Why modelling is the most advanced in Telecommunication

Emmanuel Gaudin has a technical background and developed protocol stacks in SDL. He joined a modeling tool vendor in 96 as a Field Application Engineer and as a trainer to finally become technical director of the French branch. Based on that experience, he started PragmaDev in 2001 to develop an SDL-RT tool.

This article will focus on the usage of modelling technologies in the telecommunication domain in order to demonstrate how advanced it is compared to other domains.

Conformance is not an option

A telecommunication system is a set of heterogeneous devices that communicate with each other. In order to do so, the devices **must** comply to the same protocol. Technically speaking that covers 2 complementary aspects:

- a **static** interface which is the format of the information that is exchanged from one device to the other,
- a **dynamic** interface which defines the correct sequences of exchange of information.

Conformance to both aspects of a protocol is critical for telecommunication systems, and that requires the protocol definitions to be as clear as possible and non ambiguous.

Standardization body

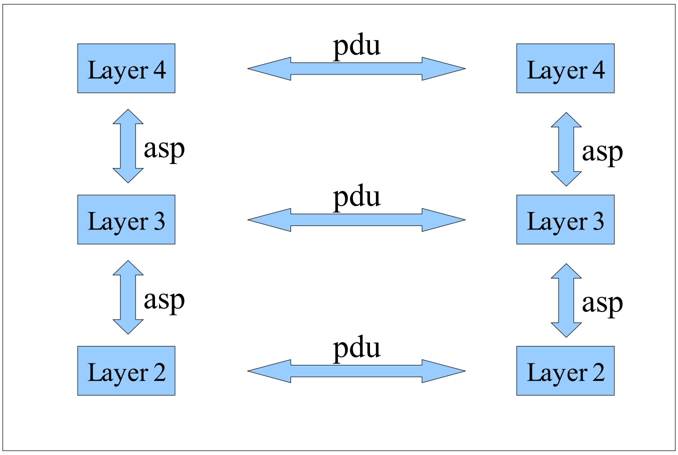
Protocols are defined by standardization bodies that publish recommandations on both these aspects. Historically, each area of the world had its own standardization body such as ETSI in Europe, ARIB in Japan, or ATIS in the US. At that time, because the standards were different, that meant a device such as a mobile phone in one area could not work in another area. Thanks to globalization, the different standardisation bodies are now working together to publish worldwide standards such as the 3G or the upcoming LTE (Long Term Evolution) which is one of the candidate for the 4G technology.

In order to produce clear and understandable standards, the European standardisation body -ETSI- has published a set of recommandations to "*Make Better Standards*" with a set of technologies such as SDL (Specification and Description Language), ASN.1 (Abstract Syntax Notation One), ECN (Encoding Control Notation) and TTCN-3 (Testing and Test Control Notation). These recommandations are available on ETSI web site: http://www.etsi.org/WebSite/Technologies/ProtocolSpecification.aspx.

This article will discuss how each of these technologies help producing better standards in the telecommunication industry.

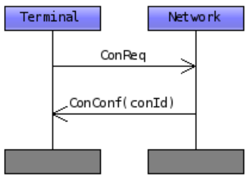
Dynamic interface

A protocol describes the exchange of messages from one layer to a remote layer of the same level. This interface is called PDU for Protocol Data Unit.



This is actually a logical view of the protocol since a layer is in practice exchanging information with the layer above or the layer below. This "vertical" interface is called ASP Abstract Service Primitive. So PDUs are sent to the remote layer through ASPs, but a protocol definition describes the exchange of PDUs.

The dynamic description of the logical view of the protocol is made with a bunch of scenarios. ETSI recommands to use MSC (Message Sequence Charts) for that purpose. Let's take a simple example of a layer requesting a connection and describe it in an MSC.



In the scenario above, messages are exchanged between a terminal and the network. A message may include parameters; such as the *ConConf* message that comes with the *conId* parameter for example.

Most of the time, a telecommunication system is a distributed system where the terminal is a separated physical device from the network. It is most likely that it will have a different processor architecture and run a different operating system. It is therefore necessary to have a way to create a flow of information that is independent from the processor and from the operating system.

Static interface

In order to define a hardware independent flow of information, an abstract representation of data is used with the ASN.1 notation. ASN.1 stands for Abstract Syntax Notation One and is standardized by ITU-T. It describes data types to be understandable by humans covering basic types such as booleans or reals, and complex types such as structures or arrays. ASN.1 describes the types but not how they will be manipulated in a given language. The main interest is that ASN.1 defines encoding and decoding rules so that the data flow of information for a given type is independent of its implementaion on a given hardware structure. Apart from standardized encoding rules such as BER (Basic Encoding Rule) or PER (Packet Encoding Rule), there is also a notation to describe a specific encoding rule called ECN: Encoding Control Notation. Last but not least ASN.1 data types can be directly imported in SDL for specification, and TTCN for testing.

