OETCS/WP3/D3.5 – 01.01

ITEA2 Project
Call 6 11025
2012 – 2015

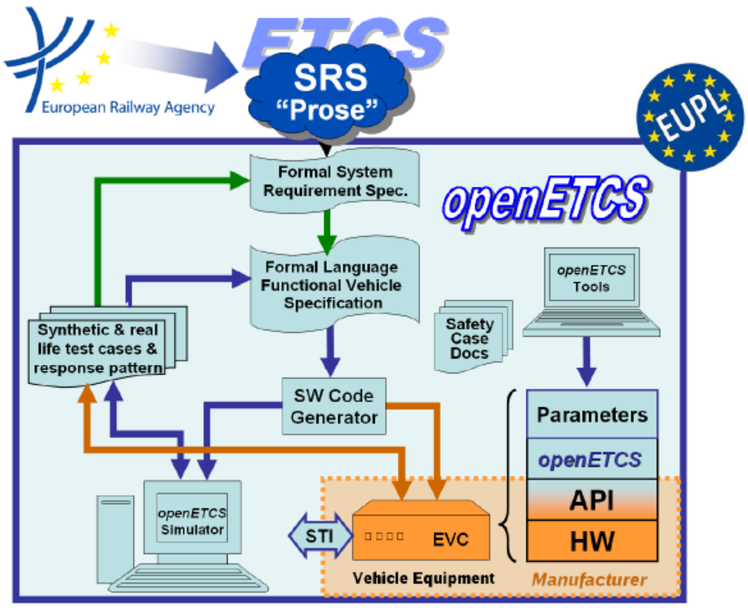
Work-Package 3: “Modeling”

openETCS System Architecture and Design Specification

Second Iteration: ETCS Kernel Functions

Baseliyos Jacob, Bernd Hekele, Uwe Steinke and Christian Stahl

September 2014



Funded by:

Federal Ministry of Education and Research

Région de Bruxelles-Capitale

GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, ENERGÍA Y TURISMO

openETCS

This page is intentionally left blank

Work-Package 3: “Modeling”

OETCS/WP3/D3.5 – 01.01
September 2014

openETCS System Architecture and Design Specification

Second Iteration: ETCS Kernel Functions

Document approbation

Lead author:	Technical assessor:	Quality assessor:	Project lead:
location / date	location / date	location / date	location / date
signature	signature	signature	signature
Baseliyos Jacob (DB Netz)	[assessor name] ([affiliation])	Izaskun de la Torre (SQS)	Klaus-Rüdiger Hase (DB Netz)

Baseliyos Jacob
DB Netz AG

Bernd Hekele
DB-Netz AG, Peter Mahlmann, Peyman Farhangi
Völckerstrasse 5
D-80959 München Freimann, Germany

Uwe Steinke
Siemens AG

Christian Stahl
TWT-GmbH

Description of work

Prepared for openETCS@ITEA2 Project

Abstract: This document gives an introduction to the architecture of the first openETCS iteration, the openETCS kernel functions. It has to be read as an add-on to the models in SysML, Scade and to additional reading referenced from the document.

Modification History

Version	Section	Modification / Description	Author
0.1	Document	Initial document providing the structure	Bernd Hekele
0.2	Sect. 9.4.2	initial contribution and some pretty printing	Christian Stahl
0.3	Sect.	taking over the author lead for the second iteration architecture	Baseliyos Jacob

Table of Contents

Figures and Tables

Figures

Tables

1 Introduction

1.1 Motivation

The openETCS work package WP3 aims to provide the kernel architecture and the design of the openETCS OBU software as mainly specified in UNISIG Subset_026 version_3.3.0.

The appropriate functionality has been divided into a list of functions of different complexity (see `https://github.com/openETCS/SRS-Analysis/blob/master/SystemAnalysis/List_Functions.xlsx`).

All these functions are object of the openETCS project and have to be analysed from their requirements and subsequently modelled and implemented. With limited manpower, a reasonable selection and order of these functions is required for the practical work that allows the distribution of the workload, more openETCS participants to join and leads to an executable—limited—kernel function as soon as possible.

While the first version of this document focuses on the first version of the limited kernel function, it is intended to grow in parallel to the growing openETCS software.

1.2 Objectives

The first objective of WP3 software shall be

- “Make the train run as soon as possible, with a very minimum functionality, and in the form of a rapid prototype.”

This does not contradict the openETCS goal to conform to EN50128.

- After a phase of prototyping, the openETCS software shall be implemented in compliance to EN50128 for SIL4 systems.

Additional goals for this document are

- Identification of the functions required for a minimum OBU kernel
- Architecture overview regarding the minimum OBU kernel
- Technical approach: Description of the proceeding and methods to be used
- Road map of the minimum OBU kernel functions
- Road map thereafter

Note: This document will be extended according to the progress of WP3.

1.3 History

1.4 Goals of the openETCS Modelling Work

1.4.1 Functional Scope: The Minimum OBU Kernel Function

The objective “Make the train run with a kernel functionality” shall be in terms of ETCS OBU translated into

- The Train should at the end be able to operate on a track equipped with balises and determines its position such as the Utrecht - Amsterldam line..

That means, for this very first step, the train shall not supervise the maximum speed nor activate the brakes. Instead, the minimum function set shall be limited to (see `https://github.com/openETCS/SRS-Analysis/issues/9`)

- Receive, filter and manage balise information received from track (see `https://github.com/openETCS/SRS-Analysis/issues/12`)
- Calculate the actual train position based on balise and odometry information (see `https://github.com/openETCS/SRS-Analysis/issues/8`)
- Calculate the distances between the actual train position to track elements in its front

A more detailed architectural breakdown of these functions is available as a SysML model at (see `https://github.com/openETCS/modeling/tree/master/model/sysml`.

In addition, the work on this minimum functionality requires to be supported by

- The availability of the ETCS language as specified in Subset UNISIG Subset_026, chapters 7 and 8
- The abiltiy to link intermediate and final results with the requirements of the ETCS specification (subset_026, ...)
- The usability of a data dictionary (see `https://github.com/openETCS/dataDictionary`)

These supporting prerequisites are under construction and therefore not completely operable actually. How to deal with these restrictions, will be outlined in chapter ???

1.4.2 Actual Status

Some first analysis steps for the required minimum functionality have been performed as results from the SRS-Analysis task force. These results are available at `https://github.com/openETCS/SRS-Analysis`.

Furthermore a first iteration on a part kernel functio is available on at `https://github.com/openETCS/Modelling`.

Further Architecture examples has been by Alstom and NS. Alstom document `https://github.com/openETCS/SRS-Analysis`

1.4.3 Practical Approach

The architecture and design of the minimum OBU kernel shall be developed in consideration of the actual status, restricted prerequisites and limited resources as follows.

1.5 Glossary and Abbreviations

1.6 References

SRS-Subset 26
QA-Plan: D1.3.1
Process: D2.3
Methods: D2.4
API: D2.7

2 Functions ERTMS/ETCS

2.1 introduction

The ERTMS / ETCS system was developed with a view to interoperability of trains on the different European rail networks. It is divided into "tracks" - and "board" finishes and shall establish a mutual message operation, by beacons or through a "radio" - The transmission system (in this case a mobile telephone network GSM-R) is performed. It defines several operating levels, and the system must also interfaces with the existing monitoring systems of the trains (using STM) have. The ERTMS / ETCS system provides the transport operator (the track) the choice of conditions concerning the use and operation. The train must therefore may go with different operating conditions on routes. Thus has the onboard equipment but must be implemented, to the interoperability of the train to ensure on the other networks. These functions must therefore correspond to one standard: the SRS (version x.x.x).

application functions, which have two different species of origin: defined in the SRS: here one finds in particular the speed monitoring- and transfer functions; these functions must be implemented in full accordance with the SRS; they can in indeed be on any network on which the train is used; these functions are described below in Section x.x.x;

Moreover, there are functions to adapt to the train: so, for example, the processing a "separation distance" in the airborne equipment trigger: This is dependent on the distribution of functions between the Control monitoring equipment (which the ERTMS / ETCS), and the other CCS Systems.

