RTAG01: Technical Specification

v0.6.1 (2009-06-15)

All contributions and comments
should be sent to feedback@radiodns.org

This document is still a work in progress and is regularly being updated. Please ensure you have the latest revision available from http://radiodns.org/

Table of Contents

Introduction

Radio devices span a wide range of price and functionality. It is unrealistic to expect the same level of functionality across all types of device. This document defines a very simple application for capturing listener interest in what they are hearing on the radio.

The simplicity is intended to allow for the greatest possible scope of implementation across all devices. It is recognised that some simple devices may do no more than simply implement a single “tag” button, whereas other more capable devices may use the “tag” as the starting point for more complex interaction.

The proposed method also supports radios which do not have a persistent connection to the Internet, thus allowing an implementation on devices such as portable audio players.

Concept

The concept allows the listener to signal interest in the content of an audio service. The process creates a "**tag**", a collection of meta-data that uniquely references a point in time to a specific audio service on a specific audio device. The generation of the tag would be from user interaction with a device through a simple interface at, or close to, the moment of interest. This process is referred to in this document as "**tagging**".

A successful transaction would involve the following stages:

1. **Service Selection**

   The user selects a service on the device and begins listening.

2. **Tag Generation**

   The user interacts with the device to express an interest in a current audio event on the current service. The device generates a tag with relevant parameters required to provide contextual information on the event at a later time.

3. **Service Lookup**

   The process described in the previous RadioDNS document takes place and provides a location for which to deliver tags to for the service provider associated with the current service. The SRV record for this application will be `_radiotag.`

4. **Tag Delivery**

   The tag data compiled at stage 2 is delivered to the location obtained in stage 3. If the device does not have an outgoing transmission path at the stage, the data is stored on the device for later transmission, either directly or via a proxy service.

Tags relevant to each service are returned only to the respective originating service provider. Where appropriate, the device must use RadioDNS to locate the domain for the desired service(s). The service provider must provide a SRV record on their domain for the protocol `_radiotag` which provides the URI to which tags may be delivered. The absence of a `_radiotag` SRV record indicates a service providers inability to receive tag submissions.

Devices must make available a tagging option until it is explicitly determined that tag submissions cannot be received by the current service's provider. This allows the ability for a roaming user, disconnected from the public IP network, to discover and tag a new service before the device can definitively establish if tagging is supported.

Tags can be returned to the originating service provider immediately, or sent periodically. Tags can be returned individually or in batches.

Tagging can be implemented on any audio capable device that has periodic or permanent access, either directly or via a proxy service, to services on public networks utilising TCP/IP.

One possible future use of this service is that the service provider will collate the tags from each unique audio device, and offer a process where a user can register his/her ownership of one or more devices to view the collated tags, or authorise access by third party applications to the tags that they have created. The service provider may supplement the tag information with contextual information relevant to that point in the original audio stream. This proposed usage is expected to be proprietary to each service provider and is not defined within this document.

Implementation

This document specifies the implementation of tagging for the following audio delivery protocols:

- VHF/FM
- DAB Digital Radio
- DRM (Digital Radio Mondiale)
- HD Radio (iBiquity Digital Corporation's HD Radio)
- IP-delivered Audio

Other bearers may be supported, and will require subsequent definition.

This document specifies the implementation of tagging for the following listening environs:

- **Absolute audio**
  Tags recorded during live linear output from a service, or a time-shifted version of such, are referred to as Static Tags. This is where the data recorded to identify the tagged event is a static concrete time such as 13:15pm on Tuesday, 1st January, 2008.
- **Relative audio**
  When a tag is recorded during a pre-recorded item, such as a podcast or piece of on-demand content, there is no concept of concrete timing. This is referred to as a Relative Tag because the timing information recorded is an offset in seconds relative to the beginning of the file.

The protocol for the submission of tags to a service provider is a REST-style HTTP interface, using the JSON data interchange format (*RFC 4627, "The application/json Media Type for JavaScript Object Notation"*) optimised for efficient transactions and simple device implementations.