An ASN.1 example structure:

```
PersonRecord ::= SEQUENCE {
      name        IA5String,
      age         INTEGER,
      member      BOOLEAN
      }
```

An example value:

```
examplePerson PersonRecord ::= {
      name        "Smith",
      age         35,
      member      true
      }
```

XML encoding of the value:

```
<PersonRecord>
      <name>Smith</name>
      <age>35</age>
      <member>true</member>
</PersonRecord>
```

Protocol detailed behavior

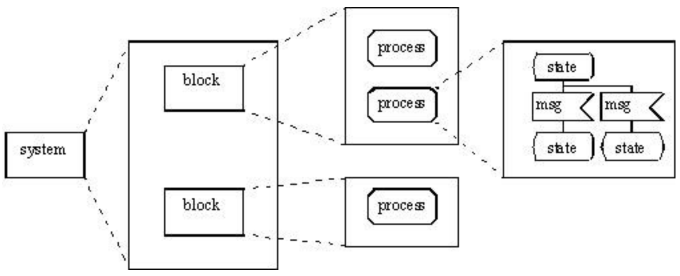
The protocol behavior described with MSCs only describes a few possible scenarios. Even though it is possible to describe alternatives, loops, or optional scenarios, MSCs can not describe all possible cases that could occur in a protocol. To do so it is relevant to describe a detailed and executable model of the protocol. In that manner, it is possible to simulate and verify its behavior so that the model is considered to be a reference model. For that purpose ETSI recommands a formal modeling language called SDL Specification and Description Language.

SDL has been designed in the first place in the 80's to describe telecommunication protocols, no need to state the language fits our requirements. But the language has evolved since then to follow up the different technology evolution such as for example the introduction of object orientation or the definition of a UML profile.

Technically speaking SDL has 4 complementary views we will briefly describe below.

- **Architecture**

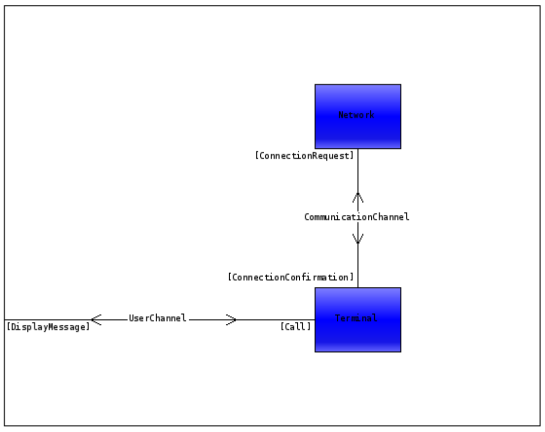
  A system is made of blocks that can be further decomposed in sub-blocks.



  The leaf of the architecture is the process which is basically a finite state machine with an implicit message queue associated to it.

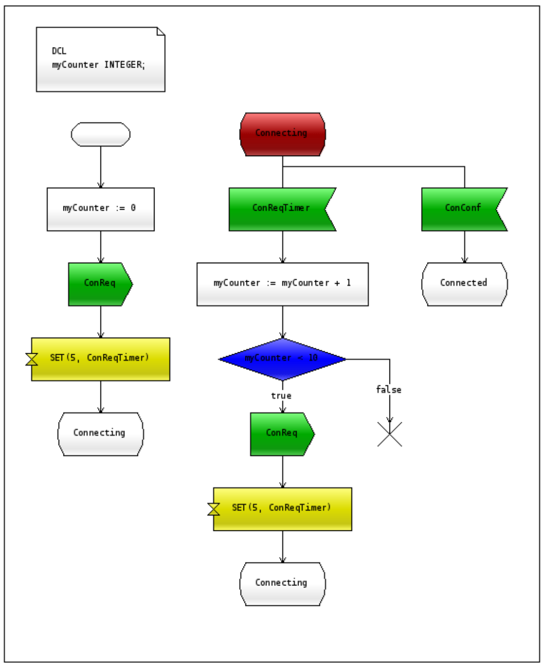
- **Communication**

  The flow of information exchanged between different blocks is described in the architecture in order to define the interfaces of the blocks.



