Language and tools ensuring consistency in data for space systems

Maxime Perrotin
ESA/ESTEC
TEC-SWE

European Space Agency
ASN.1

- International, widely used standard (ISO and ITU-T)
- Simple text notation for precise and complete data type description
- But with an added value: the physical encoding rules (compact binary encoding, endianness-neutral, but also XML encoding, legacy encoding specifications).
- Separate the encoding rules from the types specification
A very simple yet powerful syntax

Dataview DEFINITIONS ::= BEGIN

-- From simple types
Thruster-index ::= INTEGER {1..10} -- Allowed: 1 to 10
Identifier ::= ENUMERATED { cpdu1, cpdu2 }

-- To complex data structures
TC ::= SEQUENCE {
   header PUS-header,
   payload Userdata,
   crc OPTIONAL
}

Userdata ::= CHOICE {
   tc-6-1 MemoryLoad,
   tc-6-2 ...
   ...
}

END
ASN.1 to ensure consistency

TM/TC Definitions (ASN.1)

- ASN.1 to SQL
- ASN.1 to Python
- ASN.1 to C and Ada
- ASN.1 to VHDL
- ASN.1 to TMTC GUIs
- ICD Generator

- System Database
- Test Scripts
- Flight Software (cat. B)
- FPGA
- Ground segment
- ECSS Doc.
How does it work?

ASN.1 Grammar → ASN.1 Compiler

- Type definitions in C
- Encoders and decoders in C

ASN.1 Compiler →

- Type definitions in Ada
- Encoders and decoders in Ada
The fields are not application semantics! They concern the binary encoding rules of the PDUs and should not be mixed with the protocol useful information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Length (bits)</th>
<th>Values</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>3</td>
<td>'000'</td>
<td>For the first version.</td>
</tr>
<tr>
<td>PDU type</td>
<td>1</td>
<td>'0' — File Directive</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>'1' — File Data</td>
<td></td>
</tr>
<tr>
<td>Direction</td>
<td>1</td>
<td>'0' — toward file receiver</td>
<td>Used to perform PDU forwarding.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>'1' — toward file sender</td>
<td></td>
</tr>
<tr>
<td>Transmission Mode</td>
<td>1</td>
<td>'0' — acknowledged</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>'1' — unacknowledged</td>
<td></td>
</tr>
<tr>
<td>CRC Flag</td>
<td>1</td>
<td>'0' — CRC not present</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>'1' — CRC present</td>
<td></td>
</tr>
<tr>
<td>Reserved for future use</td>
<td>1</td>
<td>set to '0'</td>
<td></td>
</tr>
<tr>
<td>PDU Data field length</td>
<td>16</td>
<td>set to '0'</td>
<td>In octets.</td>
</tr>
<tr>
<td>Reserved for future use</td>
<td>1</td>
<td>set to '0'</td>
<td></td>
</tr>
<tr>
<td>Length of entity IDs</td>
<td>3</td>
<td>set to '0'</td>
<td>Number of octets in entity ID less one; i.e., '0' means that entity ID is one octet. Applies to all entity IDs in the PDU header.</td>
</tr>
<tr>
<td>Reserved for future use</td>
<td>1</td>
<td>set to '0'</td>
<td></td>
</tr>
<tr>
<td>Length of Transaction sequence number</td>
<td>3</td>
<td>set to '0'</td>
<td>Number of octets in sequence number less one; i.e., '0' means that sequence number is one octet.</td>
</tr>
<tr>
<td>Source entity ID</td>
<td>variable</td>
<td></td>
<td>Uniquely identifies the entity that originated the transaction.</td>
</tr>
<tr>
<td>Transaction sequence number</td>
<td>variable</td>
<td></td>
<td>Uniquely identifies the transaction, among all transactions originated by this entity.</td>
</tr>
<tr>
<td>Destination entity ID</td>
<td>variable</td>
<td></td>
<td>Uniquely identifies the entity that is the final destination of the transaction's metadata and file data.</td>
</tr>
</tbody>
</table>
ASN.1 philosophy

