

Extensions to Student Radio Station's Web-based Content

A project proposal by Marc Woolfson

Bailrigg FM,

Fylde College, Lancaster University, Lancaster LA1 4YF

marc.woolfson@bailriggfm.co.uk

ABSTRACT

Bailrigg FM's existing internet-based systems are becoming outdated or unusable by certain end-users. In this paper, the problems and possible resolutions to the existing systems are discussed, and a final solution outlined. The problem of having an inconsistent database system for storing played tracks has a proposed solution, allowing additional features such as playlists to be created. The inability for visitors to the website to contact the on-air DJ also has a proposed solution, as does the broadening of the existing web stream to embed interactive content. This paper also outlines the timescale for the project and splits it up appropriately, according to the various tasks at hand.

Keywords

Internet, HCI, Streaming Media, Databases, Networks, PHP, ASX, MPEG-7.

1 INTRODUCTION

Bailrigg FM is the student radio station of Lancaster University. It has its own web server delivering news and streaming audio content worldwide through radio.lancs.ac.uk. However, some of the existing systems are not being efficiently used such as large data redundancy in one of the databases, and a very basic, but easily extensible web stream. These issues are addressed in this final year project proposal.

Features of the new system include increased ability for web users to contact the on-air DJ; the ability for more accurate charts and playlists to be generated; and the development of a new web (audio) stream.

This report is broken down into six sections, including this introduction. The following section describes the three existing systems, along with their downfalls and possible areas for improvement. Section three describes the proposal itself, and section four shows how time will be split up and spent on the different tasks at hand. Section five analyses the platform and software tools required for the project, and the final section lists the references used in the writing of this document.

2 BACKGROUND

Bailrigg FM has its own web server based within the radio station. The box runs RedHat Linux, holds a collection of PostgreSQL databases, and has PHP Version 4.0 support. It also sends one audio Windows Media™ stream¹ to a 'splitter' in the Computing Department, which subsequently supports up to 1000 individual connections. This intermediate computer is concealed from the end user.

There are currently three main areas for improvement on the Bailrigg FM website. The first is the large amount of data redundancy within the databases held on the radio.lancs.ac.uk web server; the second is the inability for listeners and web users to easily contact the on-air DJ; and the third is the limitations of the current audio stream delivered on the website.

2.1 Databases

There are three databases stored on the radio.lancs.ac.uk server:

- **Library** – containing details about the record library and track logging systems
- **Members** – containing details of members, including show owners and website access levels
- **Website** – containing front-page news and relevant URLs

The main problem lies with the Library database in that there is a massive amount of data redundancy. When CDs arrive at the radio station, they are entered into the database as individual records. These records are only ever accessed by the record library search function (which is limited at that).

When a DJ uses the tracklogging system, (s)he manually types in the artist and track (often incorrectly) – data which may already be stored in the record library database.

Another problem is the playlist system not currently being connected to the record library or tracklogging system at all. This means that data redundancy (as well as time consumption) can occur when the DJ enters an artist and track combination when all they really need is a reference (such as 'A5' for the fifth A-list track). A DJ can also enter a playlist track that has already been recently played – an avoidable downfall.

The structure of the problematic tables in the existing Library database can be seen below.

2.1.1 Tracklogging

¹ via an additional Windows 2000 Media Server

This table holds data on the tracks each DJ has played during their shows. It is an extremely large table with over 5000 records. Each track is entered into the database along with the artist's name, the time, the current user (i.e. the DJ's library card number), and the current user's associated show reference number. The composition of the Tracklogging table can be seen in Table 1.

log_id	primary key (sequencer)
log_artist	artist name
log_track	track being entered
log_link	unused
log_time	time of track being entered
log_user	library card number of current DJ
log_show	show number of current DJ

Table 1: Tracklogging table and descriptions [1]

The current tracklogging system can be seen in Figure 1, and can only be accessed through the members' section (i.e. only authorised users can access it). The artist and track FORM fields correspond to log_artist and log_track in the Tracklogging table respectively.



Figure 1: Top of the current Track Logging system [2]

The log_artist and log_track entities have infinite length within the database. With over 5000 tracks being stored here, growing by a further 100 or so per day means that a large amount of space on the web server is being unnecessarily consumed. The log_link entity is currently unused, but may be suitable to hold the corresponding track reference t_id from the Tracks table, which can be seen in Table 2 in section 2.1.2, below. This would effectively eliminate the need for the log_track and log_artist entities, as these details are already held in the Tracks, and related Artists (Table 3) tables respectively.