- **Behavior**

  The detailed behavior is fully described graphically. Depending on how precise the specification is meant to be, it is possible to leave implementation details out.



  The example above shows a basic finite state machine: during initialization a *ConReq* message is sent out and a the *ConReqTimer* is started. The state machine goes to *Connecting* state and waits for the connection confirmation message *ConConf*, or the timer to go off. In the later case, the connection request is resent 10 times. If still unsuccessfull, the process is stopped.

- **Data**

  In order to make the specification complete and unambiguous, SDL embeds Abstract Data Types and a syntax to manipulate these data. As previously stated, it is preferable to import ASN.1 data types.

Protocol conformance

Now it is a question of making sure implementations of the same protocol conform to the standard published by ETSI. In the recent standards such as SIP, IP V6, 3GPP IMS... ETSI has published conformance test suite based on TTCN-3 language. To make sure it fits its needs, ETSI is also maintaining the TTCN-3 language, and it is standardized by ITU-T.

TTCN stands for Testing and Test Control Notation and was initiated by ISO as part of the Conformance testing methodology and framework document (9646-3). Where TTCN-2 was dedicated to communicating systems, TTCN-3 is suitable to test any kind of systems.

TTCN-3 could be seen as a classical textual programming language, but it has several aspects that make it a very powerful testing language.

- **Abstraction level**

  Like SDL, TTCN-3 includes the concept of messages and timers that are the basic services of a telecommunication system. It is then very easy to send or receive a message, to start or cancel a timer.

- **Templates**

  When testing a telecommunication system, most of the work is about verifying the messages exchanged contain the right information. TTCN-3 defines the concept of templates to easily verify a complex set of parameters are correct. TTCN-3 also supports optional parameters (parameters that might not be present) and can ignore some of the parameters (the parameter is present but its value is not relevant).

- **Alternatives**

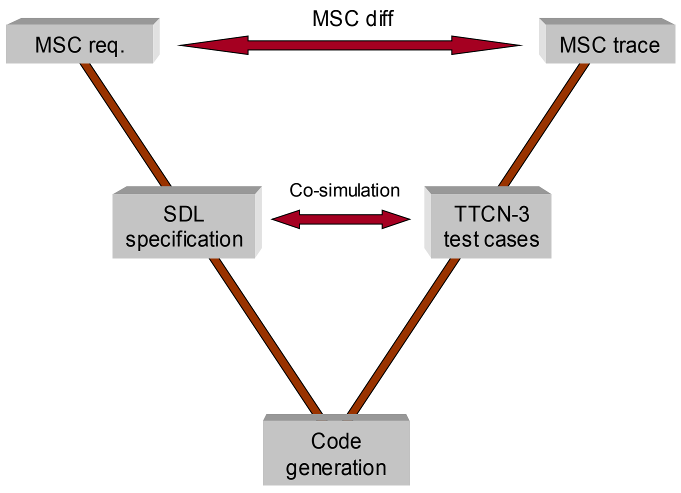
  Testing is about describing alternatives and setting a verdict depending on the different alternatives. TTCN-3 embeds the concept of alternatives whether they are based on asynchronous information such as the exchange of message or on synchronous information such as the value of a variable that is modified.

```
cEnv.send(ConReq);
ConReqTimer.start;
alt    {
       []cEnv.receive(ConConf){
             setverdict(pass);
             }
       []ConReqTimer.timeout{
             setverdict(fail);
             }
       }
```

  The example above shows a basic alternative in TTCN-3: a connection request *ConReq* is sent through the *cEnv* port and a timer *ConReqTimer* is started. In the alternative, either the *ConConf* response is received or the connection timer *ConReqTimer* goes off.

Model testing

Since SDL and TTCN-3 have the same abstraction level, the usage of both languages allow to test a specification model before implementation. Since both languages are formal, meaning complete and non ambiguous, it is possible to generate code for the protocol implementation out of the SDL specification, and to generate code for a tester out of the TTCN-3 test cases. A very early verification will in the end save a huge amount of time and effort.



Last but not least, ETSI publishes conformance test suites in TTCN-3 so that telecommunication manufacturers can make sure their implementation conforms to the standard.

Conclusion

Because telecommunication systems require by essence to conform to a common standard that covers static and dynamic interfaces, telecommunication standardization bodies as well as telecommunication equipment manufacturer have been using advanced modeling technologies such as MSC, ASN.1, SDL, and TTCN for years. In fact these technologies cover the whole development cycle from requirements, specification, design, and test. That is why it should definitely be considered when developing communicating systems.