3 The openETCS Architecture of the initial kernel functions

4 SRS Architecture

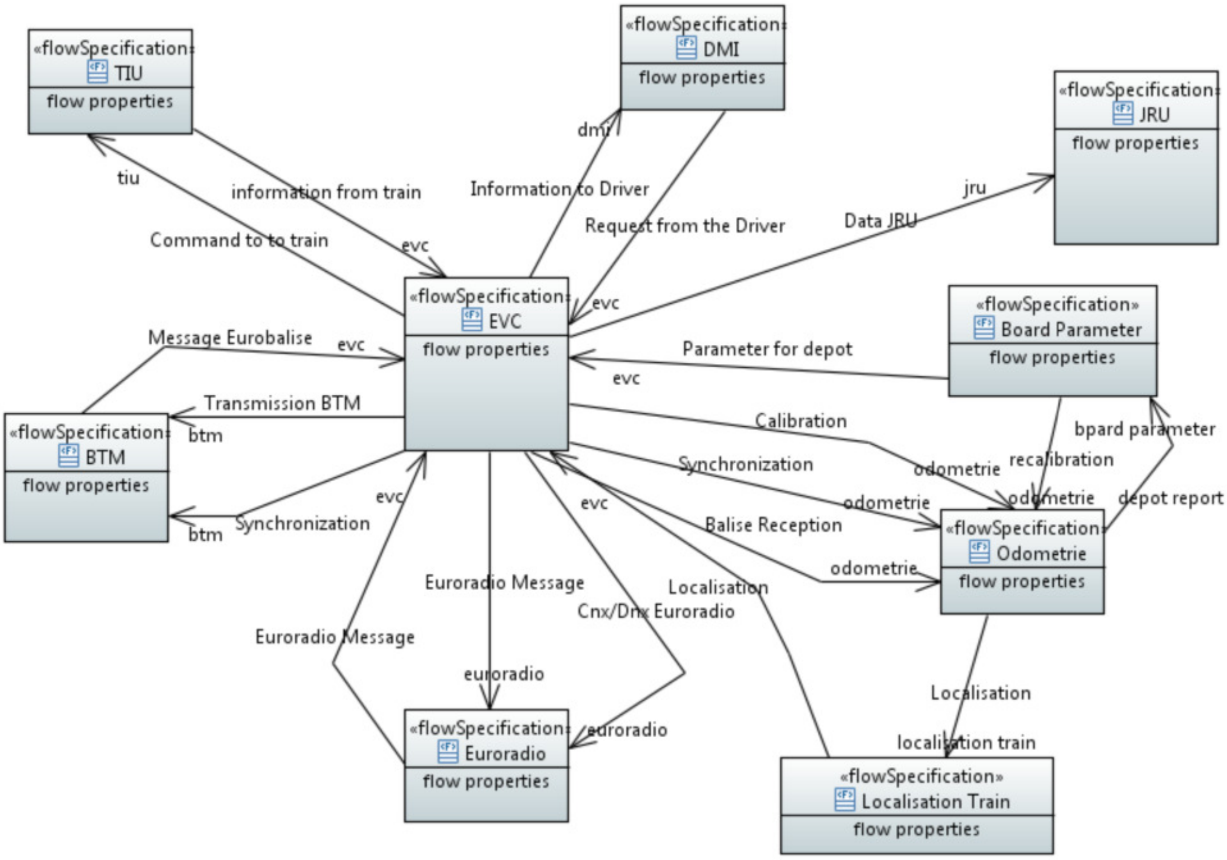


Figure 1. SRS architecture

5 Functional Breakdown

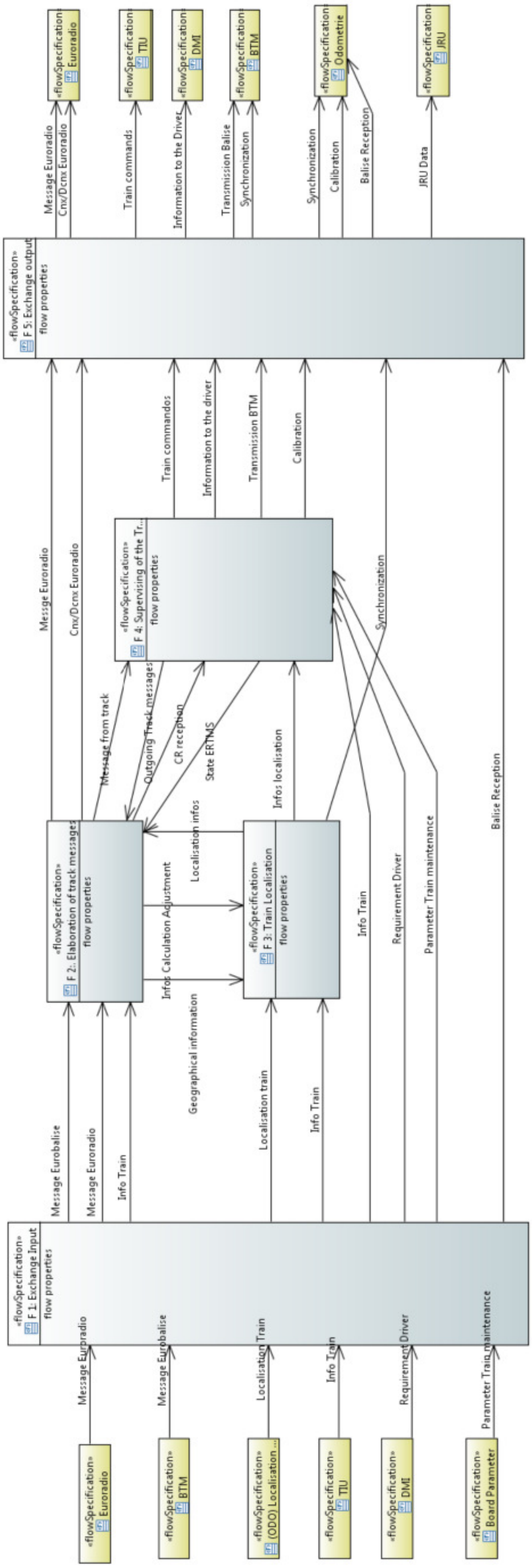


Figure 2. SRS Breakdown

6 Description of the SRS Functions

6.1 F1 Exchange input

See Figure 2 - Block F 1

Inputs:
“will be complete”

Outputs:
“will be complete”

Description: This function module manages the interaction with the functional modules at the input and output of equipment:
- TIU,
- DMI,
- BTM,
- EURORADIO,
- Parameterization board,
- Displacement measurement.

It complements their functionalities at the level: the protocols and exchange "safety layer" with the kernel external modules, with Exception of Wegmessungsmoduls (this manages its own exchange protocols. The summary of telepowering telegrams of the BTM module in a complete Eurobalise message a balise group with the same ID, with review of the Balisentelegramheader, management of the duplicated beacons and detecting the direction. In addition, the receipt of a Referenzbalise from the linked type, in addition to the Balise group, reported for the computational comparison of path measure modul.

6.2 F2 Elaboration track messages

See Figure 2 - Block F 2

Inputs:
“will be complete”

Outputs:
“will be complete”

Description: This function module provides the encoding and decoding of track messages that Management of the port and power off RBC, and the management of reference Beacons, which are based on the track messages.

6.2.1 F21: Receive Eurobalise Messages

SRS: § 3.4, § 3.6, § 3.16, § 3.17, § 4.8

Inputs:
“will be complete”

Outputs:
“will be complete”

Description: This function module provides the summary of telepowering information and the review of coherence the content of telepowering message. This module separates the eurobalise message as follows:

- The Balise telegram heads for checking the consistency of the different Telegrams, the management of the duplicated beacons and the acquisition of meaning.

- Then the information that the balise group, their reading direction and are Wegmessungsposition where the first balise group was read, in Baliseninformationen for monitoring the function "Manage the eurobalises" summarized (see the term Identifzierung and meaning in this function), which the Continuation of reading via the information Referenzbalise with a Balisenposition in Kernel-reference document approved.

- The ETCS ETCS version and packages:

The first review is the version ETCS ETCS track with the version Zugbord (Version 1.x Class1 =) to compare; these must be compatible (that means 1.y) to continue processing (a Balise version of the type is 0.y as incompatible considered, but they must not generate a negative reception report).

The second review is in compliance with the ETCS grammar in the resulting Packets with a packet 255, which terminates the beacon information of each of the group.

The reception of a packet 245 (standard package) in a eurobalise) includes the Rejection of all other packets received and generated a the same mistake as Grammatical errors ETCS, unless the OBU is in Level 2.

Only the packets 44, 65, 66 and 136 may be located several times in a same eurobalise message
For the 136 package:
• a single packet 136 per Balise telegram,
• the different packages 136 in a same message are identical.

The read packets are filtered as a function:

- from the direction of the transition to the balise group (the packet is the variable QDIR oriented), the bi-default state (level, level of announced and ERTMS / ETCS mode).

