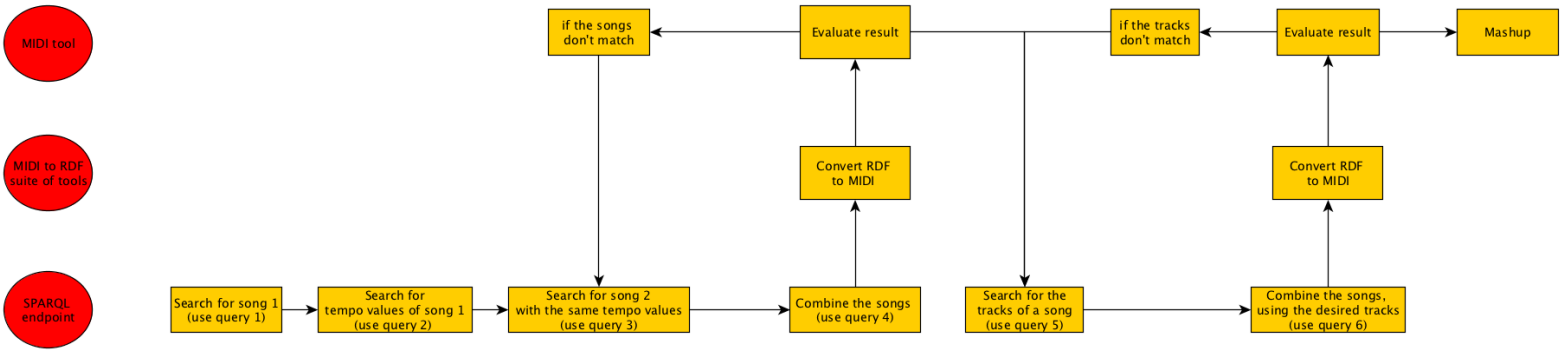

SPARQL endpoint: **http://virtuoso-midi.amp.ops.labs.vu.nl/sparql**
MIDI2RDF suite of tools: **https://github.com/midi-ld/midi2rdf/**

1. Search for song 1 (use query 1):

```
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX mid: <http://purl.org/midi-ld/midi#>

SELECT ?filename ?pattern

WHERE {
?pattern prov:wasDerivedFrom ?filename .
FILTER (regex(?filename, "war pigs", "i")) .
}
```

The result contains three songs, therefore I change the name to: “BLACK SABBATH.War pigs.mid”

2. Search for tempo values of song 1 (use query 2):

```
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX mid: <http://purl.org/midi-ld/midi#>

SELECT *

WHERE {
?pattern prov:wasDerivedFrom ?filename .
?pattern mid:hasTrack ?track .
?track mid:hasEvent ?timesignatureevent .
?timesignatureevent a mid:TimeSignatureEvent .
?timesignatureevent mid:denominator ?denominator .
?timesignatureevent mid:metronome ?metronome .
?timesignatureevent mid:numerator ?numerator .
?timesignatureevent mid:thirtyseconds ?thirtyseconds .
FILTER (regex(?filename, "BLACK SABBATH.War pigs.mid", "i")) .
}
```

This results in the following tempo values:
denominator: 4
metronome: 24
numerator: 4
thirtyseconds: 8

3. Search for song 2 with the same tempo values (use query 3):

```
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX mid: <http://purl.org/midi-ld/midi#>

SELECT DISTINCT ?pattern ?filename

WHERE {
?pattern prov:wasDerivedFrom ?filename .
?pattern mid:hasTrack ?track .
?track mid:hasEvent ?timesignatureevent .
?timesignatureevent a mid:TimeSignatureEvent .
?timesignatureevent mid:denominator 4 .
?timesignatureevent mid:metronome 24 .
?timesignatureevent mid:numerator 4 .
?timesignatureevent mid:thirtyseconds 8 .
FILTER (regex(?filename, "mozart", "i")) . #optional

}

LIMIT 1000
```

I thought it was interesting to combine War Pigs with something Classic, so I filtered on “Mozart”. This query took quite some time to load, but after it was loaded I found the song: "1mov "Eine Kleine Natchmusik".mid" which I thought was interesting. I will continue with this as second song.