One time authentication process

The first time a device connects to a service provider it is required to authenticate. This is to help prevent the "spamming" of tagged content. It is designed to be a compromise between making the process simple for users and making service attacks difficult.

The proposed process is:

1. **User registration with service provider**
   User goes to the tagging section of a service providers website and signs up for an account. The sign up process includes prompts to associate their first device to their account. It may also offer the ability to associate an additional device with their account, as each device will follow this process the first time it wishes to be used with a service provider.

2. **Device registration with service provider**
   The registration process will ask for the Unique Device Identification (UDI) which will be surfaced on the device via a menu or be clearly visible on the casing. This number is supplied to the service provider.

3. **Registration process**
   The website will then promote the user to tune to any of that providers services and select the register option on the tagging menu of the device. This sends a particular register request to the service provider.

4. **Device responds with authentication token**
   The device then sends an authentication token response back to the device. This must be stored on the device against the domain being used for the service provider and included in all future tag submissions to that domain, else they will be rejected in an attempt to avoid spam.

Note: Due to the nature of many existing radio devices, the necessary steps involved with distributed authentication protocols such as OAuth are not possible.
Please see "Multiple User Devices" in the "Considerations" section for additional thoughts on devices with multiple users, such as kitchen radios.

Registration request

Once a service lookup has been performed using RadioDNS the device can submit a registration request to the acquired location for the service provider. This is performed via a HTTP POST request in JSON format to the URL **`http://<fqdn>/register`** resource. The JSON content body is contained with the **`data`** POST variable.

The following parameters are mandatory:

Parameters	Description	Value	Status
`v`	**Protocol Version** Used to ensure the server can support the version of the protocol the client is using	*float*	*mandatory*
`d`	**Unique Device Identification (UDI)** The device should supply a uniquely identifiable string, such as a serial number (please see the "Considerations" section for further information)	*string (max. length 128 char)*	*mandatory*

An example registration request:

```
{
      "v": 1.0,
      "d": "ABCDEFGH0123456789"
}
```

Registration response

The HTTP status code of the response indicates the success or failure of the registration attempt. For example a 404 determines that the registration service is not located where the device expected, and 200 determines the request was successful. 403 determines that a reason occurred which prevents the device from registering.

If successful, a token will be returned which must be included in all future transactions with this service provider, therefore it should be stored on the device against the domain. The token value is an MD5 hash, a 32-character hexadecimal number.

A typical successful registration response:

```
{
     "token": "535b966a0db751a78638828cd4e7577av"
}
```

If an error occurred, an optional text message may be returned from the service provider to define the reason behind the unsuccessful registration attempt.

A typical unsuccessful registration response:

```
{
     "rsp": "Device not recognised. Please first register your
     device at provider.net/tags"
}
```

Tag submission

The tag submission is performed in the same way as the registration process, by sending a JSON formatted request in the data POST parameter to the **`/tag`** resource on the domain obtained for the tagged service.

The tag is composed of a set of common paramters, followed by an array of one or more tags. This allows devices to store many tagged events for a particular service and submit them in one single submission.

Common parameters

For any tag, the following parameters are mandatory:

Parameters	**Description**	**Value**	**Status**
`v`	**Protocol Version** Used to ensure the server can support the version of the protocol the client is using	*float*	*mandatory*
`d`	**Unique Device Identification (UDI)** The device should supply a uniquely identifiable string, such as a serial number (please see the Considerations section for further information)	*string (max. length 128 char)*	*mandatory*
`a`	**Authentication Token** Token associated to the domain resolved for this service, which the tag is being submitted to acquired during registration. If no value held, provide an empty string to force failure	*string (32 char)*	*mandatory*
`u`	**System Timestamp** The current Unix epoch on the device. This is used to calculate drift and correct the timing of subsequent tagged events	*integer*	*mandatory*
`b`	**Bearer** Details which bearer the service is broadcast upon (either **`vhf-fm`**, **`dab`**, **`drm-amss`**, **`hd`** or **`ip`**)	*enum* **`vhf-fm`**, **`dab`**, **`drm-amss`**, **`hd`**, **`ip`**	*mandatory*