The tables of filtering per level and ERTMS / ETCS mode are in the description of the Function "received EUR radio messages" contain.

These packets are transmitted in four different groups:

- Geographical information (packet 79 for the "train locations")

- Information CNX / DCNX (packets 42, 131) for the "the CNX and DCNX Euro Radio Management "

- Information Balise (package 136: Referenzbalise for in-fill information) for the function "Manage eurobalises"

- Track-mail received (other packages) for the "monitor train".

Any error of Referenzbalise, the version or the grammar ETCS is in CR (radio channel) Receiving reported.

6.2.2 F22: Receive Euroradio Messages

SRS § 3.4, § 3.6, § 3.16, § 4.8

Inputs:
“will be complete”

Outputs:
“will be complete”

Description: This function module provides the summary of the Euro radio information and the Verify the consistency of the contents of the Euro radio message. The first guaranteed by this module check is to monitor the radio link. This only applies to a normal communication channel (see § 8.3.1 interface gSM-R).

The review is broken down as follows:

- Basic principle: the RBC is providing its messages with a time stamp based on the Time marking the bi-standard the equipment (if the timestamp of the messages on position is undefined, the message is accepted, but only during the initialization of the Communication session EVC - RBC).

Verification of the sequence:

- the ETCS OBU train equipment rejects any message that is "older" than the last received message.

- the ETCS OBU train equipment is up to any message that is "younger" than the last received message.

6.2.3 F23: Manage Eurobalise Messages

SRS § 3.4, § 3.6, § 3.16

Inputs:
“will be complete”

Outputs:
“will be complete”

Description: This function module manages the Balise reference document of the Bi-Standard-train equipment, is said to know that the information supplied by the track following terms have:

- either the balise which provides the information,

- or delivered in the radio message Referenzbalise,

- or delivered in the infill balise message Referenzbalise (package 136).

A balise group contains 1-8 eurobalises. The balise group is referenced by an identification NIDLRBG (NIDC: identification + NIDBG region: identification beacon). Each eurobalise has an internal number from 1 to 8, which describes the position of the beacon relaltive in the group.

A consisting of only one beacon balise group called "simple beacon", as any other balise group managed; However, it produces features that described in Function module F23 and in the function module F25.

6.2.4 F24: Manage Cnx and Dncx Euroradio

SRS § 3.5, § 3.15.1, § 5.15

Inputs:
“will be complete”

Outputs:
“will be complete”

Description: This function module ensures the production ending a Euroradio- Communication. For the ETCS OBU equipment it is possible to Euro Radio communication session initiate:

- for a "start of mission": train data

- after one obtained from the track command: info CNX / DCNX. The command, an RBC to contact, the identity of the RBC and its telephone number (Packets 42 or 131).

6.2.5 F25: Send Euroradio Messages

§ 3.4, § 3.16

Inputs:
“will be complete”

Outputs:
“will be complete”

Description:

This function module ensures the completeness (location and time stamp) and the Transmission of radio messages euro on the basis of information "radio session message", "Outgoing track message", "message acknowledgment" (for a radio message 136) and "CR Radio channel reception "(for a radio message 136 along with a package 4).

A radio message is sent only if between the Bi-Standard-board equipment and the RBC opened a euro radio session.

The radio messages, with the exception of the message confirmation (radio message 146) and the Session messages (radio message 154, 155, 156, 159) contain a "position report" package (O or 1 packet).

The structure of this "position report" package based on the following information:

- Info for locating the data LRBG, position, velocity, position error, moving direction, Bi-default state for the data concerning the level and the ERTMS / ETCS mode.

The package 1 "special position report" is used for one or two LRBG whose Crossing direction is not known ((LRBG type simple balise group).

The package 0 "position report" is used when the direction of the LRBG is known (group not simple beacons, or group simple beacons, whose Balisenrichtung over by the track was positioned a link or package 135) (see function F23)).

See the functional breakdown of the function in F2 in the next paragraph (Figure 3)

6.2.6 F2 breakdown

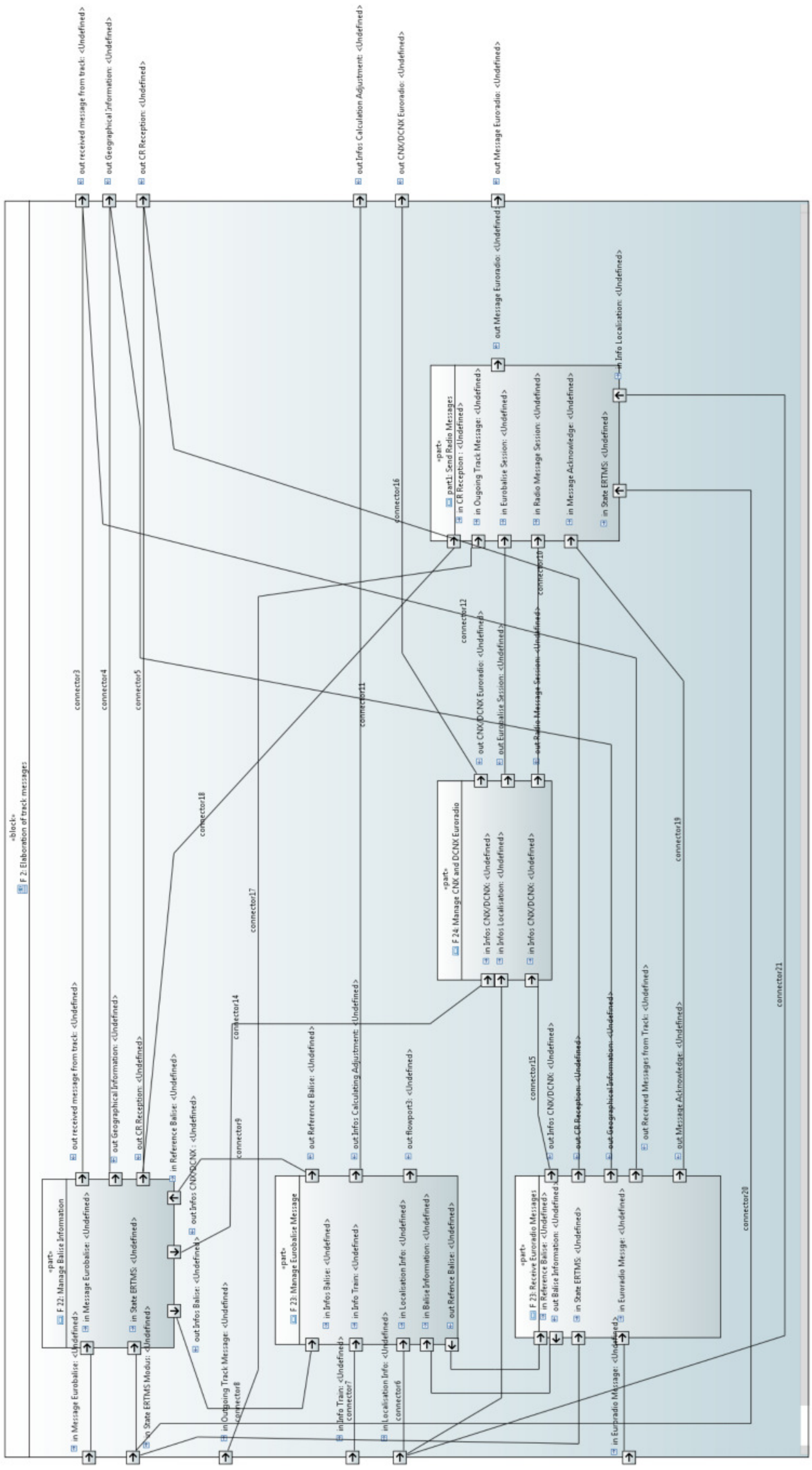


Figure 3. Elaboration track messages breakdown

6.3 F3 Train loction

See Figure 2 - Block F 3

SRS § 3.6.4, § 3.6.5, 3.6.6

Inputs:
“will be complete”

Outputs:
“will be complete”

Description: This function module provides the location of the train in a kernel-reference document (Locating information), based on the following reference documents:

- path measurement calculation reference documents the function "distance measurement": train location,

- Balise the "track messages draw": Information computational comparison.