- Keep only application-semantic data
- Tools will generate encoders and decoders to add the other fields

```
Packet-ty ::= SEQUENCE {
    version                 Version-ty,
    direction               Direction-ty,
    transmission-mode       Transmission-mode-ty,
    crc-flag                CRC-flag-ty,
    source-entity-id        Entity-id-ty,
    transaction-sequence-number Transaction-sequence-number-ty,
    destination-entity-id   Entity-id-ty,
    data                    Datafield-ty
}

Version-ty ::= INTEGER (0..7)

Direction-ty ::= ENUMERATED { toward-file-receiver, toward-file-sender }
```
Our ASN.1 compiler

- Developed and maintained by Neuropublic for ESA
- Free software
- Features:
  - Generates safe and optimized C and Spark/Ada code (fast, low memory footprint)
  - Automatically generates test cases for a given grammar
  - Generates ICDs documents in HTML format
  - Supports customized (legacy) encodings (e.g. PUS format)
  - API and tools to interface ASN.1 with SDL, Simulink, SCADE, VHDL, SQL, and Python
Legacy encodings

- ACN allows to specify legacy encodings
- It can be used to describe the binary format of PUS packets, leaving the interesting part only (payload data) in the ASN.1 side.

\[
\text{MySeq} ::= \text{SEQUENCE} \{
\text{alpha INTEGER},
\text{gamma REAL OPTIONAL}
\}
\]

\[
\text{MySeq[]} \{\text{alpha []},
\text{beta BOOLEAN []},
\text{gamma [present-when beta, encoding IEEE754-1985-64]}
\}
\]
Apply it to the PUS (1)

---

-- General Telecommand structure
---

T-telecommand ::= SEQUENCE
{
  packet-header            TC-packetHeader,
  data-field-header       T-to-dataFieldHeader,
  application-data        T-to-applicationData,
  crc                      T-uint16
}

---

-- Telecommand application data
---

-- List of all available TCs categorized by their respective pus(-sub)types
-- Definition of actual payload data is done in respective types below
-- In the AON-file this type is used to automatically assign the pustype and subtype fields
-- In encoding and determine the packet type from pustype and subtype in decoding
-- Types defined as T-NULL have no actual payload data besides the fields
-- for pustype and subtype
T-to-applicationData ::= CHOICE
{
  tc-3-27-update-hk-period    TC-UPDATE-HK-PERIOD,
  tc-6-2-load-memory          TC-LOAD-MEMORY,
  tc-6-5-dump-memory          TC-DUMP-MEMORY,
  tc-6-9-check-memory         TC-CHECK-MEMORY,
  tc-6-129-transfer-image     TC-TRANSFER-IMAGE,
  tc-210-3-reset-dpu          T-NULL,
  tc-210-4-enable-watchdog    T-NULL, -- T-NULL is for TCs which don't have any applicationData
  tc-210-5-disable-watchdog   T-NULL, -- but only service type and subtype. Still they have to
  tc-210-6-boot-tasw          T-NULL, -- Appear in the list of valid commands. T-NULL ensures that 0 bits will be encoded
  tc-197-2-report-boot       T-NULL
}
Apply it to the PUS (2)

--- Table which maps the pusType and subtype to the corresponding
--- packet payload data
T-tc-applicationData(T-uint8:pusType, T-uint8:pusSubType) []
{
  tc-3-27-update-hk-period [present-when pusType== 3 pusSubType== 27 ],
  tc-6-2-load-memory [present-when pusType== 6 pusSubType== 2 ],
  tc-6-5-dump-memory [present-when pusType== 6 pusSubType== 5 ],
  tc-6-9-check-memory [present-when pusType== 6 pusSubType== 9 ],
  tc-6-129-transfer-image [present-when pusType== 6 pusSubType== 129 ],
  tc-210-3-reset-dpu [present-when pusType==210 pusSubType== 3 ],
  tc-210-4-enable-watchdog [present-when pusType==210 pusSubType== 4 ],
  tc-210-5-disable-watchdog [present-when pusType==210 pusSubType== 5 ],
  tc-210-6-boot-lasw [present-when pusType==210 pusSubType== 6 ],
  tc-197-2-report-boot [present-when pusType==197 pusSubType== 2 ],

---Telecommand application data

--- List of all available TCs categorized by their respective pus/subtypes

--- Definition of actual payload data is done in respective Types below
--- In the ACN-TShN this type is used to automatically assign the pusType and subtype fields
--- in encoding and determine the packet type from pusType and subtype in decoding
--- Types defined as T-NULL have no actual payload data besides the fields
--- for pusType and subtype.

