *search-and-subscribe* easier and hence, allowing the users to access their preferred feeds.
4. A feed exchange mechanism where a user downloads the feedlists from the centralized repository and also exchanges these feedlists with their peers.

[1] http://purl.org/rss/1.0/

So keeping this objective in mind, we propose the description of mediacasts in RDF with the use of existing metadata schemas like DC (Dublin Core)[2], FOAF (Friend of a Friend metadata)[3], iCal[4] and store them in a Semantic Repository where people can search for the feeds and then be able to download and exchange their feedlists using structured RDF vocabularies for feedlists. This mediacasts then can be shared amongst the neighbours through pseudo serving thereby resulting in reducing server side congestion to a large extent.

2. Mediacasting

Mediacasting can be referred to as a publish/subscribe model where a content provider publishes certain multimedia content in a feed and the subscriber chooses from the available feeds. The content provider publishes this information through a file available on the Internet. The content provider then has to specify the existence of that file by referencing it into the feed, which is another file in RSS 2.0 format, and then to make it available on the web identified by a feed URI. The subscriber now can 'aggregate' these feeds with the aggregators, which allows the users to periodically receive updates, and downloads the latest episodes to have access to them at their own convenience.
These mediacasts are enabled by the introduction of the <enclosure> element in RSS2.0 that allows <item> to have an enclosure, something like an email having an attachment, thus allowing to describe a multimedia file. The <enclosure> has three attributes, the first, 'url' specifies the actual location of the multimedia file, the second 'length' determines the size of the file in bytes, and the last 'type' describes the MIME3 type of the multi-media file.
Here's what a simple mediacast in an RSS file looks like:

```
<?xml version="1.0"?>
<rss version="2.0">
   <channel>
          <title>Podcasting </title>
          <link>http://example.org</link>
          <description>A sample Mediacast Feed</description>
          <item>
             <title>All about Mediacasting</title>
             <link>http://www.exampleorg/Podcasting/about.html</link>

  <description>Mediacasting explained.</description>
             <enclosure url="http://www.example.org/mediacastt/about.mp3"
                    length="640561 " type="audio/mpeg"/>
          </item>
   </channel>
</rss>
```

2.1 Adding structure to a Mediacast

This above description of a Mediacast in RSS 2.0 format as such does not convey a lot of meaning associated with the information contained in the mediacast as it does not exploit the capabilities of RDF (Resource Description Framework). A lot of work is going on in using RDF to bring order to the existing World Wide Web by presenting information in a more

[2] http://purl.org/dc/elements/1.1/
[3] http://xmlns.com/foaf/0.1/
[4] http://ilrt.org/discovery/2001/06/schemas/ical-full/hybrid.rdf

machine processable form [1] and the same can be extended into introducing structured formats into a mediacast. As already mentioned, with publishing information becoming very easy for individuals in the form of mediacasts, a lot of mediacasts are springing up on the web everyday and the multimedia content that they feature essentially can be classified into certain categories such as audio/video recordings of keynote speeches and presentations at conferences, interviews, reviews, news, and pictures of individuals or groups and so on. A lot of multimedia information is hence put up but is very loosely organized and thus using some existing RDF vocabularies, we can add some metadata to these mediacasts and add meaning to the information contained in the corresponding multimedia files that will enable the sharing, distribution, syndication, and aggregation for this information. For example, a podcast /photocast about a keynote address at a conference should ideally have details of the conference itself, like the conference title, date of the conference, the location of the conference, names of popular attendees, besides the identity information of the speaker and the information about the multimedia content. Similarly, a podcast (audio/video) interview should have information about the Interviewer, interviewee, the topic of interview and the date.

2.2 Mediacast Feed description in RDF

To make mediacasts more structured and in order to attach meaning to the multimedia content they refer to, we have used a proposed Podcast metadata schema and build upon it to include more structured information in the form of mediacast types such as events, conferences, interviews, reviews etc. With this- 3 - metadata added to a mediacast feed, the aggregators and crawlers will be able to retrieve such mediacasts and automatically sort them into various categories for the user and also the users will be able to search for the content they are looking for in a more efficient manner.
For the description of the mediacasts in RSS 1.0, we make use of existing metadata schemas like Dublin Core, FOAF, and a simplified form of a RDF Schema for calendar proposed by RDF Calendar taskforce and a modified schema for Podcast for describing Podcasts in RDF1.0 used in the Podcast PinPointer application [2]. The classes and properties that we use in our specification are detailed here.

rss:channel
This class is an overview of the feed itself, providing a virtual table of contents. It contains the meta-data, which describes the feed itself with title, description, details of people involved and the URL link to the described resource.

rdf:seq
An RDF Seq (sequence) is used to contain all the items that contain the RDF Table of Contents, to denote item order for rendering and reconstruction. In our description of Mediacast, this class would act as the container for the Mediacast:episode listings.