6.4 F 4 Train Supervising in ERTMS Modus

F4: Train supervision in ERTMS Modus
F41: Manage Level ERTMS/ETCS SRS § 5.1, § 3.6.5
F42: Manage Modus ERTMS/ETCS § 4, § 3.6.5, § 3.15.4, § 5.5, § 5.6, § 5.7, § 5.9, § 5.11, § 5.13
F43: Train Speed supervision SRS § 3.7, § 3.8, § 3.10, § 3.11, § 3.12, § 3.13, § 5.7, § 5.8, § 5.9
F44: Train Movement supervision § 3.14
F45: Train position supervision § 3.6.5, 4.4.8, § 4.4.11
F46: Data storage (§ 4.3), § 3.18
F47: Dialog with the driver § 5.4, § 3.12.3
F48: Manage brake controll § 3.14.1
F49: Manage Train Controll, § 3.12.1
See Figure 4 - Block 4

6.5 F4 breakdown

7 F4 Functional Breakdown

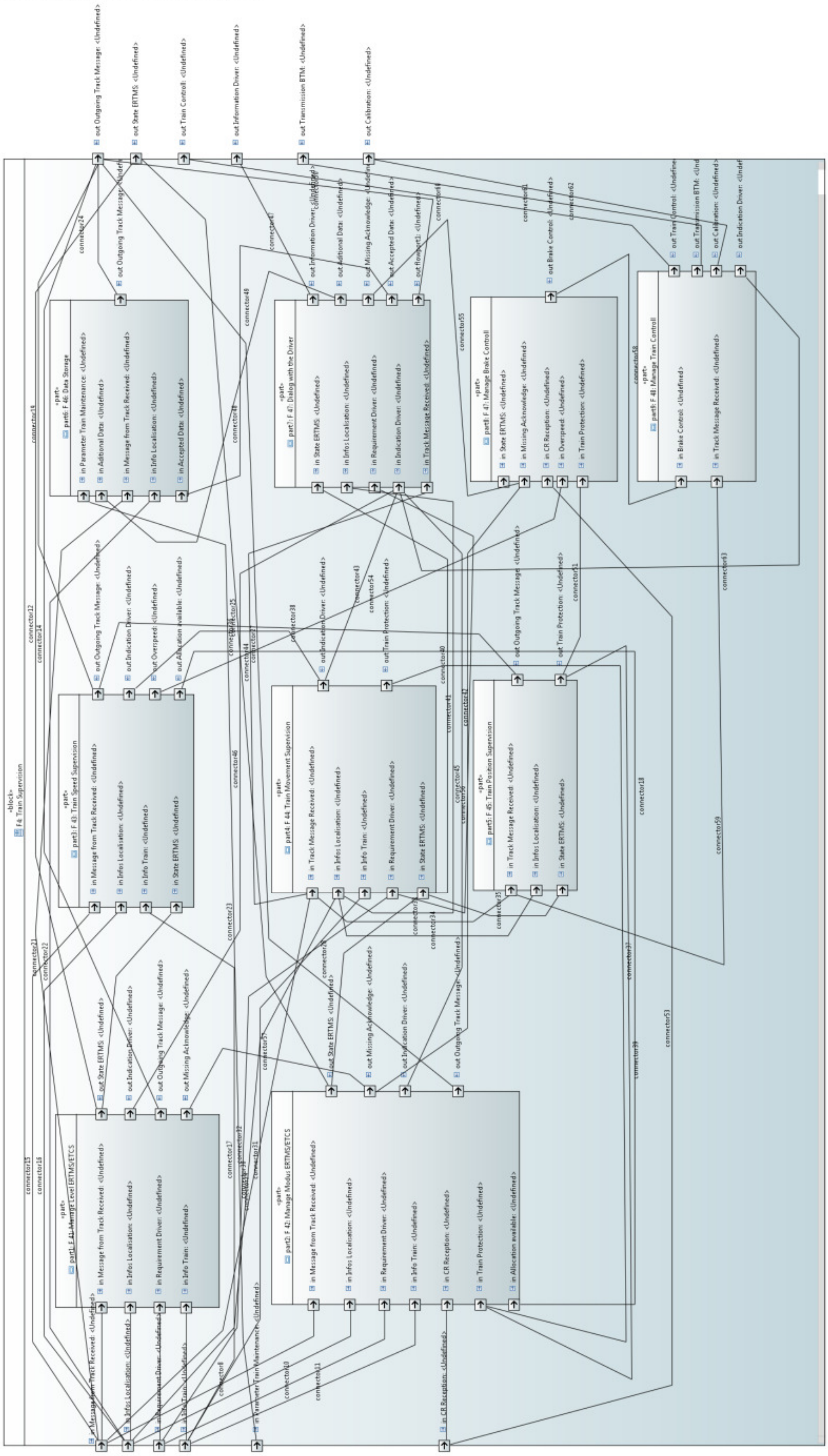


Figure 4. Train Supervising in ERTMS Modus

7.1 F5 Exchange outpout

F5: Exchange output
See Figure 2 - Block F 5

7.2 Data Flow Description

Number	Flow	Source/Sink	Description
1.0	Missing Acknowledge	Kernel xx/Kernel xx	This information indicates the absence of the Acknowledgment of the driver to a confirmatory text message on the Level Transition, the transition mode or a track Train-text message
2.0	Acknowledge	Kernel xx/Kernel xx	Radio message 146 transmission Confirmation - Confirmation message depending on the variable § 8.6.7
3.0	Action Driver	Kernel xx/Kernel xx	Election / confirmation / detection Driver via DMI
4.0	Display Driver	Kernel xx/Kernel xx	For the driver specific DMI display
5.0	Reference Balise	Kernel xx/Kernel xx	information; in the Flow "information Balise" delivered Balise confirmed as a reference beacon.
6.0	Allocation available	Kernel xx/Kernel xx	Information message that an allocation for a given technical mode is available.
7.0	Calibration	Kernel xx/Kernel xx	Specifying a calibration range of the path measurement.
8.0	Cnx/Dcnx Euroradio	Kernel xx/Kernel xx	Command on the connection and disconnection of a or of a RBC, the identity and Phone RBC contains.

9.0	Brake Controll	Kernel xx/Kernel xx	Control of the service and the emergency braking.
10.0	Train Controll (brake)	Kernel xx/Kernel xx	Functional output.
11.0	CR Reception (Radio Channel)	Kernel xx/Kernel xx	CR (radio channel) on receipt of a balise or Radio message: Checking version ETCS, Grammar ETCS, testing reference balise (Position), monitoring radio link
12.0	Accepted Data	Kernel xx/Kernel xx	Information valid confirmation of the train data and Consequently, at the very beginning of the mission.
13.0	Additional Data	Kernel xx/Kernel xx	Data relating to the mission of the train: Identity of the driver, selected level, identity and mandatory. of the RBC, if Level 2, no. Mission
14.0	JRU Data	Kernel xx/Kernel xx	Recorded legal data.
15.0	Input Train	Kernel xx/Kernel xx	From the train coming TOR digital inputs and Digital outputs
16.0	State ERTMS	Kernel xx/Kernel	Current Level and operating mode ERTMS, announced ERTMS Level and Current Level and operating mode ERTMS, announced ERTMS Level
17.0	Indication Driver	Kernel xx/Kernel xx	each specific for the driver Specification. This includes the dynamic information speed monitoring the different messages and symbols, the Exchange during the "start of mission"

18.0	Info Train	Kernel xx/Kernel xx	From Train comming functional information TIU
19.0	Message Eurobalise	Kernel xx/Kernel xx	Information, which the ID Balise group (country ID + ID Balise), their reading direction and their Contains path measurement for review (see Flow "reference balise")
20.0	Information cnx/dcnx	Kernel xx/Kernel xx	Information identifying the RBC ID (Identification of land + the identification of the RBC), RBC his phone number and the authorization
21.0	Information Driver	Kernel xx/Kernel xx	Specific for the driver Function DMI information. They contain the different specifications for the Driver, which are generated from the different Function modules.
22.0	Geographical information	Kernel xx/Kernel xx	Absolute current position of the train, which the add the flow "location information"
23.0	Information Localisation	Kernel xx/Kernel xx	Locating the train: position and speed of the train in the different reference documents, including the absolute geographical Position.
24.0	Message Eurobalise	Kernel xx/Kernel xx	Flow, summarizing the message / messages eurobalise. BTM to F1: the entire telegrams F1 to F2: Message (the safety layer and the the safety layer and the transfer of Telegrams in a message by F1 F1 guaranteed)

