Eastlan Playlist Compilation

Objective: A method of collecting continuous airplay data (Title, Artist, Date/Time) from 100 or so "Pop" radio stations, generally in smaller markets, all being served by Eastlan's ratings product.

Background: In a conversation with Howard Rosen & Dave Sholin, we spit-balled a couple different ways of doing this. The first thought was to have each radio station of interest export a music log to a common destination (Dropbox or FTP folder), where we could then import the information and collate it with other contributing stations. A major downside to this approach is the lack of humans at most radio stations, and convincing someone at the station to actually do this each day. Secondary concerns are that some stations buffer each hour with "fill songs", which may not get played on-air, as well as the fact that some stations might be hesitant about sharing their music log data with a third-party, especially when logs are usually created a day or so before they air.

A better approach: Most pop stations stream their music, and many even post "Recently Played" titles on their website, or in their streaming music player. There are two ways to snag the song metadata; the first being easier than the second:

1 - If the station posts song Title/Artist info on their website, it's a fairly simple process to have an automated app go out every 3-minutes and retrieve that data from the .html page and add it to a database. Some stations post the last 10-15 songs played. In that case, the app could go out every 20-minutes or so, and update all the most-recent songs it hadn't yet collected. In either case, you could probably continuously "monitor" 100 or more radio stations, using just one program on one computer with a solid Internet connection.

2 - In the case that the previous scenario is not feasible, another method would be to intercept the station's IceCast or ShoutCast or Wide Orbit, etc. stream, and pull the metadata for each song from the stream. This can be done manually today using something like WinAmp. I'd imagine a developer could automate the process or build something custom to achieve this. For example, VLC is an open-source program with many add-ons & extensions. More on VLC, below.

As a proof of concept exercise, I selected 5 random Eastlan markets (generally the larger of their markets), and chose a Top 40 station from each. Then I went down the rabbit hole to see how this might be achieved in each case, or if it was even feasible.

Johnson City / Bristol, TN - WAEZ - Electric 94.9 - https://www.electric949.com
Wide Orbit Automation.
Last 10 songs played info is here:
https://www.electric949.com/station/iframe_last10played.shtml
It's .html text; easily parsed out, could be pinged once every 20-minutes.

Utica, NY - WSKS/WSKU - KISS-FM - https://cnykiss.com/
Envisionwise / LinkedUpRadio
TuneGenie media player lists most-recently played on home page, but the last two hours of songs are here, and easily parsed from this .html page: http://wsks.tunegenie.com/#listenlive

South Bend, IN - WNDV - U93 - https://u93.com/
Envisionwise / LinkedUpRadio
TuneGenie link:
http://wndv.tunegenie.com/#listenlive
The last 2-hours (~24 songs) are listed here; even with the times they aired! It's in .html, easily parsed out, and could be pinged every hour or more.

San Luis Obispo - KWWV - Wild 106.1 - https://wild1061.com/
Most recent song and the time it played is here: https://socast-public.s3.amazonaws.com/player/np_2749_2436.js
This is .html and could be parsed out, but only one song is listed, so it should be pinged every 3-minutes.

Springfield, IL WDBR - https://wdbr.com/
Wide Orbit Automation.
Stream is here: https://v7player.wostreaming.net/7920
Last 40 songs are here: https://wdbr.com/recently-played/ (*note*: while the Title / Artist and even the time played are displayed, it's within a JavaScript app and not in the .html of the page. Some method to scrape the metadata from the JS app would need to be implemented, otherwise it would have to be pulled from the stream).

In conclusion: In the case of all 5 of these sample stations, it would seem possible to collect the desired Title, Artist and Date/Time information on a continuous basis. I think there's a way to make this happen if you could find the right person to cobble together the elements you would need. Someone who could dig into the JavaScript and TuneGenie apps many of these stations are using could likely provide a streamlined solution. My programming days (computer *and* radio) are well behind me, but I'm glad to provide some rough pseudo-code below to give you an idea of how a software program like this might function.

In the case of stations not offering a "Recently Played" page, I would ask their web developer to create a "hidden" page on their website, similar to those above, that would have at least the last 10 songs played. This page would not be listed on their website for the public, but could be accessible to us to parse the Title/Artist information.

Earlier I mentioned some sort of plug-in or media player component that can load a station's stream, read the Title & Artist of the song playing, and store that information in a database. There are many options; for example, VLC is an open source app: https://www.videolan.org/vlc/features.html
While it's often used for video files, it also handles all kinds of audio streams; MP3, AAC, WAV, etc. There are lots of add-ons and extensions available. I found this one in just a couple of minutes that reads Title & Artist info from a streamed URL and saves to a .CSV file:
https://addons.videolan.org/p/1154018

Here are other methods of doing the same thing:

To parse meta data from an MP3 stream (old PHP code from 2013):
https://gist.github.com/fracasula/5781710

Reading Meta Data from ShoutCast streams:
https://medium.com/appseed-io/display-song-title-and-cover-by-utilizing-shoutcast-s-meta-data-fb00011cb086

JS Music Metadata Parser for streams:
https://www.npmjs.com/package/music-metadata

OGG & MP3 streams:
https://bugzilla.mozilla.org/show_bug.cgi?id=908902

Sample software pseudo-code...

Database elements:

> **Station:** WAEZ Electric 94.9 (name of each station to monitor)
> **Method of collection:** 1 = parse from .hmtl 2 = pull from stream, etc.
> **Interval:** 3-minutes (how often to ping for new data)
> **URL:** http://waaa.com/last-songs (where to find data.)
> **Optional fields:** Market size, Format, City name, etc.

Data storage:

> **Station:** WAEZ
> **Title:** Song Title
> **Artist:** Artist
> **Date:** XX/XX/2022
> **Time:** 19:25:37

The Gist of the Program...

```
     If CurrentTime > LastTime + Interval ! It's time to get more data
          Get(Station,1)  ! WAEV
          Case of Method
          Of 1 ! Parse out .html
            Load(URL)  ! Go to Web page
            Search(#LastArtist)  ! Whatever search parameter to find data
            Search(#LastTitle)
            Close(URL)
          Of 2 ! Get metadata from stream.
            Load(URL) ! In Player widget
            NewTitle = Get(metadata Title)
            NewArtist = Get(metadata Artist)
            Close(URL) ! Close Player widget
          Of 3  ! any other methods derived to extract data
          ...
          End
          ! Add some logic to test for duplicate titles; if the current song
          ! playing was already added 3-minutes ago, don't add it again.
          Current Time = Time
          Add(Info to Database)

          ... Go to Next Station, etc.
    End
```

Collecting data in this manner would make it easy on the backend to sort/filter results based on market size, format, date range, number of plays, etc.