Automatic Generation of SDM Application Source Code from xTEDS

24th Annual AIAA / USU Conference on Small Satellites
Logan, UT
August 9 - 12, 2010

Jacob Christensen and Scott Cannon
Utah State University, Logan, Utah, 84321, USA

Bryan Hansen
Space Dynamics Laboratory, North Logan, Utah, 84341, USA

Jim Lyke
AFRL / RVSE, Albuquerque, NM, 87117, USA
Introduction

- The Satellite Data Model
  - PnP for satellites
  - Self-discovery & auto-configuration

- eXtensible Electronic Data Sheet (xTEDS)
  - Contains information about the data products, service and commands for a device

- SDM Application
  - Searches, selects and consumes data
  - Provides data and services
The Problem

- Developing an SDM App is a non-trivial task
  - SDM Messaging
  - SDM Registration
  - Query, select, and subscribe to data
  - Manage data subscriptions

- Creates extra steps in the development of the flight software
Our Solution

- Write the code for the application developers
- Create a tool that will generate much of the code
- How much code can we automatically generate?
xTEDS

-xTEDS contain lots of information
  - Data products
  - Timing constraints
  - Command message
  - Message formats

Let’s use this information and generate code from it
Example xTEDS

```xml
<Interface name="ExampleInterface" id="1" description="An example">
    <Variable name="celcius" kind="temperature" format="UINT16" />
    <Notification>
        <DataMsg name="GetTemperature" id="2" msgArrival="PERIODIC" msgRate="1.00">
            <VariableRef name="celcius" />
        </DataMsg>
    </Notification>
    <Command>
        <CommandMsg name="SetTemp" id="1">
            <VariableRef name="celcius" />
        </CommandMsg>
    </Command>
    <Request>
        <CommandMsg name="GetTemp" description="" id="3">
            <VariableRef name="celcius" />
        </CommandMsg>
        <DataReplyMsg name="TempData" id="4">
            <VariableRef name="celcius" />
        </DataReplyMsg>
    </Request>
</Interface>
```
SDM App Code Architecture

- Library code
  - Sending messages
  - SDM registration
  - Listen and queue incoming messages

- Automatically Generated Code
  - Dataflow
  - Subscription management and data production
  - Query, select and subscribe to data

- Regular Hand Written Code
  - Algorithm and data processing
  - Command handling
How do we get messages out of the queue and into the function that needs it?

Using the message descriptions in the xTEDS, a switch statement is setup to move data to the intended destination.

```c
if (messageManager.IsReady())
{
    switch (messageManager.GetMessage(buf))
    {
        case SDM_Subreqst:
            subReqstMsg.Unmarshal(buf);
            subManager.AddSubscription(subReqstMsg);
            break;
        case SDM_DeleteSub:
            deleteSubMsg.Unmarshal(buf);
            subManager.RemoveSubscription(deleteSubMsg);
            break;
        case SDM_Command:
            printf("SDMCommand msg received\n");
            CommandHandler(buf);
            break;
        case SDM_RegInfo:
            RegInfoHandler(buf);
            break;
        case SDM_Data:
            DataHandler(buf);
            break;
    }
}
```
Auto-generated Program Code

Data Production

xTEDS

Generated Code

```c
<Notification>
   <DataMsg name="GetTemperature" id="2" msgArrival="PERIODIC" msgRate="1.00">
      <VariableRef name="celcius" />
   </DataMsg>
</Notification>

curTime = GetCurTime();

if(curTime - GetTemperature_1_lastPubTime >= GetTemperature_1_msgDelay) {
    char buf_1_2[2];
    unsigned short celcius = Getcelcius1();

    PUT_USHORT(&buf_1_2[0], celcius);
    if(subManager.Publish(GetTemperature_1, buf_1_2, 2)) {
        GetTemperature_1_lastPubTime = curTime;
    }
}
```
SDM Application can get multiple responses to a query for data.
- i.e. a query for temperature

Setting up the query and selecting the most appropriate result is non-trivial.
- Perhaps the hardest part

Not enough time to go over it in 13 minutes.
- Come to a tech demo at the SDL @ 6:30 PM
- Check it out at https://pnpsoftware.sdl.usu.edu
SDM AppWizard

SDM App Wizard
Version 1.0.4

Create an ASIM Tester App
Create an SDM App Framework
SDM AppWizard

---

```cpp
#include "../common/message/SDMxTEDS.h"
#include "../common/message/SDMMessage_ID.h"
#include "../common/message/SDMData.h"
#include "../common/message/SDMCancelxTEDS.h"
#include "../common/MessageManager/MessageManager.h"
#include "../common/MessageManipulator/MessageManipulator.h"
#include "../common/Time/SDMTime.h"
#include "../common/SubscriptionManager/SubscriptionManager.h"
#include "../common/message/SDMSubregst.h"
#include "../common/message/SDMDeleteSub.h"
#include "../common/message/SDMCommand.h"
#include "../common/message/SDMSerreqst.h"

#include <string.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <fstream>

using namespace std;
```
Future Work and Conclusion

- SDM App Wizard does not use optional attributes
- Add more compiler / architecture options
- Upgrade the code to make it more robust

Extra work in developing a SDM Application is reduced by generated most of the code.

Automatic generation of SDM Application code saves time by decreasing the learning curve
Tech Demo

6:30 PM
SDL