25.0	Infos Calculation Adjustment	Kernel xx/Kernel xx	Information, which contains the ID Balise group (country ID + ID Balise), their path measurement and her for the computational comparison of the on-board Contains reference document expected position
26.0	Localisation Train (ODO)	Kernel xx/Kernel xx	path measurement in comparison to a Balise (+ safety margin), Speed (+ Safety distance), direction of travel and capture of stopping
27.0	Euroradio Message (train to track)	Kernel xx/Kernel xx	Transmitted Euro radio message (the safety layer is guaranteed by F5)
28.0	Euroradio Message (track to train)	Kernel xx/Kernel xx	Received Euro radio message (the safety layer is guaranteed by F1)
29.0	Radio Message Session	Kernel xx/Kernel xx	Message / parcel Euro Radio for creating and Termination of the radio connection
30.0	Outgoing Track Message	Kernel xx/Kernel xx	Before Inform creation and timestamp (except the message flow Radio link, message confirmation and CR Reception) sent to the track to function message

31.0	Message from Track Received	Kernel xx/Kernel xx	Via radio or beacon message received function (except the Riverside current Information, cnx / dcnx information, Balise information)
32.0	Parameter Train maintenance	Kernel xx/Kernel xx	Fixed data, train data, domestic values
33.0	Train Protection	Kernel xx/Kernel xx	Information, protection against rolling Twitching, monitoring of stopping, the Transition to an unauthorized Balise in SH or SR reports
34.0	Balise Reception	Kernel xx/Kernel xx	Information of receipt of a reference Balise NPIG = 0 from the associated type for the computational comparison of the displacement measurement
35.0	Requirement Driver	Kernel xx/Kernel xx	By driver outbound (see flow "Action Driver") or from DMI DMI produced function information
36.0	Euroradio Session	Kernel xx/Kernel xx	State of the Euroradio connection
37.0	Signal Eurobalise	Kernel xx/Kernel xx	Airgap Eurbalise -> BTM
38.0	Signal Euroradio	Kernel xx/Kernel xx	Airgab RBC ->Euroradio Board
39.0	Output Train	Kernel xx/Kernel xx	Train digital outuputs
40.0	Overspeed	Kernel xx/Kernel xx	Information message overspeed of the train relative to the speed curves
41.0	Synchronization	Kernel xx/Kernel xx	Synchronization (xw, V, T) between the kernel, the BTM and the path measurement

42.0	Transmission BTM	Kernel xx/Kernel xx	Control BTM - antenna

7.3 current partly openETCS Architecture

Needs to be integrated into the overall architecture .....