mediacast:Episode
mediacast:Episode is used to describe a single episode in the Mediacast. The episode is described with the help of properties like title, creator and other file attributes.
Mediacast:episode is a subclass of rss:item. Where it contains multiple pod:File entries of the same content, the rdf:about attribute should refer to the default file.

mediacast:File
mediacast:File is used to describe the physical multimedia file. It has the following optional properties that contain the primary metadata entries needed to index and organize media content:

fileSize is the number of bytes of the media object. It is an optional attribute.
type is the standard MIME type of the object. It is an optional attribute.
bitrate is the kilobits per second rate of media. It is an optional attribute.
framerate is the number of frames per second for the media object. It is an optional attribute.
samplingrate is the number of samples per second taken to create the media object. It is expressed in thousands of samples per second (kHz). It is an optional attribute.

duration is the number of seconds the media object plays. It is an optional attribute.
height is the height of the media object. It is an optional attribute.
width is the width of the media object. It is an optional attribute.
lang is the primary language encapsulated in the media object. It is an optional attribute.
mediacast:episodeFile
mediacast:episodeFile is a property of mediacast:Episode referring to a mediacast:File elements contained within a mediacast Episode

foaf:person
foaf:person is used to describe the person. It has the following properties that contain the primary metadata entries needed to describe the individual:
homepage
mbox
img
surname
firstName
weblog

dc:date
A date associated with an event n the life cycle of the resource.

dc:creator
dc:creator refers to an entity primarily responsible for making the content of the resource. A creator may include a person, an organization, or a service.
dc:subject
dc:subject refers to the topic of the content of the resource. Typically, a Subject will be expressed as keywords, key phrases or classification codes that describe a topic of the resource.

rdf:about
rdf:about provides the subject for all the properties of the class it describes and usually refers to the URL of the resource.

rdf:li
rdf:li is a property of the rdf:Seq denoting individual items.

rss:description
rss:description is a brief summary of the content of the rss:channel or podcast:Episode

evt:location
evt:Location has a Geographical location applied to it.

evt:Geo
evt:Geo represents a geographical location. May have a description and/or specify longitude and latitude properties.

evt:dtstart
Specifies when the event, conference begins.

evt:dtend
Specifies when the event, conference ends.

evt:Date-Time
evt:Date-Type is used to identify values that specify a precise calendar date and time of day.

evt:attendee
evt:attendee defines a person with in an event. This person may be a participant at a conference.

evt:role

Specifies the role of a person in the Episode. e.g., he may be the speaker, a participant, interviewer or an interviewee

A sample feed of a structured mediacast is shown in Figure 1. Figure 2 (a higher resolution version can be seen at http://intranet.daiict.ac.in/~narendra_shukla/dc2006/) shows an episode of event type from the Podcast feed shown in Figure 1 rendered as a RDF Graph using IsaViz[5].

```
<?xml version="1.0" encoding="WINDOWS-1252" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:podcast="http://example.com/Podcast#"
xmlns:evt="http://example.com/evt#"
xmlns:foaf="http://xmlns.com/foaf/0.1/"
xmlns:dc="http://purl.org/dc/elements/1.1/"
xmlns="http://purl.org/rss/1.0/">
<channel rdf:about="http:/example.org/Podcasts/index.rdf">
   <title>Podcast Feed</title>
      <description>An example Podcast feed</description>
      <items>
         <rdf:Seq>
            <rdf:li rdf:resource="http://example.org/Podcasts/item1.mp3" />
            <rdf:li rdf:resource="http://example.org/Podcasts/item2.mp3" />
         </rdf:Seq>
   </items>
</channel>
<podcast:episode rdf:about="http://example.org/Podcasts/item1.mp3">
   <dc:creator>Robert S</dc:creator>
    <dc:date>2006-02-01</dc:date>
   <dc:type>Conference</dc:type>
   <dc:subject>Windows Live Conference</dc:subject>
   <title>Windows Live at London</title>
   <description>Bill Gates' Address at Bloggers Conf. London</description>
   <podcast:hasMedia>
      <podcast:File rdf:about="http://example.org/Podcasts/item1.mp3"
            podcast:fileSize="1345873"
            podcast:type="audio/mpeg"
            podcast:bitrate="64"
            podcast:duration="112"/>
      </podcast:hasMedia>
      <evt:attendee>
         <evt:Cal-address rdf:about="mailto:billg@microsoft.com">
            <foaf:mbox>billg@microsoft.com</foaf:mbox>
            <foaf:name>Bill Gates</foaf:name>
            <evt:role>Chairman</evt:role>
         </evt:Cal-address>
      </evt:attendee>
      <evt:attendee>
         <evt:Cal-address rdf:about="mailto:roberts@hotmail.com">
            <foaf:mbox> roberts@hotmail.com </foaf:mbox>
            <foaf:name>Robert S</foaf:name>
            <evt:role>participant</evt:role>
         </evt:Cal-address>
      </evt:attendee>
      <evt:dtstart>
         <evt:Date-time>
            <rdf:value>2004-11-22T15:00:00+00:00Z</rdf:value>
         </evt:Date-time>
      </evt:dtstart>
      <evt:dtend>
      <evt:Date-time>
         <rdf:value>2004-11-22T16:00:00+00:00Z</rdf:value>
      </evt:Date-time>
   </evt:dtend>
   <evt:location>
      <evt:Geo>
         <foaf:name>London</foaf:name>
      </evt:Geo>
   </evt:location>
 </podcast:episode>
<podcast:episode rdf:about="http:/example.org/Podcasts/item2.mp3">
   <dc:creator>Robert S</dc:creator>
   <dc:date>2006-03-01</dc:date>
   <dc:type>Review</dc:type>
```

```
<dc:subject>Book</dc:subject>
<dc:identifier>ISBN:0789012345</dc:identifier>
<dc:title>Anna Kerenina</dc:title>
<title>Anna Kerenina Book Review by Scoble</title>
<description>A truly amazing piece of work… A review of the classic by Tolstoy</description>
<podcast:hasMedia>
  <podcast:File rdf:about="http://example.org/Podcasts/item2.mp3"
    podcast:fileSize="1437452"
    podcast:type="audio/mpeg"
                                                    podcast:bitrate="64"
                                                    podcast:duration="90"/>
                                                </podcast:hasMedia>
</podcast:episode>
</rdf:RDF>
```