A better structured table of tracks being played would make jobs such as compiling weekly or termly charts a lot more simple, as duplicate record library references would be queried, rather than just looking through a long list of tracks and artists (as is currently done). The current system can also lead to discrepancies where personal preference comes into the naming of artists (see Figure 2) or tracks.

Chemical Brothers	The Chemical Brothers	Chemical Brothers, The
-------------------	-----------------------	------------------------

Figure 2: Artist name discrepancies

2.1.2 Record Library

This collection of tables holds track data of CDs received at the radio station. It is quite a comprehensive set of tables that is currently not being used to its full potential.

t_id	primary key (sequencer)
t_name	name of the track
t_length	length of track
t_number	track number in a collection
t_artist	the track's artist's reference (foreign key)

Table 2: Tracks table and descriptions [1]

a_id	primary key (sequencer)
a_name	artist name
a_desc	brief description of artist
a_born	artist's date of birth
a_died	artist's date of decease
a_genre	genre reference (foreign key to genres table)

Table 3: Artists table and descriptions [1]

The actual details of the database structure (i.e. referencing of artists from tracks and vice versa) are somewhat more involved than can be seen above. There is an additional Collections table, holding details on albums or EPs (such as name, genre, type of media, release date) which can also be referenced from the Artists and therefore Tracks tables. This leads to added complexity where intermediate ‘reference tables’ are used for one-to-many or many-to-many (referentially integrated) relationships.

2.1.3 Playlist

This is an empty table present in the Library database available for future work, such as is being proposed in this document. The current playlist system is a simple three page print-out (one page for each of the lists A, B and C), with five columns:

- Reference (i.e. A1, A2,..., B1, B2,..., C1, C2,...)
- Artist (in alphabetical order, thus defining the references $x_1, x_2...$)
- Track
- Release Date
- Comments

The first column is also highlights to show whether a track is a New Entry to the list, or a climber from C to B or B to A lists. The whole playlist is compiled weekly using a spreadsheet system.

2.2 Interaction

Currently, listeners to Bailrigg FM or visitors to the website can only contact the on-air DJ by phoning or sending an e-mail. This can cause two main problems: the on-air DJ being unable to take a phone call because they are too busy or doing a ‘link’ between tracks (i.e. talking into the microphone); and the e-mail system being down (a frequent occurrence due to large amounts of ‘spam’ the address receives) or the DJ again being too busy to check the inbox.

After posting a Web Users’ Questionnaire on the existing websiteⁱⁱ, it could be seen that nearly everyone that has tried to contact the studio has at one time or other run into difficulties. Implementing an online request facility on the website, which would automatically be sent to the online DJ via the track logging system may be a way of aiding this communication lapse.

Another observation from both the Web Users’ and Members’ⁱⁱⁱ Questionnaires was that most users have not sent text (SMS) messages to the studio due to there not being a standard number to send messages to. The current popularity of this form of communication is currently being overlooked, possibly to Bailrigg FM’s disadvantage. The ideal solution to this would be the purchasing of a communications line capable of receiving SMS messages, and then enabling these messages to be passed onto the DJ, again via the track logging system.

2.3 Web Stream

The current Windows Media™ web stream has only been live for a matter of months, and is not being used to its full potential. Figure 3 shows the existing stream as viewed in the latest version of Windows Media™ Player. The only ‘interactive’ element is the BFM Online graphic and associated hyperlink and hyperlink text below the visualisation.



Figure 3: Existing Windows Media™ Stream as viewed in WMP Version 8.0

ⁱⁱ which can be found at <http://www.bailriggfm.co.uk>

ⁱⁱⁱ An alternative questionnaire targeted directly at members and found at <http://www.bailriggfm.co.uk/members>

The stream is spawned through a simple Windows Media™ Audio/Video file which can be seen in Figure 4. The ASX language is fairly simple (with similar syntax to HTML), and highly customisable. It is clear to see from the screenshot above, that the currently playing ‘Song’ for example, could be displayed. This may also allow the extension to enable artists’ names to be clicked upon by the user, and more details obtained (such as their homepage). A more useful extension may be to enable such a system for giving details on the online DJ.