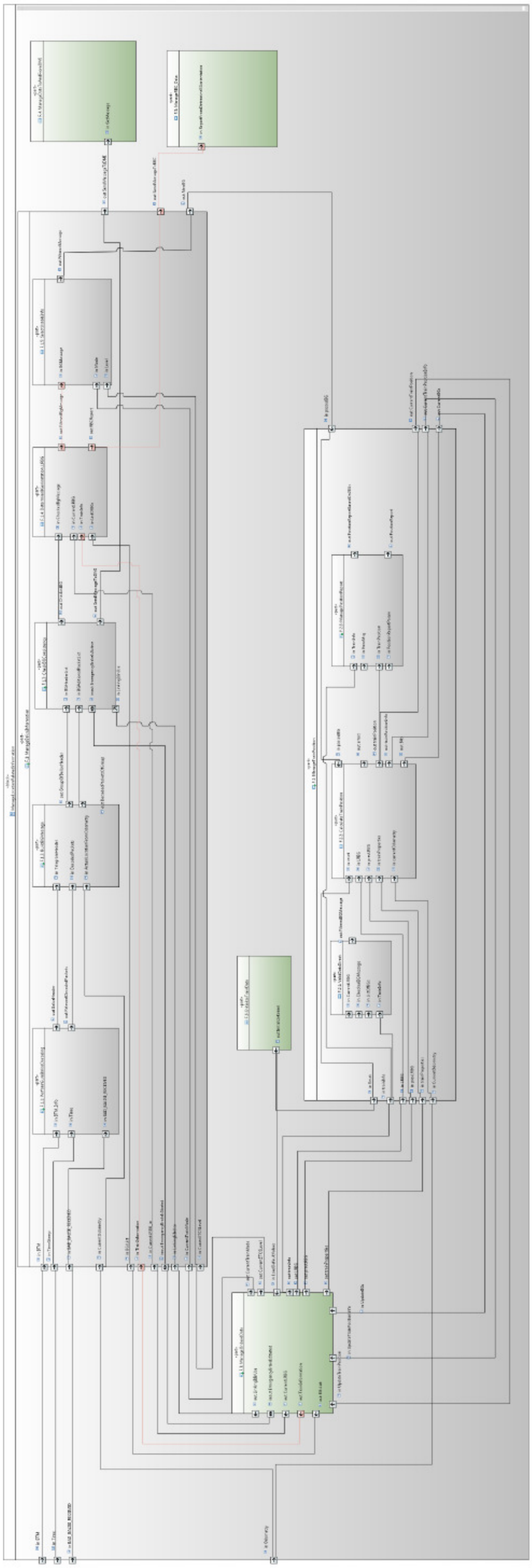


Figure 5. Centralized data structure approach architecture

7.4 centralized data structure approach

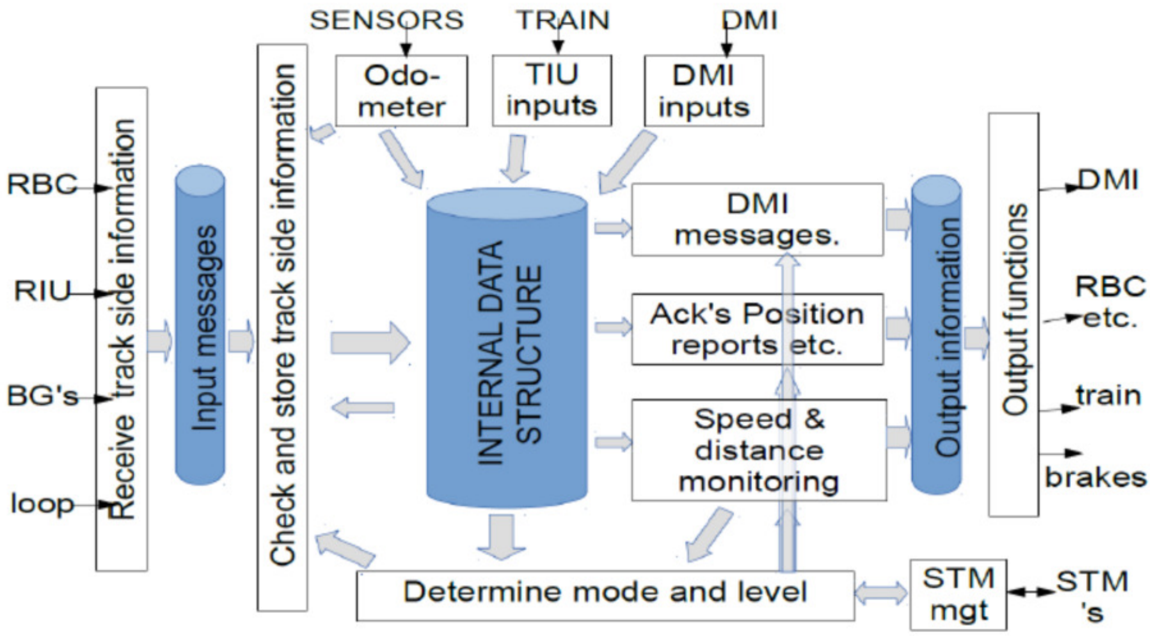


Figure 6. Centralized data structure approach architecture

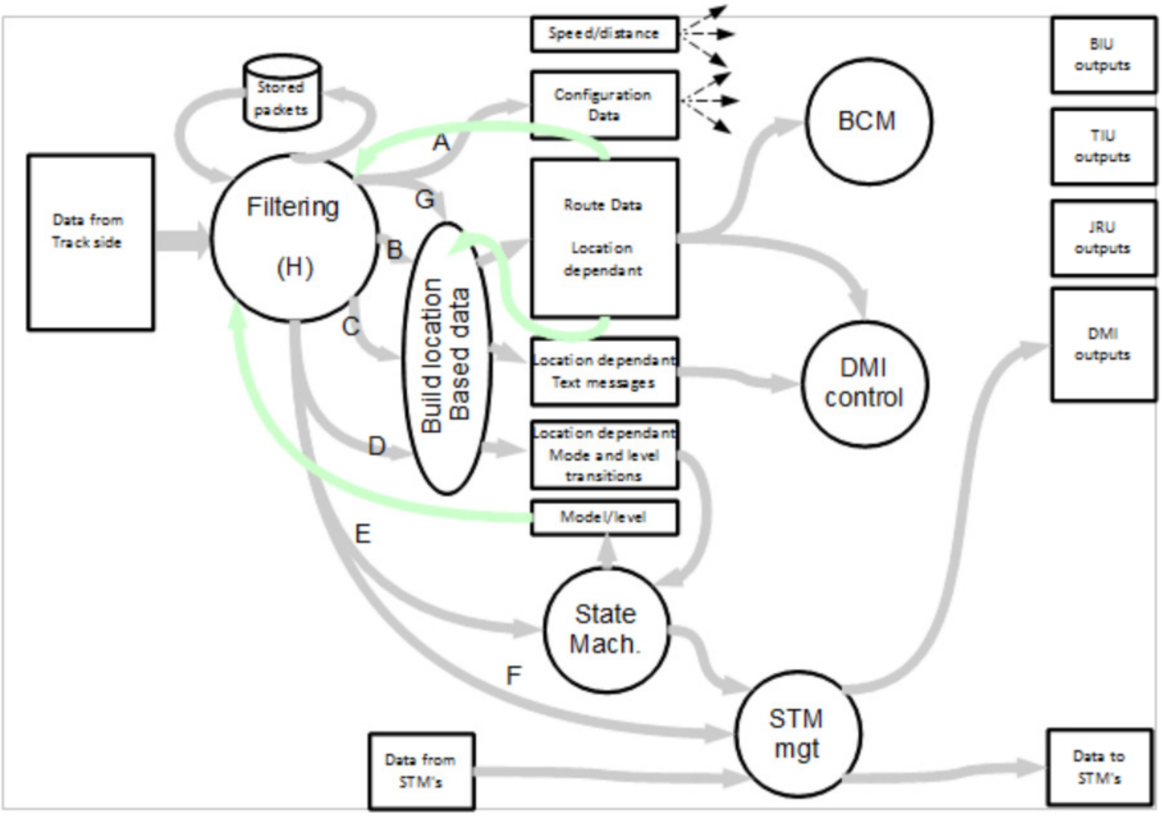


Figure 7. Centralized data structure approach breakdown

7.5 Alstom High Level Approach

8 Alstom High Level SRS Architecture

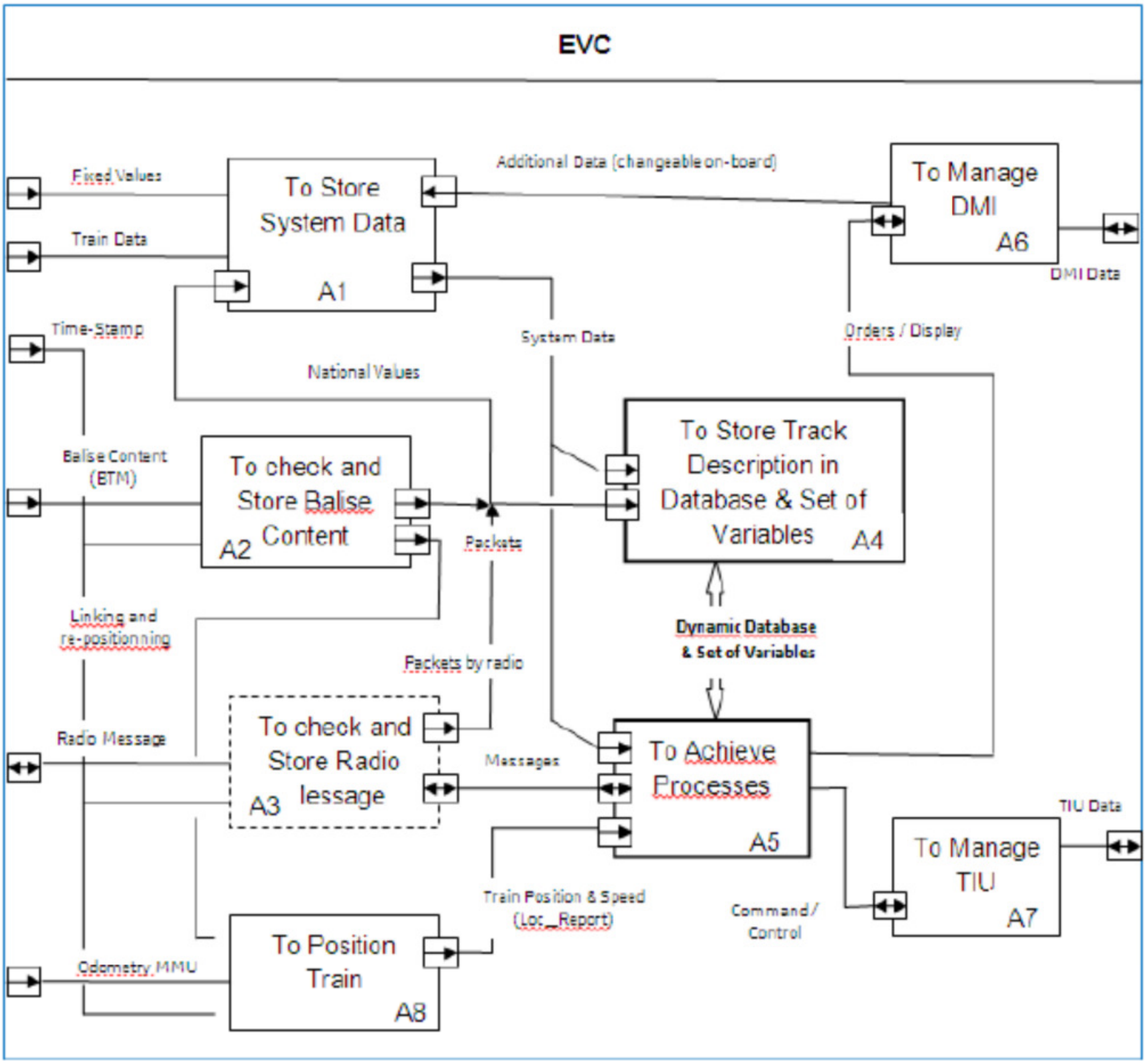


Figure 8. Alstom SRS Architecture Approach

8.1 The openETCS Tool-Chain and its impacts on the actual model

For understanding the modelling process and the modeling guidelines, we refer to `https://github.com/openETCS/modeling/blob/master/DescriptionOfWork/NewModelingDescriptionOfWork.pdf`.

To summarize the design process, the following rules are in use:

- Papyrus / SysML is used for modelling the architecture. Functions are visible on this SysML level.
- No behaviour model is allowed on SysML level.
- For referencing the requirements, links from the SysML model to the requirements document (in ProR) are being used.
- Details and especially behaviour is part of the Scade models.
- All interfaces (see also data-dictionary below) are available on bit-level.
- In the architecture model in SysML, all interfaces are available on a functional level for interfaces inside and outside the model and for interfaces between dedicated functions. Due to tool constrains the current model does not show all details for all interfaces (see dataDictionary).

The openETCS tool-chain for doing the modelling work consists of the following components:

Papyrus : for modelling the architecture (Kepler version).
In this phase only the Kepler version of the tool can be used due to incompatibilities of the Kepler and the Luna version on the SysML model. The SysML models are stored in the following location: `https://github.com/openETCS/modeling/tree/master/model/sysml`.

ProR : for keeping the requirements (REQIF).
The subset 26 is converted into a REQIF-format and also stored in the modeling repository on Github. The openETCS toolchain supports the linking of SysML model parts to SRS-Requirements. These results are also part of the architecture.

Scade : for designing and formalising the functions Scade version 15.2 is used.
The models are stored in this location: `https://github.com/openETCS/modeling/tree/master/model/Scade`. With the component Scade System Scade also has a component for designing the architecture.

In principle, the synchronisation mechanism of Scade was planned to be used for synchronising the SysML architecture and the Scade models. The idea is to automatically synchronise the SysML types and blocks with the Scade type definitions and the Scade Operators. Unfortunately, with the current set of tools this idea cannot be realised. We will investigate other tools and models to find a solution.

In addition, faults in the Kepler Papyrus version made it difficult for several members of the team to work on different submodels of the openETCS model. The issue will be solved when changing to the Luna version of Papyrus.

9 Functions of the openETCS Model

9.1 openETCS Data Dictionary

/subsubsectiondataDictionary

9.2 openETCS Generic API

9.2.1 Generic API

Because the API is currently not defined to that level, the following assumptions are implemented:

- **Eurobalise (BTM)**: One telegram per call ...