5 http://www.w3.org/2001/11/IsaViz/

Figure1: Sample feed of a structured Podcast.

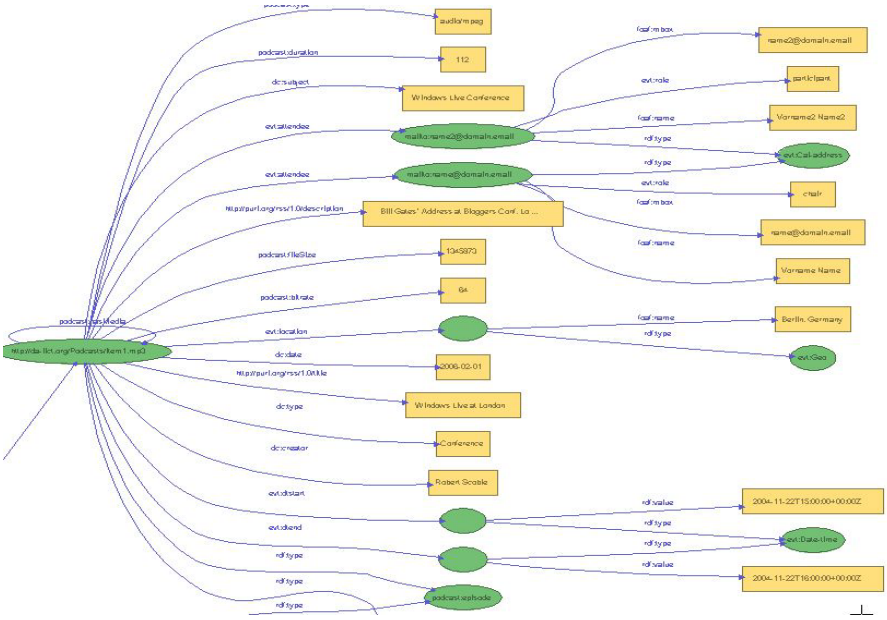

Figure 2: Episode section of the feed when rendered as a RDF Graph for the first episode of type Conference

3. Centralized Search and Subscription

Dave Winer, the designer of RSS 2.0 opines that syndication can bust through with centralized subscription and the ability to search feeds easily [3]. It is with this in mind that we propose a system where people can search for the feed and then subscribe to them. Also we integrate into the architecture a mechanism where people can get a feedlist of a desired subset of the results using the well structured RDF vocabulary for feedlists Open Content Syndication[6] (OCS) Directory Format [4] and are able to exchange this feedlists with their peers.

3.1 Locating Podcasts feed on the web

In order to facilitate the search and subscribe mechanism, we propose to create a centralized repository of all structured mediacast feeds that can be semantically searched. A possible way to achieve a semantic centralized repository is to crawl the web for all such mediacasts described in RDF and then add them to the repository. Another alternate mechanism is to have the mediacast publishers ping the repository whenever there is any update to the Podcast feed.

3.2 Feed Search and Feedlists

With the semantic repository ready, the users are able to search for a Podcast based on a number of parameters such as mediacast format (audio, video, picture), type (event, interviews, reviews), the participants in the event, location of the event, interviewer name, interviewee name and can then subscribe to the result set obtained or a subset of it in the