Bearer-specific parameters

The common parameters are then followed by the bearer specific parameters, dependent on the value of **`b`** (Bearer):

VHF/FM

Parameters	Description	Value	Status
`ecc`	**Extended Country Code (ECC) and country code** The broadcast RDS ECC code concatenated with the first character of the broadcast RDS PI code (country code) must be supplied if available	*3-char hexadecimal*	*mutually exclusive*
`country`	**ISO 3166 two-letter country code** In the event that a broadcast ECC is unavailable, an ISO 2-letter country code must be provided	*2-char string*	
`pi`	**Programme Identification (PI)** The broadcast RDS PI code	*4-char hexadecimal*	*mandatory*
`freq`	**Frequency** The frequency on which the service is received in units of 10KHz	*float (5, 3)*	*mandatory*

For a detailed explanation on these parameters and their values, please refer to the VHF/FM section of the RadioDNS Specification (*"FQDN construction for Broadcast Protocols", "RDNS01: Technical Specification" RadioDNS.org*).

DAB Digital Radio

Parameters	Description	Value	Status
`ecc`	**Extended Country Code (ECC)** The service's broadcast multiplex ECC code	*3-char hexadecimal*	*mandatory*
`eid`	**Ensemble Identifier (EId)** The service's broadcast multiplex ensemble ID	*4-char hexadecimal*	*mandatory*
`sid`	**Service Identifier (SId)** The service's broadcast identifier	*4 or 8-char hexadecimal*	*mandatory*
`scids`	**Service Component Identifier within the Service (SCIdS)** The service's broadcast component identifier	*1 or 3-char hexadecimal*	*mandatory*

If the audio service is delivered as data via X-PAD, the following parameter is also mandatory:

`appty-uatype`	**X-PAD Application Type (AppTy) and User Application type (UAtype)** The X-PAD Application Type number and User Application Type, concatenated with a hyphen (only for applications broadcast in X-PAD). Where Application Types are allocated in pairs, the lower value (indicating the start of the application data group) must be used.	*2-char hexadecimal, hyphen, 3-char hexadecimal*	*mandatory when referring to an X-PAD component, omitted otherwise*

If the audio service is delivered as data in an independent Service Component, the following parameter is mandatory:

`pa`	**Packet Address** The packet address of the data service delivering the audio service	*integer between 0 and 1023*	*mandatory when referring to a data service component, omitted otherwise*

For a detailed explanation on these parameters and their values, please refer to the DAB Digital Radio section of the RadioDNS Specification (*"FQDN construction for Broadcast Protocols", "RDNS01: Technical Specification" RadioDNS.org*).

DRM (Digital Radio Mondiale) / AMSS (AM Signalling System)

Parameters	Description	Value	Status
`sid`	**Service Identifier (SId)** The service's broadcast identifier	*6-char hexadecimal*	*mandatory*

For a detailed explanation on these parameters and their values, please refer to the DRM / AMSS section of the RadioDNS Specification (*"FQDN construction for Broadcast Protocols", "RDNS01: Technical Specification" RadioDNS.org*).

HD Radio (iBiquity Digital Corporation's HD Radio)

Parameters	Description	Value	Status
`tx`	**Transmitter Identifier** The service's broadcast transmitter identifier	*5-char hexadecimal*	*mandatory*
`cc`	**Country Code** The service's broadcast country code	*3-char hexadecimal*	*mandatory*

For a detailed explanation on these parameters and their values, please refer to the HD Radio section of the RadioDNS Specification (*"FQDN construction for Broadcast Protocols", "RDNS01: Technical Specification" RadioDNS.org*).