Extending the ASX capabilities would also allow for multiple streams to be broadcast, and the appropriate quality stream delivered depending on the bandwidth capabilities of the client. For example, a home user connecting at 56kbps would receive a lower quality stream via the MMS protocol than a LAN user on a 100Mbps connection.

```

<asx version = "3.0">
  <entry>
    <title>87.7 Bailrigg FM</title>
    <banner href="http://radio.lancs.ac.uk/2001/images/WebBanner.gif">
      <moreinfo href = "http://www.bailriggfm.co.uk" target="Frame"/>
      <abstract>The Best New Music :: www.bailriggfm.co.uk</abstract>
    </banner>
    <ref href = "mms://148.88.153.3/BFMLiveStream"/>
  </entry>
</asx>

```

Figure 4: Contents of Stream.asx file

A more advanced extension to the stream may be to embed MPEG-7 (more formally known as *Multimedia Content Description Interface*) video, as the broadcast sector is an identified target domain for this very new standard [3]. This would allow some kind of ‘interactive studio’ to be developed as part of the stream, so that the user could for example click on a CD player graphic to view details of the current track playing; or click on the webcam for the live^{iv} webcam feed. Furthermore, web users could instantly send requests to the DJs by integrating the system mentioned in section 2.2.

It is felt that delivering high-quality radio over the internet is imperative in the ‘future-proofing’ of Bailrigg FM’s image both on and off line, especially with the inevitable increase in number of internet-ready systems, such as PDAs and so called third-generation mobile phones.

3 PROPOSED PROJECT

Three separate solutions to the three problems detailed above are discussed in this section.

3.1 Database Solution

This solution essentially has two individually solved parts, namely the integration of Track Logging with the Record Library and the introduction of an automated playlist and chart, itself to be included within the new web-based tracklogging system.

Solving the former will not just allow more accurate listings of tracks that have been played to be collated, but also enable a warning system to be deployed, whereby a DJ receives a notification if (s)he plays a track that was played in the previous hour.

3.1.1 Integration of Track Logging and Record Library systems

The way in which DJs use the tracklogging system will have to be changed in order to account for the cross-referencing of the record library system.

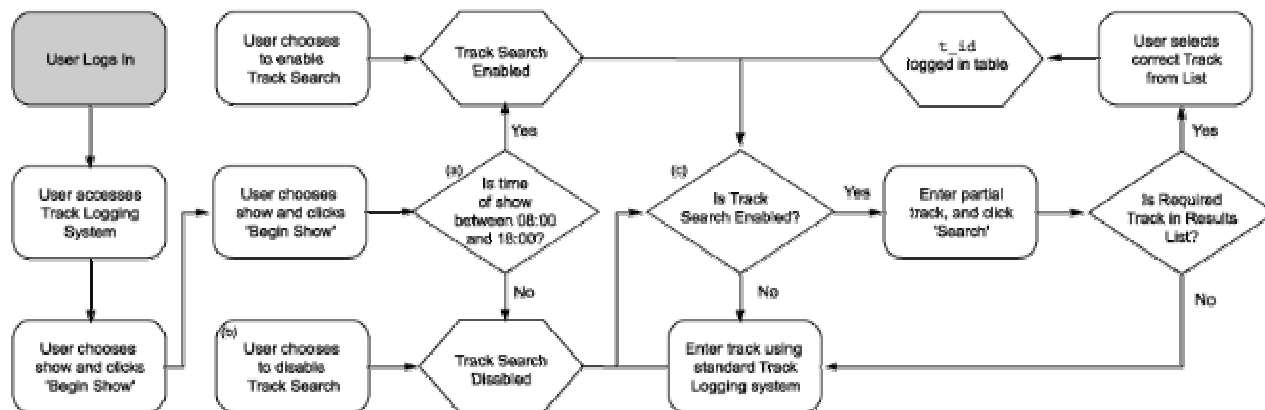


Figure 5: Stages through proposed track logging system

Figure 5 shows how a user navigates through the proposed system, starting at the login stage in the top-left. If the time of the show starting is between 8am and 6pm (i.e. non-specialist shows), the track search facility is automatically enabled (a), as it

^{iv} All streaming media currently provided by Bailrigg FM actually has a delay of up to thirty seconds

is assumed that most daytime DJs will be playing tracks present in the record library (such as playlist tracks). Evening DJs are less likely to warrant the use of the search facility, and so it is disabled for them (they can easily re-enable at the click of a button if they want, as the event marked (b) shows). Stage (c) will always be reached as long as there are still tracks to be entered (i.e. until End Show is selected).