form of a feedlist.
The feedlist format that we propose to use is Open Content Syndication (OCS) Directory Format [4]. OCS Directory format provides a concise, machine readable-listing of a set of syndicated services. This use of OCS Directory format allows describing the content available in the feedlist. The feedlist generated by the user has all the items selected by the user to be a part of this feedlist under the same channel with information about the channel content and the name of the user as the creator of the feedlist. A sample feedlist generated will be as follows:

```
<?xml version="1.0" ?>
<rdf:RDF
 xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:dc  = "http://purl.org/metadata/dublin_core#"
 xmlns = "http://purl.org/ocs/directory/0.5/#">
   <directory rdf:about="http://InternetAlchemy.org/ocs/ocsdirectory.rdf">
    <dc:title>Poddr- Feedlist for Search Results </dc:title>
   <dc:creator>Robert S</dc:creator>
   <dc:description>Search results Keynote speaker Bill Gates at Mix06 Conference</dc:description>
   <dc:date>2001-04-01T12:00+00:00</dc:date>
   <channels>
     <rdf:Bag>
       <rdf:li rdf:resource="http://www.example.org/podcast/ocs/110101/" />
       <rdf:li rdf:resource="http://www.example.org/podcast/ocs/110102/" />
     </rdf:Bag>
   </channels>
 </directory>
 <channel rdf:about=" http:// www.example.org/podcast/ocs/110101/">
   <dc:title>Feedlist – Podcasts with Bill Gates Keynote at Mix06</dc:title>
   <dc:creator> ABC </dc:creator>
   <dc:description> Search results Keynote speaker Bill Gates at Mix06 Conference </dc:description>
   <formats>
     <rdf:Alt>
       <rdf:li>
         <rdf:Description rdf:about=" http://example.org /Podcasts/index.rdf ">
         </rdf:Description>
       </rdf:li>
     </rdf:Alt>
   </formats>
 </channel>
<channel rdf:about=" http:/ www.example.org /podcast/ocs/110102/">
   <dc:title>Feedlist – Podcasts with Bill Gates Keynote at Mix06</dc:title>
   <dc:creator> XYX</dc:creator>
   <dc:description> Search results Keynote speaker Bill Gates </dc:description>
   <formats>
     <rdf:Alt>
       <rdf:li>
         <rdf:Description rdf:about="http://example.org/Podcasts/fifeed.rdf ">
         </rdf:Description>
       </rdf:li>
     </rdf:Alt>
   </formats>
 </channel>
</rdf:RDF>
```

Figure3: Sample feedlist in an Open Content Syndication Directory Format.

3.3 Sharing Feedlists

The users generating feedlists for their own consumption are not only indexing it for their own use but are also annotating the whole lot of feeds out there. In order to make the process of feed exchange easier, we have included a mechanism where a user who has created a feedlist also shares his content with his peers to facilitate the collaboration between people working in the same project area or sharing the same interests.

For example, a user Maya is interested in feeds for the book review of Anna Kerenina by Leo Tolstoy. She searches the semantic repository and gets a list of all feeds that contain a review of the book Anna Kerenina, as shown in Figure 4. She selects a few feeds from the result set based on the description of the feeds, and downloads a feedlist of the selected feeds.

6 http://internetalchemy.org/ocs/directory/0.5/

Figure 4: Search Results for the Book Review of Anna Kerenina

Now a user Paul also queries the repository for similar search and fetches a feedlist with some of the feeds common to those that Maya had added to her feedlist. The application tells Paul about all users that have added similar feeds to their feedlists, Maya included, and then Paul can look at all their foaf descriptions, and subscribe to their generated feedlists too, thus enabling collaborative feed sharing and social navigation. (See Figure 5)

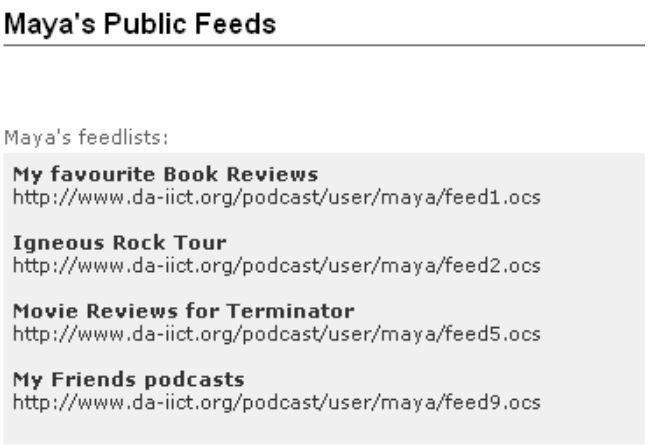

Figure 5: Public Feedlists of a person

4. The architecture

The architecture that we propose to sort out the issues of easy search and syndication, and collaborative resource sharing in the form of Pseudo serving, as shown in Figure 6, comprises of a centralized semantic repository that is periodically updated with newer and updated podcasts feeds updating its database. The users can then search this repository and then can download and subscribe to these search results (or a subset of the result) with the OCS format. The architecture also includes Poddr desktop aggregators where the users can aggregate the Podcast feeds. The desktop aggregators in the architecture also facilitate resource sharing through Pseudo serving amongst the peers.