<table>
<thead>
<tr>
<th>No.</th>
<th>Field</th>
<th>Comment</th>
<th>Present.</th>
<th>Type</th>
<th>Constraint</th>
<th>Min Length</th>
<th>Max Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td>pusType</td>
<td>TC-UPDATE-HK-</td>
<td>N.A.</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>PERIOD</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td>TC-LOAD-MEMORY</td>
<td>N.A.</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>TC-DUMP-MEMORY</td>
<td>N.A.</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td>TC-CHECK-MEMORY</td>
<td>N.A.</td>
<td>72</td>
<td>72</td>
</tr>
<tr>
<td>6</td>
<td>tc-3-27-update-hk-period</td>
<td></td>
<td>pusType=3 AND pusSubType=27</td>
<td>TC-UPDATE-HK- PERIOD</td>
<td>N.A.</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>tc-6-2-load-memory</td>
<td></td>
<td>pusType=2 AND pusSubType=2</td>
<td>TC-LOAD-MEMORY</td>
<td>N.A.</td>
<td>112</td>
<td>8080</td>
</tr>
<tr>
<td>8</td>
<td>tc-6-5-dump-memory</td>
<td></td>
<td>pusType=5 AND pusSubType=5</td>
<td>TC-DUMP-MEMORY</td>
<td>N.A.</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>9</td>
<td>tc-6-9-check-memory</td>
<td></td>
<td></td>
<td>TC-CHECK-MEMORY</td>
<td>N.A.</td>
<td>72</td>
<td>72</td>
</tr>
<tr>
<td>10</td>
<td>tc-6-129-transfer-image</td>
<td></td>
<td>pusType=129 AND pusSubType=129</td>
<td>TC-TRANSFER- IMAGE</td>
<td>N.A.</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td></td>
<td></td>
<td>T-NULL</td>
<td>N.A.</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>tc-210-3-reset-dpu</td>
<td></td>
<td>pusType=210 AND pusSubType=3</td>
<td>T-NULL</td>
<td>N.A.</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>tc-210-4-enable-watchdog</td>
<td></td>
<td>pusType=210 AND pusSubType=4</td>
<td>T-NULL</td>
<td>N.A.</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>tc-210-5-disable-watchdog</td>
<td></td>
<td>pusType=210 AND pusSubType=5</td>
<td>T-NULL</td>
<td>N.A.</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>tc-210-6-boot-lasw</td>
<td></td>
<td>pusType=210 AND pusSubType=6</td>
<td>T-NULL</td>
<td>N.A.</td>
<td>80</td>
<td>80</td>
</tr>
<tr>
<td>16</td>
<td>tc-197-2-report-boot</td>
<td></td>
<td>pusType=197 AND pusSubType=2</td>
<td>T-NULL</td>
<td>N.A.</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
And use the code

typedef struct {
enum {
    T_tc_applicationData_NONE,
    Tc_3_27_updatehk_period_PRESENT,
    Tc_6_2_load_memory_PRESENT,
    Tc_6_5_dump_memory_PRESENT,
    Tc_6_9_check_memory_PRESENT,
    Tc_6_129_transfer_image_PRESENT,
    Tc_210_3_reset_dpu_PRESENT,
    Tc_210_4_enable_watchdog_PRESENT,
    Tc_210_5_disable_watchdog_PRESENT,
    Tc_210_6_boot_lasm_PRESENT,
    Tc_197_2_report_boot_PRESENT,
} kind;
union {
    TC_UPDATE_HK_PERIOD tc_3_27_updatehk_period;
    TC_LOAD_MEMORY tc_6_2_load_memory;
    TC_DUMP_MEMORY tc_6_5_dump_memory;
    TC_CHECK_MEMORY tc_6_9_check_memory;
    TC_TRANSFER_IMAGE tc_6_129_transfer_image;
    T_NULL tc_210_3_reset_dpu;
    T_NULL tc_210_4_enable_watchdog;
    T_NULL tc_210_5_disable_watchdog;
    TC_BOOT_LASM tc_210_6_boot_lasm;
    T_NULL tc_197_2_report.boot;
} u;
} T_tc_applicationData;

