Notes on openETCS API conf call (2014-06-05)

Attendees

- **DMe** : David Mentré, Mitsubishi Electric (Rennes) for 13 years, mainly involved in WP7 and in some tasks of WP3. In charge of the specification and implementation of the API between the EVC ETCS application and the on-board.
- **NBo** : Nicolas Boverie, Alstom (Charleroi) for 20 years (long experience in platform software), mainly involved in WP3 and highly concerned by the API since he wrote the "API Requirements for OpenETCS – V1.2" document.
- **NVL** : Nicolas Van Landeghem, ERSA (Haguenau), joined 1 month ago, mainly involved in WP5 and concerned by the API since ERSA provides an EVC demonstrator.

Summary/Notes

Note: we are starting from Alstom information as Alstom provides the most details information about its platform and NBo knows it very well.

NBo already proposed a document, based on its existing API. (openETCS API Requirement v1.2, https://github.com/openETCS/requirements/tree/master/D2.7-Technical_Appendix)

Also the existing API and documents deal with ADA code, which is not so critical since ADA to C transposition is easy to realize. Alstom already performed junctions between components sharing C and ADA codes, using the transposition mechanism. Maybe NBo will provide an example of such transposition to show the mechanism is feasible.

Concerning the services definition, two potential issues have been identified: (1) data flow definition and (2) timing requirements.

Data flow definition

Deutsche Bahn and Siemens are not willing to interface with data structures presented in the document, because these seem to be private. DB and Siemens would prefer having data structures closer to the SRS definitions.

Timing requirements

We assume that on-board computer is a distributed system. EVC, DMI, Odometry, JRU, … are independent computers communicating using asynchronous messages on a shared bus (see Fig 1

below).

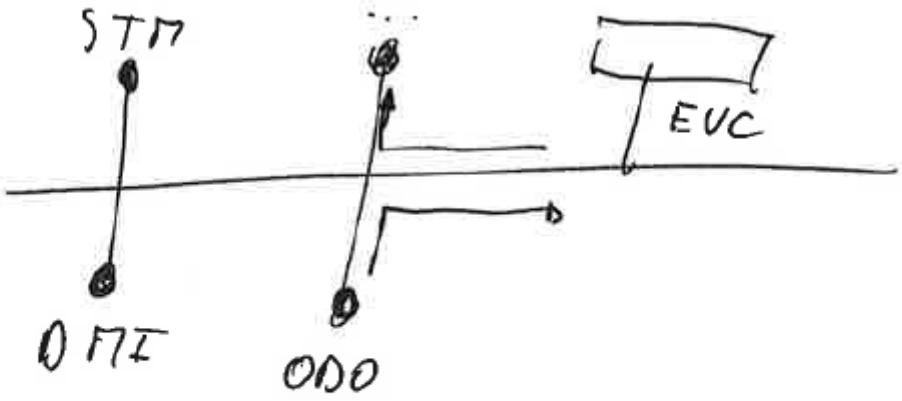


Fig. 1

Due to this architecture and due to varying processing time at each element level (DMI, EVC, ...) events received and sent from elements might be received in a different order into another element (Fig. 2).

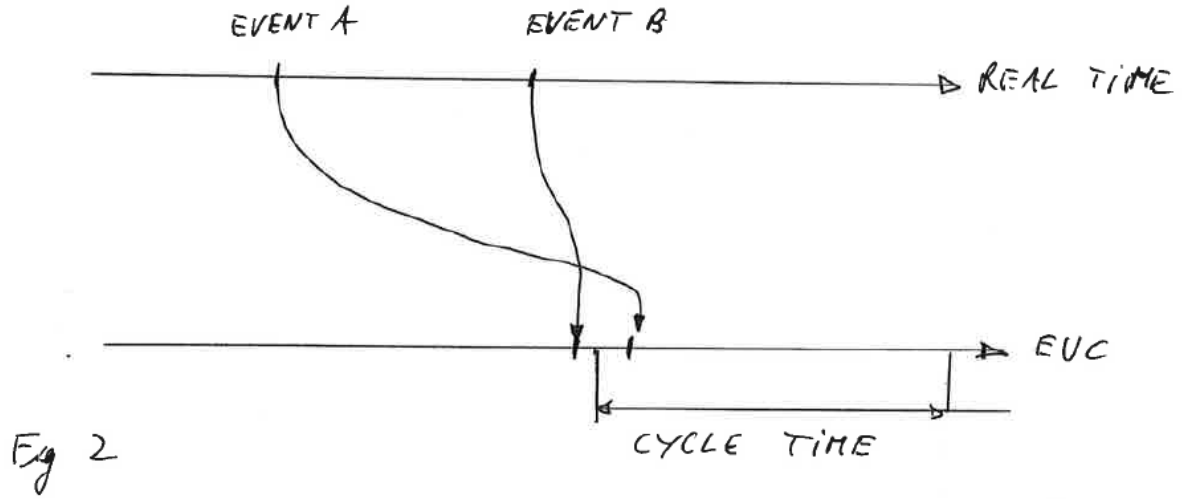


Fig 2

For example, on EVC an Event A (e.g. a Radio message) in above figure might be received **after** Event B (e.g. a balise message), even if in absolute time Event A was received on the train **before** Event B.