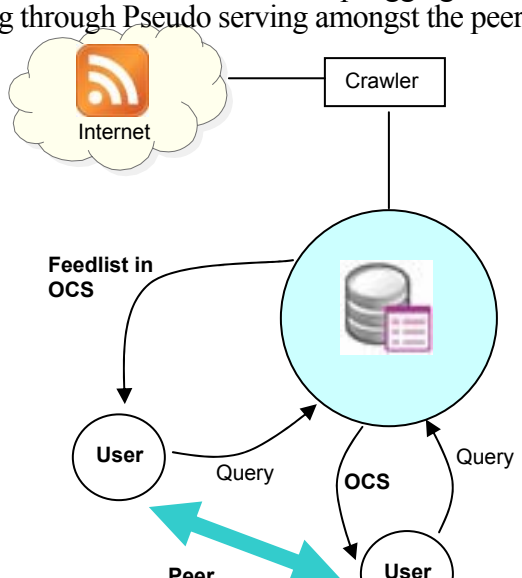

Figure 6: Proposed architecture

The centralized feed database that we have used here is Sesame, an architecture that allows persistent storage of RDF data and schema information and subsequent querying of that information [5]. Since we are looking for a persistent storage of the RDF data contained in the feeds we have the option of a file system or a database management system for it. Once we store all this data into the repository, we need means to query it. The query language for querying the Sesame semantic repository is RDQL. An RDQL consists of a graph pattern, expressed as a list of triple patterns. Each triple pattern is comprised of named variables and RDF values (URIs and literals). An RDQL query can additionally have a set of constraints on the values of those variables, and a list of the variables required in the answer set.

SELECT ?x
WHERE (?x,
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>,
<http://example.com/someType>)

This triple pattern matches all statements in the graph that have predicate http://www.w3.org/1999/02/22-rdf-syntaxns#
type and object http://example.com/someType. The variable "?x" will be bound to the label of the subject resource. All such "x" are returned

Publishing/Searching
This Publishing/searching platform that enables users to publish their own podcasts is built using Struts, a MVC Framework application as shown in Figure 7. The users can specify the different podcast details and create their own Podcast feed and make it available on the web. The users can also make use of the platform to query the centralized feed repository based on different parameters and obtain the feed details. It also converts the results (or a subset of the result) into a downloadable OCS List.

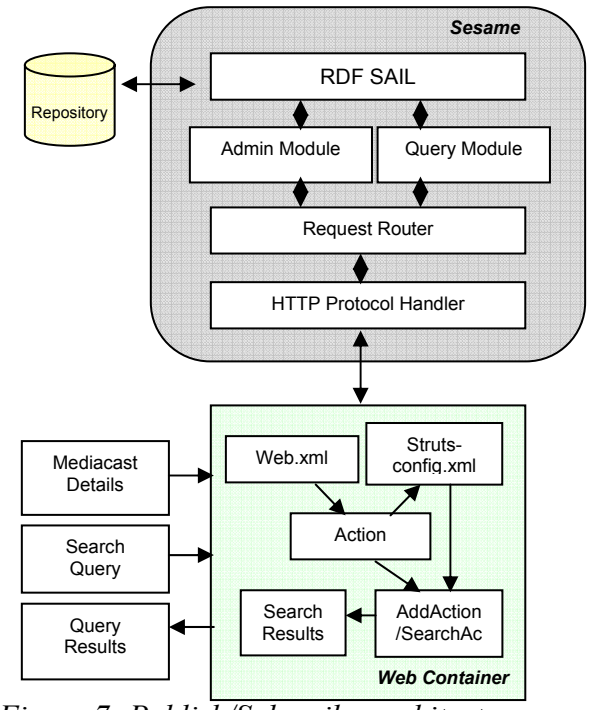

Figure 7: Publish/Subscribe architecture

Aggregation
The Poddr Desktop application is a Semantic mediacast Aggregator where users can

subscribe to the mediacasts of their interests and get their desired multimedia content downloaded to their desktops for their consumption. Along with the multimedia content the user also gets a complete description of the related information like the creator of the mediacast, the type of mediacast, the FOAF description of the people involved and the maps of the locations of the event described in the mediacast.

Every Poddr aggregator is a node of the Pseudo serving mechanism that enables us to tackle the problem of server side congestion. Every time there is a new fetch to be made for a new feed, it checks if any of its neighbours have an updated version and fetches the feed and the corresponding multimedia file from the neighbor instead of the main server. This node is now able to serve the updated version of the feed and multimedia content file to other neighbours.

Collaborative Feed Exchange- Pseudo serving

The Poddr desktop application also makes use of "Pseudo Serving" to tackle the network congestion [6][7]. Here the server agrees to provide information on where the latest feed and the corresponding multimedia files may be immediately obtained. The client, in return agrees to serve the retrieved file to other users. This mechanism in the Mediacast feed exchange can reduce total retrieval times by over an order of magnitude.

A node running Poddr desktop application is connected to its neighbours with a well-known server that provides it a list of its neighbours. Whenever it wishes to refresh a feed, it uses the list-server to get the list of its neighbours who have a more recent version of the feed and the multimedia content associated with the feed, contacts them and retrieves the recent feed. It then notifies the list-server of its updated feed. If it fails to find a more recent feed with any of its peers, it goes to the web server hosting the feed, checks if it has a more recent version available and downloads it. The neighbours can now get a copy of this feed and associated multimedia content from this node.