Allowing the user to turn the search facility off may lead to none of the DJs actually using this system. It will somehow have to be enforced that DJs *do* use this system where possible so that charts and web content are dynamically updated with the highest accuracy.

3.1.2 Searching SQL

A complex query is required to efficiently search the tracks in the Record Library collection of tables and find the required item to update in the Track Logging table. PHP allows variables to be passed very easily into such queries, and for the attempt in Figure 6, `$search` is used to represent the data in the track search FORM on the Track Logging webpage.

```
SELECT t.t_name, a.a_name, c.c_name
FROM tracks AS t, artists AS a, collections AS c
INNER JOIN tracks_in_collection AS j ON c.c_id = j.c_id
INNER JOIN t ON t.t_id = j.t_id
INNER JOIN a ON t.t_artist = a.a_id
LIKE UPPER('%%$search%')
ORDER BY t.t_name
```

Figure 6: Possible query for the track search function

The `INNER JOINS` ensure that all relational data is properly returned when the query is executed, and thus can enable the list of matching tracks to show the track name, artist and collection (i.e. album or EP). The `LIKE UPPER` keywords ensure that the search is case insensitive, and the `%` signs mean that the search text can be anywhere within the existing table data. PHP also has functions to trim search criteria down, if for example, a search string of over 20 characters is entered.

3.1.3 Usability

As mentioned above, a pop-up window should display relevant results to the track search. The user should then click on the correct track, and a hyperlink should automatically reference back to the original track logging page, and update the record as necessary. A small amount of JavaScript may be required to allow for this functionality.

As the record is updated in the database, a check should be done to see whether duplicate `t_ids` have been entered within a specified time period (or more accurately, within the previous DJ's show). Another complex query would be required to solve this problem, and a way in which the user could be notified would have to be decided (choices of dialogue boxes, pop-up windows or bold on-screen text).

Other on-screen notifications could be to include the `a_desc` entity from the Artists table so that when a track is entered, the DJ can find out more about who (s)he is playing, thereby releasing useful information to use in a 'link' between tracks.

Each stage of the process illustrated in Figure 5 should have their own shortcuts so that 'advanced' users can efficiently navigate through the system.

3.1.4 Playlist Section

A new playlist table will need to be created in order to enable DJs to simply choose which playlist track they are playing from a list within the tracklogging system. This should involve the creating of a Playlist table within the Library database (see Table 4). The current (paper) playlist system also has a 'Release Date' column, but this should automatically be entered by querying the Collections table for the corresponding `c_release`.

<code>p_id</code>	<i>unique identifier</i>
<code>p_list</code>	<i>A, B or C list track</i>
<code>p_lib_ref</code>	<i>reference to record library entry</i>
<code>p_comments</code>	<i>comments relating to track</i>
<code>p_modified</code>	<i>date modified (to show whether this track is 'new' to the list)</i>

Table 4: Proposed Playlist table and descriptions

The two sides of this problem are the Head of Music (HoM) being able to update the playlist online; and the DJs being able to simply click on a playlist track to automatically enter it into the Track Logging table.

The former can be solved by creating a page that only the HoM can access, in which there are three sections: A, B and C. Each section will list the current playlist in the correct order (determined by artist name – see section 2.1.3), but will allow the user to remove tracks, add tracks, promote or demote tracks where appropriate. Tracks can only be added (as well as promoted or demoted) when there is sufficient room in that list (currently, each playlist has twenty tracks). As soon as the table is updated, the list will automatically re-populate, with correctly ordered tracks, and highlighted as new or climbers as appropriate.

When the HoM comes to add a new track, s(he) should use a similar system to the new Track Logging one mentioned above, whereby they enter a partial track name, and then choose the correct track from the list of results. For this to work correctly, all playlist tracks will have to be entered into the record library as soon as they arrive (currently, there is a short backlog of CDs being entered).

The latter will involve listing the playlist tracks within the track logging system, perhaps on the right-hand side, where there is currently 'Station Information'. This will be the most up-to-date copy of the A, B and C lists, which may be clicked upon by the on-air DJ as opposed to performing a track search.

If a track was played within the last hour, it should be coloured differently to the rest of the playlist so that the DJ is put off playing that track. It would be ideal to block the selecting of such a track altogether, but this would mean that should a DJ play a track before seeing whether it is available for selection, they may not be able to log it on the system at all.

3.2 Interaction Solution

It is proposed that a solution should be written to overcome the problems web users currently have in getting in contact with the radio station, as discussed in section 2.2.