IP-delivered Audio

Parameters	Description	Value	Status
`url`	**Uniform Resource Locator (URL)** The URL from which the audio is being served, including the path including additional parameters that must uniquely identify the audio	*string*	*mandatory*

Tagged events

The tagged events themselves are then returned in an array variable:

Parameters	Description	Value	Status
`tags`	**Tags** An array of tagged events (specified below)	*array*	*mandatory*

Within the tags array the following parameters are specified:

Parameters	Description	Value	Status
`u`	**Time** Unix epoch at the time in which the user interacted with the device, must be supplied for absolute audio type services	*integer*	*mutually exclusive*
`p`	**Period** Offset time in seconds from the start of the audio, must be supplied for relative audio type services	*integer*	
`f`	**Filename** If the audio being consumed is not that of a live or time-shifted linear audio service (for example a podcast, listen-again content) the filename of the file being consumed must be supplied	*string (max. 255 char)*	*mandatory for non-linear services, omitted otherwise*
`r`	**Rating** If the interaction type has the ability to provide feedback on the intensity of interest in the event, this can be reported via this parameter	*float(2,1), max value 1.0*	*optional*
`n`	**Note** An optional string, defined by the user, can be associated with a tag to aide recollection	*string (max. 255 char)*	*optional*

It is required to provide the time at which the event of interest occurred. This is done in one of two ways, dependent on the type of audio the event occurred within.

When a static audio service is consumed, the specific point in time must be provided via the `u` (time) parameter. When consuming time-shifted content the device must calculate the original broadcast time by taking the current time and subtracting the known buffer length.

When a relative audio service is consumed with no concrete timing the value of `p` (period) must be the offset in seconds relative to the start of the content.

A device may feature the ability to express a rating for a tagged event. For example, a star system where a rating from one to five stars could be attached to the tag, the lowest value (one star) would be represented in the `r` variable as `0.2` and the highest value (five stars) would be represented as `1.0`. In the absence of this optional parameter, the service provider may apply their own interpretation of the rating value.

The `n` (Note) variable is optional and should only be populated by the user manually entering a note on the device related to the tagged event.

Request

An example content body for a tag request:

```
{
      "v": 1.0,
      "d": "ABCDEFGH0123456789",
      "a": "535b966a0db751a78638828cd4e7577a",
      "b": "vhf-fm",
      "ecc": "e1c",
      "pi": "c123",
      "freq": 105.4,
      "tags": [
            {
                  "u": 1207942205,
                  "r": 0.8,
                  "n": "What is this song?"
            }
      ]
}
```

The above request depicts a tag request for a live VHF/FM service broadcasting with an ECC and country code value of `E1C`, a PI of `C123` on a frequency of 105.4 MHz. The tagged event happened at 20:30:05 UTC on Friday, April 11th 2008 and was associated with a rating of four out of five stars (`0.8`).

A request can contain multiple tag events. As tags are to only be delivered to the service provider of the service being tagged, a device must only send multiple tags for the same original service in any one submission.

An example content body of a multiple tag submission:

```
{
    "v": 1.0,
    "d": "ABCDEFGH0123456789",
    "a": "535b966a0db751a78638828cd4e7577a",
    "b": "dab",
    "ecc": "E1",
    "eid": "C4F2",
    "sid": "5C4D",
    "scids": 0,
    "tags": [
        {
            "u": 1207942205,
            "r": 0.8
        },
        {
            "p": 128,
            "r": 1.0,
            "n": "Funny joke"
        },
        {
            "u": 12023435554
        }
}
```

Response

When sending a tag submission request, the HTTP status code of the returned response indicates the overall success or failure of the entire request.

A status code 200 implies that one or more of the tags submitted succeeded. Only if all tag submission requests failed would a HTTP status code 400 indicating total failure of the request be returned.

