APRX software Requirement Specification

This is *Requirement Specification* for a software serving in Amateur Radio APRS service.

Reader is assumed to be proficient with used terminology, and they are not usually explained here.

Purpose:

This describes algorithmic, IO-, and environmental requirements for a software doing any combination of following three tasks related to APRS service:

1. Listen on messages with a radio, and pass them to APRSIS network service
2. Listen on messages with a radio, and selectively re-send them on radio
3. Receive messages from APRSIS network, and after selective filtering, send some of them on radio

Usage Environments:

The **aprx** software can be used in several kinds of environments to handle multiple tasks associated with local APRS network infrastructure tasks.

1. License-free Receive-Only (RX) i-gate, to add more “ears” to hear packets, and to pipe them to APRSIS
2. Licensed bidirectional i-gate, selectively passing messages from radio channels to APRSIS, and from APRSIS to radio channels, but not repeating packets heard on a radio channel back to a radio channel.
3. Licensed bidirectional i-gate plus selectively re-sending of packets heard on radio channels back to radio channels
4. Licensed system for selectively re-sending of packets heard on radio channels back to radio channels, and this without bidirectional i-gate service.
5. Licensed system for selectively re-sending of packets heard on radio channels back to radio channels, and doing with with “receive only” i-gate, so passing information heard on radio channel to APRSIS, and not the other way at all.

In more common case, there is single radio and single TNC attached to digibeating (re-sending), in more challenging cases there are multiple receivers all around, and very few transmitters. Truly challenging systems operate on multiple radio channels. As single-TNC and single-radio systems are just simple special cases of these complex systems, we consider the complex ones:

1. 3 different frequencies + in use, traffic is being relayed in between them, and the APRSIS network.
2. On each frequency there are multiple receivers, and one well placed transmitter.

Treatment rules:

Generally: All receivers report what they hear straight to APRSIS, after small amount of filtering of junk messages, and things which explicitly state that they should not be sent to APRSIS.

General rules for these receiving filters are described here:

http://www.aprs-is.net/IGateDetails.aspx

Gate all packets heard on RF to the Internet EXCEPT

1. 3rd-party packets (data type '}').
   3rd-party packets should have all before and including the data type stripped and then the packet should be processed again starting with step 1 again.
2. generic queries (data type '?').
3. packets with TCPIP, TCPXX, NOGATE, or RFONLY in the header (last 2 are optional).

Gate message packets and associated posits to RF if

1. the receiving station has been heard within range within a predefined time period (range defined as digi hops, distance, or both).
2. the sending station has not been heard via RF within a predefined time period (packets gated from the Internet by other stations are excluded from this test).
3. the sending station does not have TCPXX, NOGATE, or RFONLY in the header.
4. the sending station has not been heard via the Internet within a predefined time period.
   A station is said to be heard via the Internet if packets from the station contain TCPIP* or TCPXX* in the header or if gated (3rd-party) packets are seen on RF gated by the station and containing TCPIP or TCPXX in the 3rd-party header (in other words, the station is seen on RF as being an IGate).

Gate all packets to RF based on criteria set by the sysop (such as callsign, object name, etc.).

With more advanced looking inside frames to be relayed, the digibeater can use filtering rules, like “packet reports a position that is too far from my service area.”

From multiple receivers + single (or fewer) transmitter(s) follows, than when a more usual system does not hear what it sent out itself, this one will hear, and its receivers must have a way to ignore a frame it sent out itself a moment ago.

Without explicit “ignore what I just sent” filtering, an APRS packet will get reported twice to APRSIS:

rx -> igate-to-aprsis + digi -> tx -> rx -> igate-to-aprsis + digi (dupe stops)

Digibeating will use common packet duplication testing to sent similar frame out only once per given time interval (normally 30 seconds.)