9.3 openETCS Balise Group

9.3.1 Receive Eurobalise From API

- **Short Description of Functionality**
  This function defines the interface of the OBU model to the openETCS generic API for Eurobalise Messages. On the interface, either a valid telegram is provided or a telegram is indicated which could not be received correct when passing the balise. The function passes the telegram without major changes of the information to the next entity for collecting the balise group information.
- **Design Constrains and Choices**

  1. Decoding of balises is done at the API. Also, packets received via the interface are already transformed into a usable shape.
  2. Only packets used inside the current model are passed via the interface:
     Packet 5: Linking Information.
     Linking Information is filled into the linking array starting from index 0 without gaps. Used elements are marked as valid. Elements are sorted according to the order given by the telegram sequence.

9.3.2 Build BG Group

- **Short Description of Functionality**
  This entity collects telegrams received via the interface into Balise Group Information.
- **Reference to the SRS (or other requirements**
- **Design Constrains and Choices**

  1. Telegrams received as invalid are passed to the “Check-Function” in order to process errors in communication with the trackside according to the requirements and in a single

place. Telegrams are filled into the telegram array starting from index 0 without gaps. Used elements are marked as valid. Elements are stored according to the order given by the telegram sequence.

2. Only packets used inside the current model are passed via the interface:
   Packet 5: Linking Information.

3. In this function packets past with the telegrams is accumulated into a balise group information.

4. Further assumption on packet 5 (based on SRS subset 26, section 8.4.1.4)
   (In this statement the term "message" is ambiguous since it can reflect to a telegram or a balise group message) Linking Information can only be passed once. This means, if linking information for the balise group is already collected with one of the earlier telegrams, the information will not be accumulated but overwritten.

9.3.3 Check BG Consistency

9.3.4 Determine BG- Orientation and LRBG

- **Short Description of Functionality**

- **Reference to the SRS (or other requirements**

- **Design Constrains and Choices**

9.3.5 Select Usable Info

9.3.6 Perform Balise Decoding

9.4 openETCS Train Position

9.4.1 Calculate Train Position

- **Short Description of Functionality**
  The main purpose of the function is to calculate the locations of linked and unlinked balise groups (BGs) and the current train position while the train is running along the track.

  1. The input data received from the balises groups must have been checked and filtered for validity, consistency and the appropriate train orientation before delivering them to calculateTrainPosition.
  2. The storage capacity for balise groups is finite. calculateTrainPosition will raise an error flag when a balise group cannot be stored due to capacity limitations.
  3. calculateTrainPosition will raise an error flag if a just passed balise group is not found where announced by linking information. It will not (yet) detect when an announced balise group is missing.

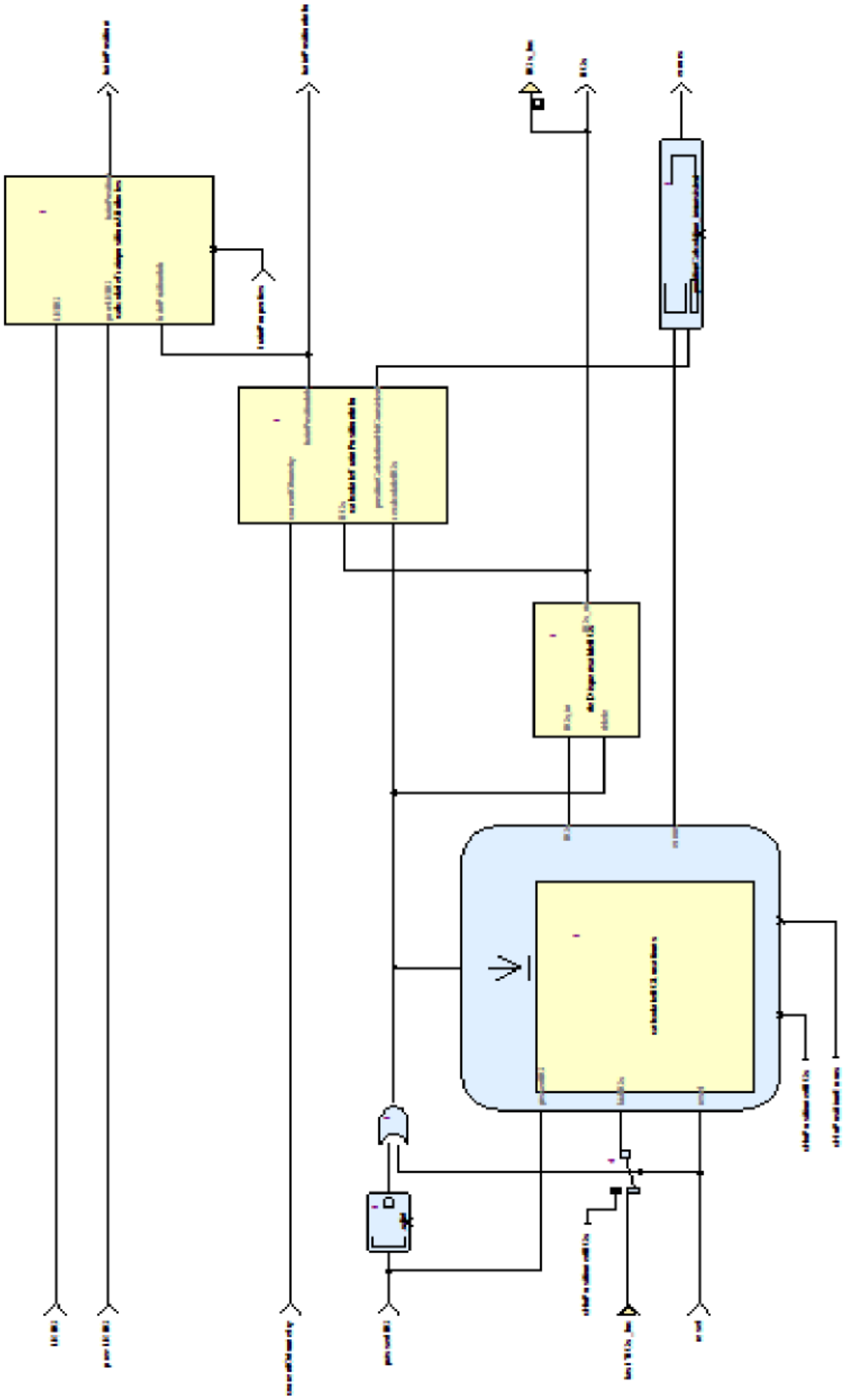


Figure 9. Structure of calculateTrainPosition

4. calculateTrainPosition is not yet prepared for train movement direction changes.
5. calculateTrainPosition does not yet consider repositioning information.