3.2.1 Web-based

There are a number of different approaches to solving the track-request problem, excluding a simple FORM-to-e-mail. The most suitable solution is to use a simple database option, such as adding a Requests table to the Website database (see Table 5).

req_id	<i>unique identifier</i>
req_name	<i>name of web user</i>
req_college	<i>college of web user</i>
req_track	<i>track being requested</i>
req_ded	<i>dedication for track</i>
req_time	<i>date field for Now()</i>
req_done	<i>boolean value for completion</i>

Table 5: Proposed Requests table

A web-based form would be filled in by a web user, with their name, college and requested track. req_done would be set to false until cleared by the on-air DJ. Any waiting requests should appear on the track logging page, as long as req_done is set to false, and the request is less than 2 hours old^v. There seems very little point in allowing the web-user to choose the corresponding track from the record library, as with the DJ, but the system should be easily extendable to encompass this.

3.2.2 SMS-based

Further research will be required to see what the feasibility of integrating some kind of SMS gateway would be, including costs and hardware/software required.

3.3 Web Streaming Solution

There are a large number of benefits from extending the existing web stream, most of which involve it being more accessible to external bodies, such as other media groups, or specialist 'internet radio' websites. The following two solutions, namely extending the existing ASX-based system, and creating a new MPEG-7-based system would both cater for such accessibility issues.

3.3.1 Extending existing system

It is proposed that the current Windows Media™ stream should be extended to integrate relevant content that is currently available elsewhere on the current website. This includes currently playing information, current DJ and show information, as well as the new webcam.

These should be enabled by appropriately extending the ASX file that delivers the current stream, but will require a large amount of research into the capabilities of the ASX scripting language.

3.3.2 Deployment of new system

If it is at all feasible, it is also proposed that MPEG-7 video should be embedded within the stream, either running alongside the existing stream, or as a separate option for the latest versions of Windows Media™ Player only. Due to the fact that MPEG-7 is a very recent standard, there is very little documentation and support, and so this proposal may be infeasible. Research into the viability of embedding such a new technology will have to be carried out.

Among the advantages of this system would be the possibility for media to be easily searched via special online media-searches. Should the various interviews that Bailrigg FM have done over the years be made 'stream-able', a web user could search for a group, and end up at the central BFM stream, allowing them to listen to the interview and find out more about the group. The 'currently playing' section would also allow the stream to be listed on specialist websites, where a web user can see what is being played on any internet radio station at any one time.

^v determined by the req_time entity

4 PROGRAMME OF WORK

The timescale for the completion of this piece of work is 20 weeks, running from Week One of Michaelmas Term 2002 to the end of Lent Term 2003. The following programme of work is based upon these 20 weeks. Figure 7 shows these weeks' delegation as tasks (4.1 to 4.5) over time (1-20).

4.1 Phase One – Research

An amount of research will need to be carried out at the start of the project. This will take place across the first three weeks.

4.1.1 End-User Analysis

The first phase of this project is to officially analyse results obtained from the questionnaires targeted at Bailrigg FM members and BFM Online users, concerning the status of the current website. It will also involve interviewing potential end-users of the system, such as the Head of Music and Record Librarian, to find out their level of competence in using a new system, and to also settle any questions they may have.

4.1.2 MPEG-7 Research

The feasibility of using an MPEG-7 based solution should also be researched during this time (and possibly even during the summer holiday period). The software tools and relevant platforms should be analysed to find out if this part of the solution is viable.

4.1.3 Learning ASX

The XML-based ASX language needs to be fully understood before a new web-streaming solution is devised. The MSDN provides a large amount of information on this, and other Windows Media™-based solutions, and so this should be analysed appropriately.

4.1.4 SMS Feasibility

The possibility of incorporating an SMS gateway of some kind would also need to be investigated during this period.

4.2 Phase Two – Designing Solution

The whole system will be designed and developed across these ten weeks. Time has been split up to allow each section an appropriate length of time to complete, without the need to deal with more than one section at a time.

4.2.1 Database solution

The generation of new tables, queries, dialogues and PHP scripts will need to be undertaken in developing the database solution. This should take no longer than three weeks.

4.2.2 Interaction solution

This solution will also involve the creation of tables, queries and PHP scripts. If possible, it will also involve the integration of an SMS-based solution, to allow listeners to send 'text-requests' directly to the DJ's track logging screen. This should take no longer than three weeks.