Status for the individual tags will be indicated in the response content body via an array (**`rsp`**). The order of responses in the returned array is the same as that of the original tag submission requests, in the following format:

```
{
      "rsp": [
            {
                  "stat": 200,
                  "url": "http://www.hotfm100.com/mytags"
            }
      ]
}
```

The status codes are as follows:

Status Code	**Description**
`200`	The request was successful
`400`	There was an error in your request, either the JSON was malformed, a required variable was missing or the value was invalid
`403`	Device is not registered with the service provider
`410`	Service not recognised
`500`	A server-side error occurred which means the tag was not received correctly
`501`	Protocol version is not supported by the receiver

Please see the official HTTP status code definitions (*Section 10, "RFC2616: Hypertext Transfer Protocol – HTTP/1.1", Fielding et al.*) for detailed information.

The URL indicates the location a user should go to view the rich meta-data associated with the submitted tag. If a status code of 403 is returned, the URL indicates a location the user should go to to authorise the device.

Unauthorised tag submissions should be held on the device for a fixed amount of time so that further submission attempts can be made following successful device authorisation.

Please refer to the “Recommended device behaviour” section for suggestions on the way devices respond to user interaction.

Manufacturer portals

There may be a situation where a device manufacturer would prefer to place their own portal or proxy between the device and the service provider. This scenario is possible using the previously defined tagging specification, with the expected process as follows:

1. **User registration**
   A user registers their device with the manufacturers portal, as guided by documentation enclosed with the device

2. **Tagged events are sent from the device to the device manufacturers portal**
   The method for sending these tags is at the device manufacturers discretion and falls outside of the scope of this document.

3. **Portal server utilises the centralised lookup for radio services**
   As previously described in both this document and the centralised lookup documentation, a service is resolved by the portal server in the same way as a directly connected device. This establishes where the portal is to send the tags to.

4. **Portal server authenticates with service provider**
   Where the device authentication token is usually included in the tag, the service provider will include their own unique authentication token agreed with the service provider (see below.)

5. **Portal server delivers the tags to the service provider**
   The portal server can then complete a standard tag delivery as described previously in this document.

Beyond this specification, manufacturers would then be required to enter into bilateral agreements with service providers to be able to access the rich meta-data generated by users' tags. This falls outside the scope of this document and is not covered further.

Recommended device behaviour

This is a selection of recommendation to manufacturers on how their devices can improve the usability experience of tagging.

Human readable tag data

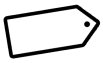
It is recommended that device manufacturers store the RDS, DAB or IP meta-data service identifying name in the cached tag data (where data is stored on the device prior to submission.) This allows both the device itself and proxy software that sits between a device and tag submission to browse the cached tag data and display the stored information more accessibly to the user.

For example, the user could be presented with a question such as “Add, Remove or Delete tag on Sunday, 1st January at 6pm on HOTFM99”. This allows the user to make decisions on the stored tags prior to the rich meta-data being acquired via the service provider.

Providing feedback to the user

It is important for devices to clearly display the status of a tag interaction to the end user. Example cases are included below, with a suggestion on how the device may provide feedback to the user.

- **Tagging support is unknown or explicitly defined as available**
  Tag icon displayed / tag button displayed



- **Tagging support is explicitly defined as unavailable**
  Tag icon removed from display / tag button removed from display
- **Tag button is pressed**
  Tag icon flashes to acknowledge the button push
- **During submission of tags to service provider**
  Tag icon changes and animates to show data transfer

- **Submission succeeded**
  Tag icon with a tick struck through displayed

- **Submission failed**

  Tag icon with a cross struck through displayed

- **Submission unauthorised**

  Tag icon with a padlock displayed

As a device may be submitting multiple tags, for example if it has previously been disconnected and recently been reconnected to a public IP network, a simple visual indication may be difficult to tie up to a particular tagged event.