5. Scenario

5.1 We create a Podcast feed using the Poddr Publisher with an episode describing a Conference taking place at Seattle. The attendees are Richard Stallman and Tim Berners Lee, both as speakers. The publisher generates the feed in RDF and also adds the feed information into the Sesame Repository. (Figure 8)

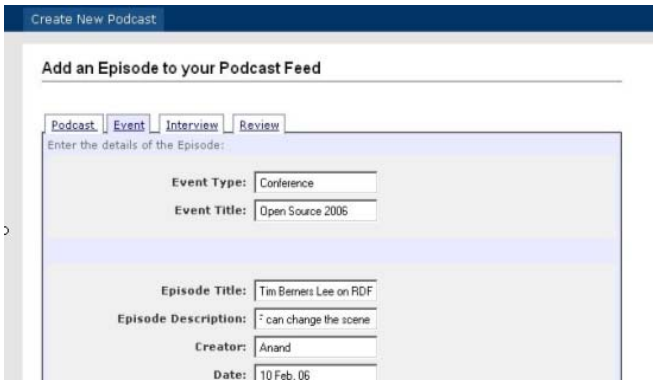

Figure 8: Screenshot – Creating a Podcast feed

A search for all Podcasts for Conferences where Richard Stallman is a speaker with the Poddr Feed Search results in a list of all such Podcasts (Figure 9)

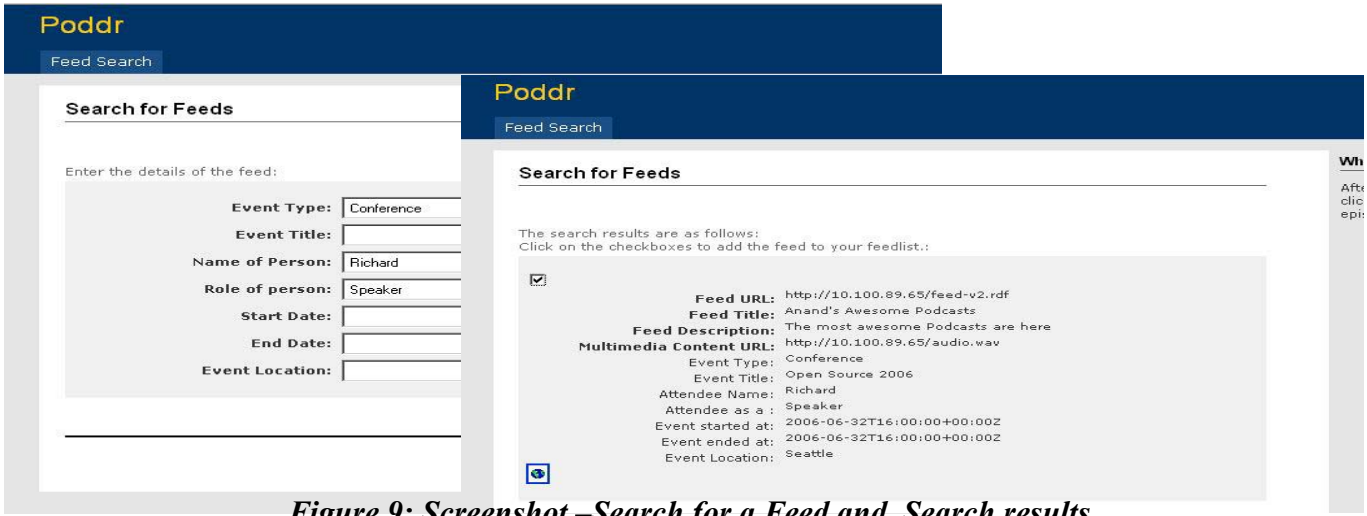

Figure 9: Screenshot –Search for a Feed and Search results

Looking at the results, a user may be interested to subscribe to a couple of feeds. He selects the feeds he wishes to subscribe to and downloads the feedlist on his computer for his use
He then uses his Podcast Aggregator and Imports this feedlist to subscribe to these feeds as shown in Figure 10.