#define T_tc_applicationData_REQUIRED_BYTES_FOR_ENCODING 1007
#define T_tc_applicationData_REQUIRED_BITS_FOR_ENCODING 8049
#define T_tc_applicationData_REQUIRED_BYTES_FOR_ACN_ENCODING 1010
#define T_tc_applicationData_REQUIRED_BITS_FOR_ACN_ENCODING 8080
#define T_tc_applicationData_REQUIRED_BYTES_FOR_XER_ENCODING 2272

void T_tc_applicationData_Initialize(T_tc_applicationData* pVal);
flag T_tc_applicationData_IsConstraintValid(const T_tc_applicationData* val, int* pErrCode);
flag T_tc_applicationData_ACN_Encode(const T_tc_applicationData* val, BitStream* pBuffer, int* pErrCode, flag bCheckConstraints);
flag T_tc_applicationData_ACN_Decode(T_tc_applicationData* pVal, BitStream* pBuffer, int* pErrCode, T_uint8 pUsrType, T_uint8 pUsrSubType);
#endif
#define ERR_T_tc_applicationData_unknown_choice_index 1037 /*/
SDL and ASN.1

```
DataModel DEFINITIONS ::= BEGIN

.. Input Types
  Digital-Inputs ::= SEQUENCE {
    sw-cmd BOOLEAN,
    sw-grpper BOOLEAN
  }

  Analog-Inputs ::= SEQUENCE (SIZE(16)) OF REAL (0.0 .. 6.0)

.. Output Types
  VR-Model-Output ::= SEQUENCE {
    x1 REAL (-1000 .. 1000),
    y1 REAL (-1000 .. 1000),
    z1 REAL (-1000 .. 1000),
    p1 REAL (-1000 .. 1000),
    q1 REAL (-1000 .. 1000),
    x2 REAL (-1000 .. 1000),
    y2 REAL (-1000 .. 1000),
    z2 REAL (-1000 .. 1000),
    p2 REAL (-1000 .. 1000),
    q2 REAL (-1000 .. 1000),
    x3 REAL (-1000 .. 1000),
    y3 REAL (-1000 .. 1000),
    z3 REAL (-1000 .. 1000),
    p3 REAL (-1000 .. 1000),
    q3 REAL (-1000 .. 1000),
    j retard SEQUENCE (SIZE(16)) OF REAL (-1000 .. 1000)
  }

  VR-Arm-Configuration ::= SEQUENCE {
  ...}

PROCDURE control_law
  IN in_analog Analog-Inputs,
  IN in_digital Digital-Inputs,
  IN/OUT outvr VR-Model-Output
  ... call;
```

```
SDL, MSC and ASN.1
MSC and ASN.1
TASTE relies on ASN.1 to ensure consistency of data at each level of the process: Engineering, processing, testing, documentation, communication, data storage and retrieval.
ASN.1 to SQL magic

- Use the same ASN.1 model to create SQL schemas → keep consistency (one SQL table per ASN.1 data type is created by the toolchain, automatically)

- Use case: telecommand/telemetry storage
  - Describe TM/TC data format in ASN.1 and ACN
  - Use C/Ada binary encoder/decoders in flight code
  - Use ICD generator to document format at binary level
  - Pick TC/Store TM in the SQL database for post-processing – field format is correct by construction

- Very flexible: using SQLAlchemy to be compatible with Oracle, SQLite, PostgreSQL...

- Python interface
A simple API

MyInt ::= INTEGER (0..20)

# Can work with any DB. Here is an example with PostgreSQL
engine = create_engine(
    'postgresql+psycopg2://taste:tastedb@localhost/test', echo=False)

# Create data using the ASN.1 Python API
a = MyInt()
a.Set(5)

# Add the value to the SQL table called MyInt
aa1 = MyInt_SQL(a)
aid1 = aa1.save(session)
A simple API – Retrieve data

# Data is retrieved using SQL queries, or SQLAlchemy API

# Retrieve ALL records in the MyInt table
all_values = self.session.query(MyInt_SQL)

for record in all_values:
    # The magic: data is transparently converted back to ASN.1
    print record.asn1.Get()

Query data with the full power of databases. It will be converted automatically to ASN.1 structures.

Use case:
Query all TC with type=XX and subtype=YY (1 line of code)
Select the ones you are interested in
Encode them with ASN.1/ACN to a PUS packet (1 line of code)
Send them to the satellite (1 line of code)
Check the results

- Demo of the complete features in /home/assert/tool-src/DMT/tests-sqlalchemy
- Run make (password for the db is tastedb)
- Run pgadmin3