**Proposed Data Capture Architecture**

Author: Keith Martin, with input from Amit Kuthiala, Raj Garikapati, David Datta, Yong Gao, Trevor Lai, and Yishai Sered

DRAFT 2014-12-05

**Data transport**

The data transport between SCM and the Data Capture Service will be a POST URL sent over HTTPS, with the payload data encoded in JSON format (data fields specified below).

**Server side**

[DST (Amit & Raj) have indicated the desire to complete full functionality of this by the April software release.]

The Data Capture Service will be independent of Marge.

The Data Capture Service will run on a Tomcat Web Container, connected via AWS SQS (a managed Queue Service) to an Amazon S3 Data Repository. Data will be time-windowed (i.e., events older than, e.g., 10 days, will be archived to slower, cheaper storage, while active data will remain in the faster S3 repository).

The database layer itself has not been determined yet, but it will explicitly support a Spark layer (and any layers that can be built on Spark, e.g., Shark for SQL queries).

**Client side**

[HED (Yong, Trevor & Yishai) have indicated that full completion of this will require approximately three sprints of their team and could be completed by the July software release. Note, this is in no way a delivery commitment.]

Only SCM is required to send data to the Data Capture Service for this initial set of requirements. Gabbo need not be modified.

SCM will not send data unless the EULA has been accepted and WiFi is connected (i.e., we will not collect AirPlay/Bluetooth data unless the user has accepted our license terms, and we will not cache data while disconnected from WiFi).

The Data Capture Service will maintain a “blacklist” that allows Bose to specify, for particular regions and particular accounts, which data will not be transmitted from SCM to the Data Capture Service. Each time the SCM comes out of network standby, it requests the appropriate blacklist from the server. The blacklist specifies the data fields that the SCM should **not** transmit. These data fields are structured, so for example, if the blacklist specifies “track”, the SCM will not transmit any of the subfields for the track data structure (i.e., artist, album, trackTitle, etc.).

Each event (button press, transport event), the SCM will immediately queue an asynchronous POST task. In the event of a failure, the SCM will retry at least once. In the event of failed retry, the SCM will throw away the data, but keep a flag that some amount of data has been lost. The SCM will continue retrying periodically to send a failure event until that event is successfully transmitted.

For each event, we will collect the following data set:

event:

               transportEvent:started|ended|skipforward|skipback|stopped|paused|unpaused

               buttonPress: remote buttonID (we need a lookup table to understand what the buttons are)

               failure: (an event of this type indicates that some number of events have been lost)

               eventOrigin: gabbo|remote|device

               volume:Final Volume in volume change session

httpdata: [added by server from incoming request]

               address:IP Address

               timestamp: GMT Time/Date

track:

               artist: Artist Name

               album: Album title

               trackTitle: Track title

               sourceProvider:[Source Provider ID]

               sourceName: [Text of station name, pandora station, etc.]

               location:[source location ID as stored in recents]

               trackType: [Advert, song, talk, etc. if available]

device:

               deviceID: [Mac address]

               boseID: BoseID

               playEverywhere:client|master [only if multi-room enabled & active]

               volume: current device volume level