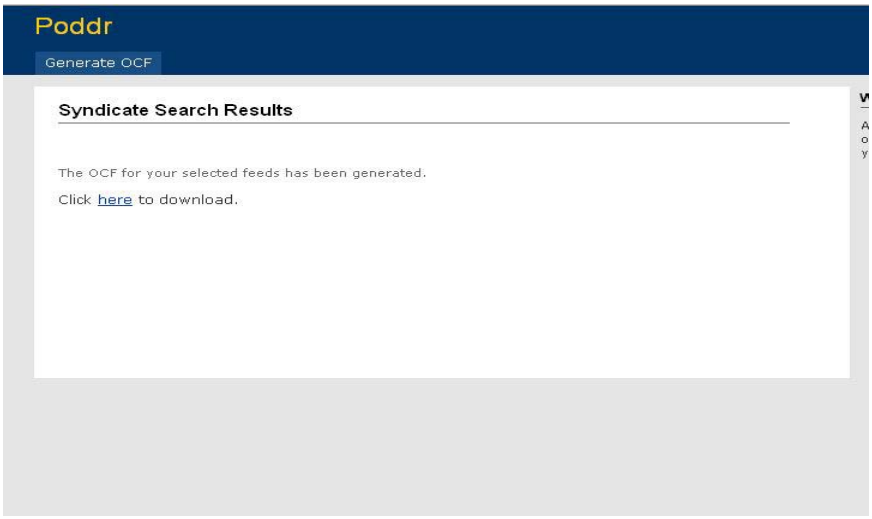

Figure 10: Screenshot – Download feedlist in the form of OCS

Now a user has subscribed to these feeds and can receive regular updates and get all the semantic information that is contained in that feed as shown in Figure 11. For all the feeds, he has access to the foaf description of all people involved in the Podcast, the Mapping facility to locate a place described in the Podcast and results from web services like Amazon for the books and movies reviewed in the Podcast.
Along with the above Semantic Podcast aggregator- Poddr shown in Figure 12, we have also built an extension for the Mozilla Firefox Web Browser (http://www.firefox.com) through which a user can subscribe to the feeds and have access to this metadata (Figure 13).

