8/31/2015

API design for the telemetry packet layout code

There is a python script that uses a spreadsheet to define the layout of telemetry packets. This script will auto-generate code that can be used by FSW to retrieve the data from the tlm database and to form the actual packet that is handed off to downlink.

There are requirements for the following elements:

- A means for FSW to get a list of packets that are to be sent this cycle
- A means for FSW to get the telemetry database identifiers associated with the fields in a specific packet
- A means for FSW to get the values retrieved from the telemetry database correctly inserted into the packet. The database returns a serializable, it may not be clear how this gets deserialized...
- Access to the formed packet so that it can be transmitted

There is an open question as to the header of the packet. It contains a timestamp but the format is TBD. Also TBD is any other information such as possible the packet ID.

A base class (see TlmPacketDefBase.hpp) will be manually coded that contains the following:

- A static method that takes as input a cycle number and returns an array of references to class instances for each packet in this cycle. These class instances are subclasses of the base class. Returning a list of class instances avoids the issue of having the caller know how to instantiate the subclass for a specific packet type. An alternate approach would be to provide a factory which takes a packet identifier and returns a class instance but this has a memory ownership or multiple copy issue.

- A pure virtual method that returns a list of field identifiers and database identifiers for the variable fields in the telemetry packet. The field identifiers are used to pass values for insertion into the packet and the database identifiers are used to retrieve values from the telemetry database. This method is implemented in the auto-generated subclass.

- A pure virtual initialization function that resets the packet to an empty state. This is used before populating the packet.

- A pure virtual method that allows a value to be inserted into the packet. This takes the field identifier and the serializable that was returned from the telemetry database.

- A pure virtual method that returns the address, size and subaddress of the packet.

The autocoder will generate a file containing the implementation for this base class and instances of classes inheriting from the packet base class for each telemetry packet type in an anonymous name space. All generated classes and data will be in an anonymous namespace and will have no global visibility.

In the code that forms and dispatches telemetry packets, an instance of the base class is declared either

statically or as an automatic. The list of packets for a given cycle number is fetched using the static member function. For each packet in the list:

Call the init function
Get the list of fields
For each field
    Get the database identifier for the telemetry item
    Fetch the current value from the telemetry database
    Insert the value in the packet
    Get the packet size / address and subaddress
Pass the packet size, address and subaddress to the 1553 code for transmission