9.4.2 Provide Position Report

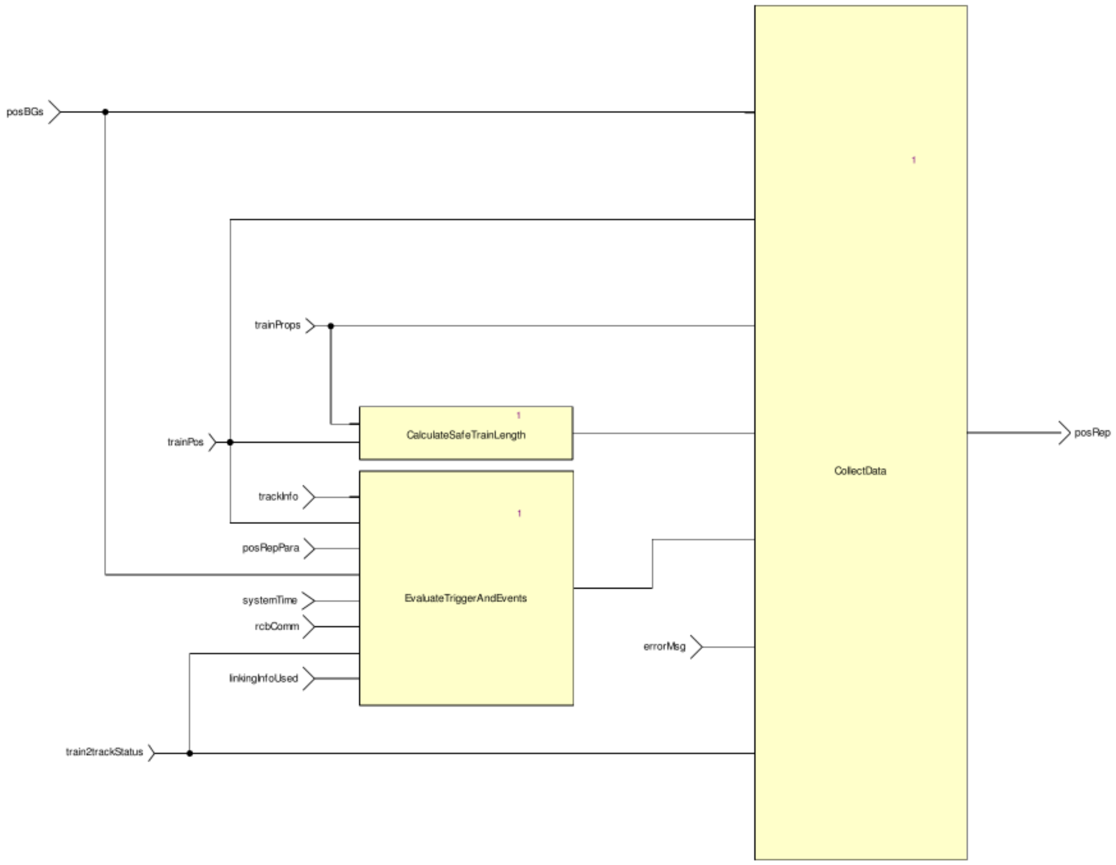


Figure 10. Structure of component ProvidePositionReport

- **Short Description of Functionality**
  This function takes the current train position and generates a position report which is sent to the RBC. The point in time when such a report is sent is determined from event, on the one hand, and position report parameters—which are basically triggers—provided by the RBC or a balise group passed, on the other hand. The functionality is modeled using three operations, as shown in Fig. 10, which are explained below.

  CalculateSafeTrainLength Calculates the the safeTrainLength according to Chapt. 3.6.5.2.4/5.
  `safeTrainLength = absolute(EstimatedFrontEndPosition - MinSafeRearEnd)`,
  where `MinSafeRearEnd = minSafeFrontEndPosition - L_TRAIN`

  EvaluateTriggerAndEvents Returns a Boolean modeling whether the sending of the next position report is triggered or not. It is the conjunction of the evaluation of all triggers (PositionReportParameters, i.e., Packet 58) and events (see Chapt. 3.6.5.1.4).

  CollectData In this operation, data of Packet0, . . . , Packet5 and the header is aggregated to a position report.

- **Reference to the SRS (or other requirements**
  Most of the functionality is described in subset 26, chapter 3.6.5.

- **Design Constrains and Choices**

  1. The message length (i.e., attribute `L_MESSAGE`) is by default set to 0; the actual value will be set by the Bitwalker/API.
  2. The attribute `Q_SCALE` is assumed to be constant; that is, all operations using this attribute do not convert between different values of that attribute.
  3. *PositionReportHeader*: The time stamp (i.e., attribute `T_TRAIN`) is not set; this should be done once the message is being sent by the API
  4. *Packet4*: When aggregating the data for this packet, an error message might be overwritten by a succeeding error message. Because the specification only allows to sent one error in one position report, errors are not being stored in a queue, for instance.
  5. *Packet44*: This packet is currently not contained in a position report as it is not part of the kernel functions.
  6. The usage of attributes `D_CYCLOC` and `T_CYCLOC` as part of the triggers specified by the position report parameters (i.e., Packet 58 sent by the RBC) may lead to unexpected results if a big clock cycle together with small values for the attributes is used. The cause is that the current model increments at every clock cycle the reference value for the distance and time by at most `D_CYCLOC` and `T_CYCLOC`, respectively and not a factor of it.

- **Open Issues**

  1. Operation *EvaluateTriggerAndEvents* currently ignores parameters `N_ITER`, `D_LOC` and `D_LGTLOC` which allow to specify up to 32 position at which a report has to be sent. The positions are relative to the location of a reference balise group. If the RBC sends packet 58, then it also provides a reference balise group; otherwise, if packet 58 is sent by a balise group, then this balise group serves a the reference balise group. Possible realisation in the model: Extend in the interface posRepPara (i.e., Packet 58) by a `NID_BG` referring to the reference balise group. Am assumption would be that this BG can be found in the list of passed balise group provided by *CalculateTrainPosition* in Sect. 9.4.1.
  2. The specification requires to store the last eight balise groups for which a position report has been sent (see 3.6.2.2.2.c).
  3. For all reports that contain Packet 1 (i.e., report based on two balise groups), the RBC sends a coordinate system. It is unclear where this has to be stored (i.e., somehow the balise groups have to be stored in a database which has then to be updated), see 3.4.2.3.3.6. Moreover, such a coordination system can be invalid and then has to be rejected (see 3.4.2.3.3.7-8). On a more abstract level, we need to think about the interface between the RBC and the OBU or a proper abstraction thereof.
  4. The decision whether a the report consists of packet 0 or packet 1, which is provided in 3.4.2.3.3, is currently not completely modeled. So far, 3.4.2.3.3.1 has only been modeled, thereby assuming “the last balise group detected” is the last balise group and not the LRBG. 3.4.2.3.3.2 is unclear. To model 3.4.2.3.3.4 I need information about the last two valid balise groups and the train running direction. This information can be obtained by adding a memory or this information will be provided by *CalculateTrainPosition* in Sect. 9.4.1. Likewise, also 3.4.2.3.3.5 requires knowledge about the last two valid balise groups.

References

[1] US Army Corps of Engineers, Engineer Research and Development Center. *Guide for Preparing Technical Information Reports of the Engineer Research and Development Center*, January 2006.

[2] Walter Schmidt. *Using Common PostScript Fonts With LaTeX. PSNFSS Version 9.2*, September 2004. `http://ctan.tug.org/tex-archive/macros/latex/required/psnfss`.

[3] Hideo Umeki. *The geometry Package*, December 2008. `http://ctan.tug.org/tex-archive/macros/latex/contrib/geometry`.

[4] Piet van Oostrum. *Page Layout in LaTeX*, March 2004. `http://ctan.tug.org/tex-archive/macros/latex/contrib/fancyhdr`.

[5] D. P. Carlisle. *Packages in the 'Graphics' Bundle*, November 2005. `http://ctan.tug.org/tex-archive/macros/latex/required/graphics`.

[6] UK TeX Users Group. UK list of TeX frequently asked questions. `http://www.tex.ac.uk/cgi-bin/texfaq2html`, 2006.

[7] Leslie Lamport. *LaTeX: a Document Preparation System*. Addison-Wesley Publishing Company, Reading, Ma., 2 edition, 1994. Illustrations by Duane Bibby.

[8] Department of Defense, Washington, DC. *Distribution Statements on Technical Documents. DoD Directive 5230.24*, 1987.

[9] Axel Sommerfeldt. *Typesetting Captions with the caption Package*, February 2007. `http://ctan.tug.org/tex-archive/macros/latex/contrib/caption`.

[10] Patrick W. Daly. *Natural Sciences Citations and References (Author-Year and Numerical Schemes)*, February 2009. `http://ctan.tug.org/tex-archive/macros/latex/contrib/natbib`.

[11] Michael Downes and Barbara Beeton. *The amsart, amsproc, and amsbook document classes*. American Mathematical Society, August 2004. `http://www.ctan.org/tex-archive/macros/latex/required/amslatex/classes`.

[12] David Carlisle. *The longtable Package*, February 2004. `http://www.ctan.org/tex-archive/macros/latex/required/tools`.

[13] Martin Schröder. *The ragged2e Package*, March 2003. `http://ctan.tug.org/tex-archive/macros/latex/contrib/ms`.

[14] Leslie Lamport, Frank Mittelbach, and Johannes Braams. *Standard Document Classes for LaTeX version 2e*, 1997. `http://ctan.tug.org/tex-archive/macros/latex/base`.

[15] David Carlisle. *The dcolumn Package*, May 2001. `http://ctan.tug.org/tex-archive/macros/required/tools`.

[16] American Mathematical Society. *User's Guide for the amsmath Package (Version 2.0)*, February 2002. `http://ctan.tug.org/tex-archive/macros/latex/required/amslatex/math/amsldoc.pdf`.

[17] Boris Veytsman. *LaTeX Support for Microsoft Georgia and ITC Franklin Gothic In Text and Math*, July 2009. `http://ctan.tug.org/tex-archive/fonts/mathgifg/`.

References

[1] Leslie Lamport, *LaTeX: A Document Preparation System*. Addison Wesley, Massachusetts, 2nd Edition, 1994.