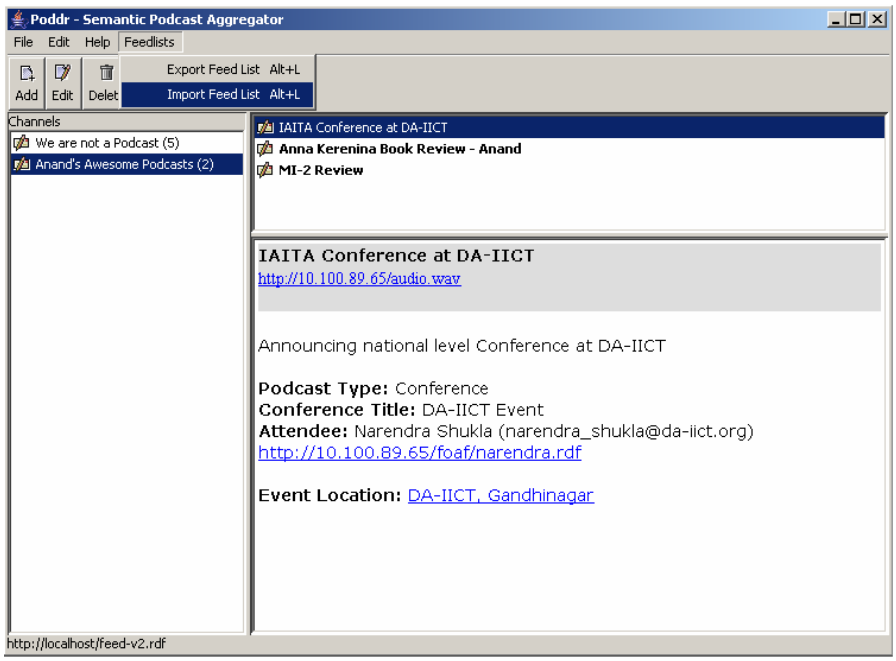

Figure 11: Screenshot – Poddr Aggregator – Import Feedlist

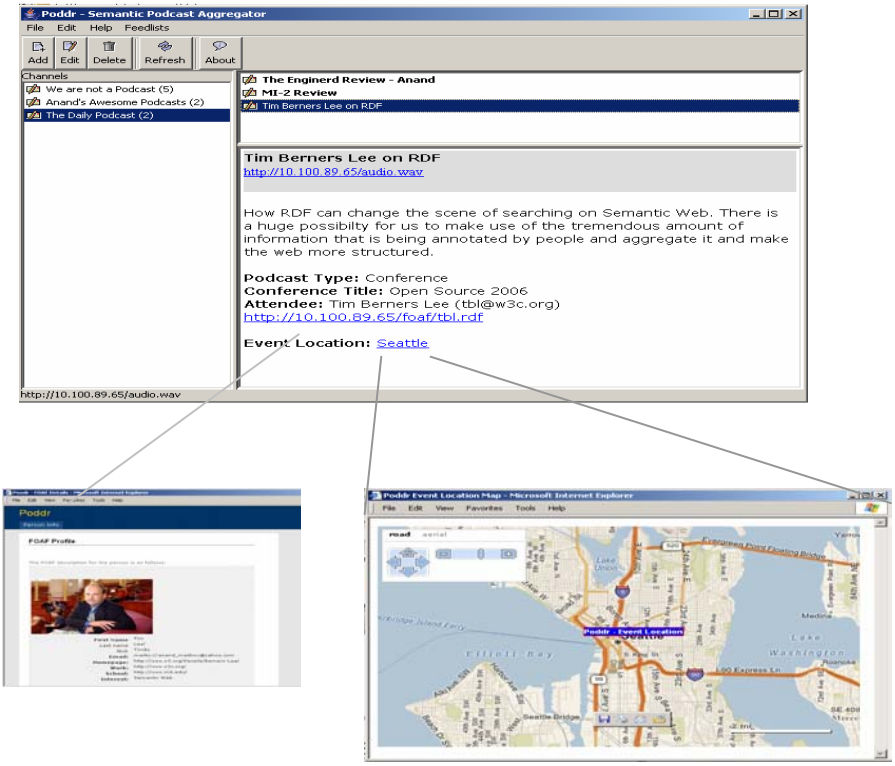

Figure 12: Screenshot – Poddr Desktop Aggregator. The user has access to all Semantic Information

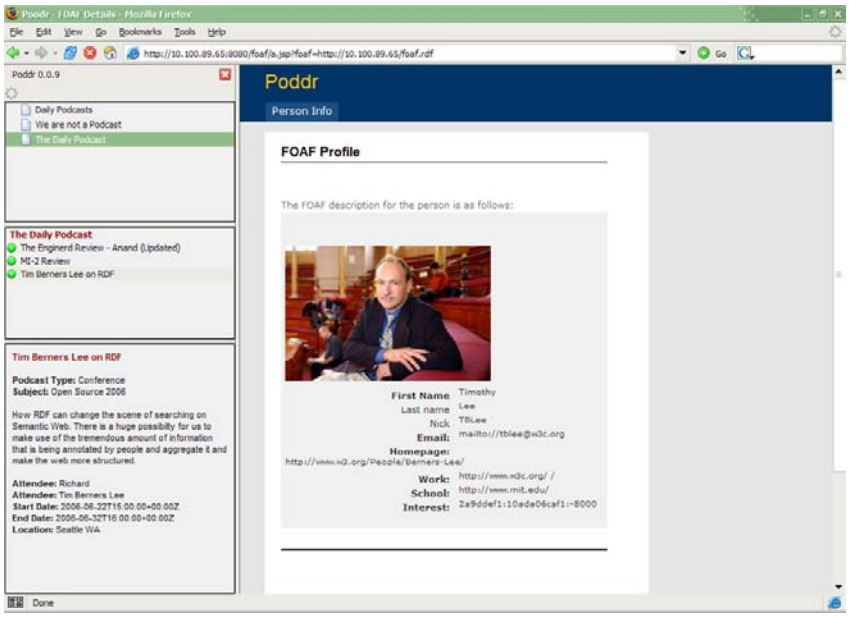

Figure 13: Screenshot – Poddr Firefox Extension

6. Conclusions and Future Work

This paper has discussed a method for adding structure to the Mediacast feeds by describing it in RDF and making use of existing vocabularies like Dublin Core, FOAF, iCal and Multicast. The idea of a centralized Semantic Repository for Multicast feeds has been introduced where users can perform search for feeds of their interest, download and share their subscription lists using Open Content Syndication Directory Format. The Poddr Firefox Extension and Poddr desktop application that has been developed enables the users to semantically query the repository and get their desired feeds, and also get shared feedlists of their peers thereby enabling social navigation.

Also, in order to address the issue of server-side congestion, this may arise when a large

number of users wish to retrieve multimedia files from a single server over a short period of time, we have developed a feed exchange mechanism among neighbours with the concept of Pseudo-Sharing.
We are currently working on the idea to introduce more semantic information in mediacasts feed including more mediacast feed types besides events, interviews and reviews. Also, we look forward to extend the idea of social navigation through peer-peer feed exchange to introduce feed archiving and a recommender system where people will be able to recommend the feeds by rating or reviewing it and also based on a user's subscription

7. References

[1] Lee Tim Berners, "Semantic Web Road Map", http://www.w3.org/DesignIssues/Semantic.html, September, 1998
[2] Harth Andreas, Hogan Aidan, Breslin John G., "Podcast Pinpointer: A Multimedia Sematic Web Application", EWIMT 2005. London.
[3] Winer, Dave, "How RSS Can Bust Through", http://scripting.wordpress.com/2006/02/05/fred-is-right/
[4] Classen Michael, "Open Content Syndication (OCS). Weaving the Web of News", Webreference.com. Jan. 30, 2000.
[5] Jeen Broekstra, Arjohn Kampman and Frank van Harmelen, "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema", Proceedings of the First International Semantic Web Conference on The Semantic Web, p.54-68, June 09-12, 2002
[6] Keith K. and Ghosal D., "Mitigating Server-Side Congestion in the Internet through Pseudoserving", IEEE/ACM Transactions on Networking, Vol. 7, No. 4, August 1999
[7] Jun, S. and Ahamad, M. 2006. FeedEx: collaborative exchange of news feeds. In *Proceedings of the 15th International Conference on World Wide Web* (Edinburgh, Scotland, May 23 - 26, 2006). WWW '06. ACM Press, New York, NY, 113-122