4.2.3 Web-streaming solution

The creation of a brand new web stream (through MPEG-7 technology), or the extension to the existing solution (through ASX scripting) should take place during this time. The four weeks allocated may not be required should the former be inappropriate for inclusion in the project. It should be known long before 'Week One' whether this will be the case or not, and so time can be reallocated elsewhere (4.2.1 and/or 4.2.2) if this part of the solution is impractical.

4.3 Phase Three – Testing (Standalone) Solution

This very important stage will cover all aspects of system testing, and will take place over two weeks.

4.3.1 Unit Testing

Each of the separate parts of the project should be fully tested, and the testing documented. Any failures in the system should be fixed, before allowing the end user to test the system.

4.3.2 Usability Testing

A sample of end-users should be asked to use the working systems before they are integrated into the rest of the website. Feedback from these users will be used to refine the system before it is fully integrated in Phase Four.

4.4 Phase Four – Integration

This stage should not take more than one week to implement. All that may be required are a few hyperlinks being added to various existing web pages, and various end-users being fully documented on how the systems work. Parts of the documentation can be written during the development progress, as also shown in Figure 7.

4.5 Phase Five – Conclusions

During this period, a further questionnaire should be written in order to find out how happy end-users with the final solution. The final report should also be completed during this time, although this *can* be started during the integration phase, as shown in the diagram below. This phase will take place over four weeks, and as the title suggests, concludes the solution.

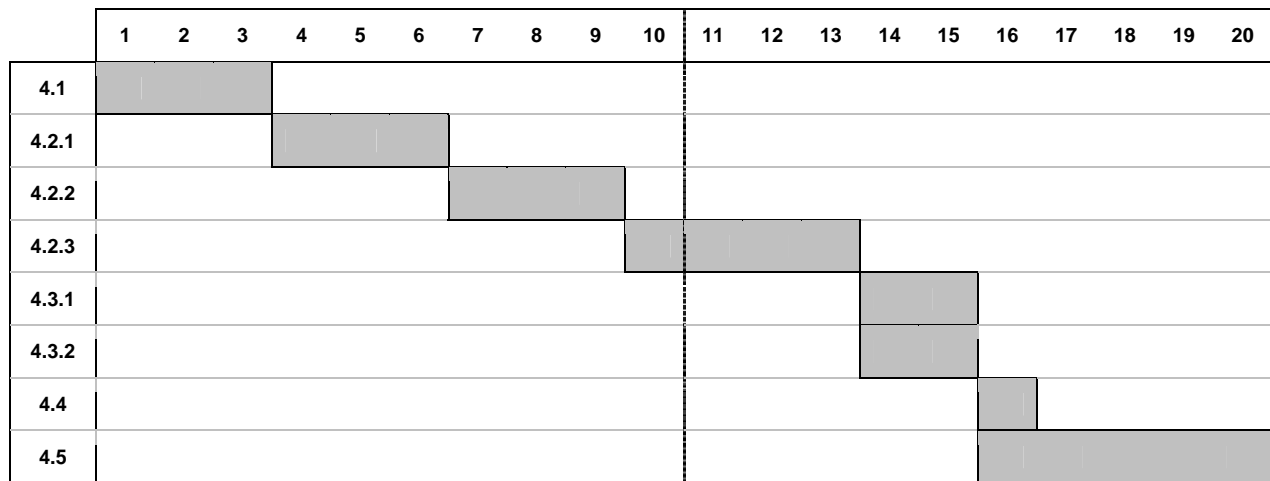


Figure 7: Time chart for project completion

5 RESOURCES REQUIRED

Most of the resources required for this project are already catered for at the Bailrigg FM office and in the Computing Department.

Software required to develop MPEG-7 and SMS solutions, and to a lesser extend ASX is not currently provided (however, ASX is an XML-based language, of which relevant tools such as Microsoft Visual Studio.NET, could be utilised). Due to the nature of the MPEG-7 solution, relevant development (and even deployment) tools may not even exist as yet.

6 REFERENCES

-
- 1 Courtesy of Adam Bardsley, Bailrigg FM
 - 2 Courtesy of Adam Bardsley and Peter Price, Bailrigg FM
<http://www.bailriggfm.co.uk>
 - 3 International Organisation for Standardisation – Coding of Moving Pictures and Audio, March 2001
<http://ipsi.fhg.de/delite/Projects/MPEG7/Documents/W3934.htm> & [.../W4035.htm](http://ipsi.fhg.de/delite/Projects/MPEG7/Documents/W4035.htm)