4. Combine the songs (use query 4)

```
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX mid: <http://purl.org/midi-ld/midi#>

CONSTRUCT { <newsong> a mid:Pattern ;
mid:hasTrack ?track .
<newsong> mid:format ?format .
<newsong> mid:resolution ?resolution .
?track mid:hasEvent ?event .
?track a mid:Track .
?event a ?type .
?event ?property ?value .
}

WHERE {
{
?pattern prov:wasDerivedFrom ?filename .
?pattern mid:hasTrack ?track .
?pattern mid:format ?format .
?pattern mid:resolution ?resolution .
?track mid:hasEvent ?event .
?event a ?type .
?event ?property ?value .
FILTER (regex(?filename, "BLACK SABBATH.War pigs.mid", "i")) .
} UNION {
?pattern prov:wasDerivedFrom ?filename .
?pattern mid:hasTrack ?track .
?pattern mid:format ?format .
?pattern mid:resolution ?resolution .
?track mid:hasEvent ?event .
?event a ?type .
?event ?property ?value .
FILTER (regex(?filename, "1mov ''Eine Kleine Natchmusik''.mid", "i")) .
}
}
```

Note that the Results Format should be set to “Turtle”

5. Convert RDF to MIDI

1. The result from the preceding step needs to be copied to a .txt file
   - see example.txt
2. After this, change the .txt extension to .ttl
   - see example.ttl
3. Follow the steps on https://github.com/midi-ld/midi2rdf/ to convert the RDF file to a MIDI file

6. Evaluate result

1. Use a tool that can play MIDI to evaluate the result (for example Garageband for Macbook users or … for Windows users)
2. Listen to the result
3. Decide whether you think the songs can be a mashup
   - In theory they match because the songs have the same tempo, but the song still needs to sound appealing. This is just your own opinion, but since it is your own mashup you can decide what you want.
4. If you think the songs can't be a mashup, go back to step 3
5. Since I think the songs can be a mashup, I am going to query for the tracks of both songs

7. Search for the tracks of a song (use query 5)

I want from War Pigs only the drum track, this is the second track of War Pigs. This can be derived from both the mashup, or if this is too hard from the War Pigs midi itself.

```
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX mid: <http://purl.org/midi-ld/midi#>

SELECT ?track

WHERE {
?pattern prov:wasDerivedFrom ?filename .
?pattern mid:hasTrack ?track .
FILTER (regex(?filename, "BLACK SABBATH.War pigs.mid", "i")) .
}
```

After querying the tracks of War Pigs, I see that the second track is referred to using:
http://purl.org/midi-ld/pattern/6702325c4f08ece510e48093035e9fa6/track01

From Eine Kleine Nachtmusik I want all tracks, so I do not have to query for the tracks of this song.

8. Combine the songs, using the desired tracks (use query 6)

```
PREFIX prov: <http://www.w3.org/ns/prov#>
PREFIX mid: <http://purl.org/midi-ld/midi#>

CONSTRUCT { <newsong> a mid:Pattern ;
mid:hasTrack ?track .
<newsong> mid:format ?format .
?track mid:hasEvent ?event .
?track a mid:Track .
?event a ?type .
?event ?property ?value .
}

WHERE {
{
?pattern prov:wasDerivedFrom ?filename .
?pattern mid:hasTrack ?track .
?pattern mid:format ?format .
?pattern mid:resolution ?resolution .
?track mid:hasEvent ?event .
?event a ?type .
?event ?property ?value .
FILTER (regex(?filename, "BLACK SABBATH.War pigs.mid", "i")) .
FILTER (?track IN (<http://purl.org/midi-ld/pattern/6702325c4f08ece510e48093035e9fa6/track00>,
<http://purl.org/midi-ld/pattern/6702325c4f08ece510e48093035e9fa6/track01>))
} UNION {
?pattern prov:wasDerivedFrom ?filename .
?pattern mid:hasTrack ?track .
?pattern mid:format ?format .
?pattern mid:resolution ?resolution .
?track mid:hasEvent ?event .
?event a ?type .
?event ?property ?value .
FILTER (regex(?filename, "1mov ''Eine Kleine Natchmusik''.mid", "i")) .
}
}
```

Since I want all tracks from Eine Kleine Nachtmusik, I omitted the second track filter.

9. Convert RDF to MIDI
- see step 5

10. Evaluate result
The result sounds like a mashup. I am going to leave it like this.

11. Mashup
We have now constructed a MIDI Linked Data mashup, generated by the SPARQL query in step 8.