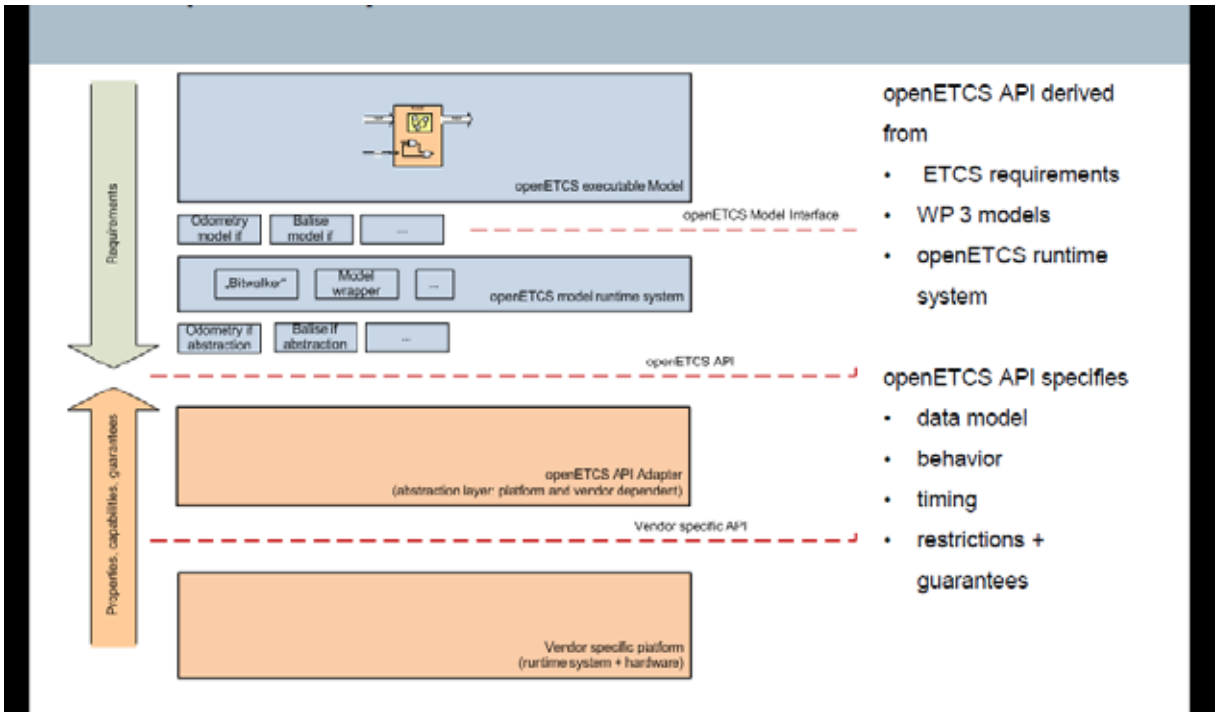
Some of those messages are timestamped (at source reception time for received events, at source emission time for sent events), allowing the application on EVC to re-order the messages taking into account (1) the absolute real-time ordering and (2) priority of messages. E.g. an RBC message starts to be received, in the meantime a balise message has been received, before the end of the RBC message reception.

Even if the JRU has its own clock for time stamping juridical messages, the internal clock of the application shall be a relative timestamp. The internal clock starts at 0 on system startup. The maximum time derivation allowed by the system is 0.1% regarding the real time.

On the contrary, SCADE model currently designed is synchronous and expects to receive all events in real-time order, with a strictly increasing internal clock ("SCADE Modelling Guide" document, §3.2).

Those two models seem currently incompatible.

One possible solution would be to temporarily store and sort received event in "openETCS model runtime system" (see blue box in figure below) to send them in real-time order to the SCADE model. However NBo states that by doing that it won't be possible to satisfy UNISIG performance requirements because there would be a too long delay in the processing of some events.



In current Alstom's proposal, the cycle time shall last 300 ms. DMe reports Deutsche Bahn and Siemens do not want such accurate cycle time definition. Moreover, the 300 ms is not fixed and could change a little due to event bursts.

Potentially solved issues

1. Ada current definition is not significant. It could (and will) be translated to C version, with easy Ada/C conversion

Identified current issues

1. Timing requirements are currently incompatible between SCADE model and Alstom's API definition

    a. Sub-issue regarding cycle time definition
    b. Sub-issue regarding ordering of events
2. The data flow definition in Alstom's API proposal is sometimes too far from SRS

Action points

- (NBo) If possible, provide an example of C API Service
- (DME, NVL) Read in detail review notes on Alstom's API 1.2 document (https://github.com/openETCS/requirements/blob/master/D2.7-Technical_Appendix/2014-05-13-Munich-Meeting/OETCS_API_review_2014_05_12.xlsx)
- (DME) Look more closely at SCADE model to understand timing requirements
- (NVL) What is ERSA expectations on API data flow definitions? Closer to SRS or current Alstom's definition are OK?
- (All) Precise the API stack, starting from Slide 6 of Uwe's presentation (see figure above)

Next meeting

Week 9-13 June, no date defined yet.