It is suggested devices also provide a status menu which can show multiple tag events and their relevant status, for example:

Recent Tags	
10/05/2008 10:33:21 ***HotFM (100.0, FM)*** LOVE THIS SONG! *Queued awaiting submission. Please reconnect your device to submit.*	📶
10/05/2008 09:12:18 ***FAB Digital (Central Multiplex, DAB)*** *An error occurred submitting this tag. Tag submission will retry each hour for next 7 days.*	✗
08/05/2008 08:16:33 ***Cool Hits Online (coolhitsonline.com, IP)*** *Submission successful. Visit http://www.hotfm100.com/tags to see your tags.*	✓
07/05/2008 16:33:43 ***BigEasy (Northern Multiplex, DAB)*** Funny interview, must listen again *Device not registered. Visit http://www.bigeasy.co.uk/ to register your device.*	🔒

Audio storage

If the device features functionality to record radio output, when a tag is generated a short 10-20” clip could stored encapsulating the tagged event. This would be particularly useful on disconnected devices, allowing the user to browse and listen to the tagged event before being in a situation to lookup the rich meta-data associated with the tagged event.

User notes

On devices that provide a usable text entry mechanism, such as devices with large touch screen keyboards, it is possible to send a short form text field with the tag submission. This would allow the user to put a memorable note alongside the tagged event, for example "Must buy this song".

When the tag is later reviewed on the service providers website after submission the note would be displayed alongside the tag allowing them to recall why they tagged the particular event.

When to perform application lookups

It is important to consider the differing connectivity states for devices. Where a continuous connection is available, such as a broadband Wi-Fi connection, it is preferred for a device to look ahead before any user interaction occurs to establish which services should remain advertised to the user. However on devices with limited or expensive connectivity, such as EDGE or GPRS, it is preferred to only perform service availability checks when interaction for that service occurs, reducing network activity.

Considerations

Unique device identification

A device is required to uniquely identify itself when submitting a tag to a service provider. This must be a totally unique identifier such as a serial number or a device's MAC address, not device manufacturer and model. It is also important that this number be easily accessible within the device's user interface as the broadcasters website will require the end user to input this number when signing up for the service.

It is suggested this be surfaced in the interface menus alongside any settings for tagging, under an option such as "Tag Device ID".

Date/time accuracy

The accuracy of the time associated with a tag is crucial for the broadcaster to correctly match it to the on-air event. It is required that the devices clock be maintained to an accuracy of one second. The device should use one of the following sources to obtain an accurate time (ordered by preference):

1. NTP (Network Time Protocol)
2. DAB FIG 0/10 ('Long Form')
3. DAB FIG 0/10 ('Short Form')
4. RDS CT (Clock Time)

It is expected that devices which require a proxy service for public network access may perform a clock synchronisation on connection to the relevant proxy service. The device must take in to account the effect this has on previously tagged times. The device should calculate the offset in time between that previously held on the device and that which has been synchronised, altering the tagged data accordingly to avoid incorrect submissions.

In the cases of DAB FIG 0/10 ('Short Form') and RDS CT, the device must pay attention to the special handling required to ensure accuracy to the nearest second in the absence of this information being explicitly transmitted.

Tag submission throttling

It may be appropriate to specify a maximum rate at which tags can be submitted to the service provider, to prevent abuse of the system through uncontrolled submission of tags. This “tag spamming” could disproportionately consume resource at the service provider, and allow for undesirable use of the system.

It is suggested that consideration is given to:

- Restricting the number of tags that may be submitted in any one POST to a service provider
- Restricting the number of POST events that a device can make to a service provider within a specified time window

Dynamic availability of tag submission

It has been requested to provide ability to enable/disable tag functions more dynamically, for example on a show by show basis. This would be possible on a fully connected device by use of the TTL parameter of the `_radiotag` SRV record to force the device to query the availability of the SRV record more frequently and thus update the availability more frequently.

However, this would not work as desired on intermittently connected devices. It is also questionable how this would impact the end user experience, where tagging would be available at certain times on a service and not at others.