OETCS/WP5/M5.X

openETCS

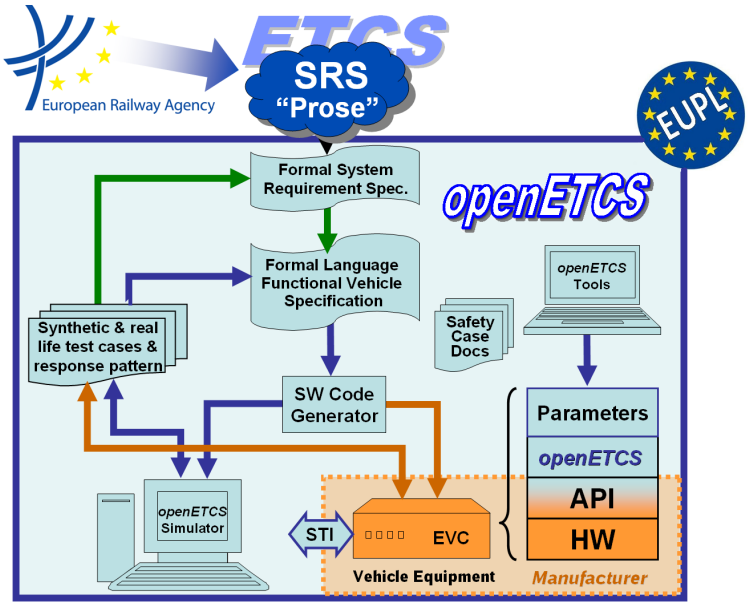
ITEA2 Project
Call 6 11025
2012 – 2015

Work-Package 5: “Demonstrator”

ERSA Demonstrator 1st release API

Nicolas Van Landeghem, Didier Weckmann, Alexis Julin,
Stéphane Chenevoy

August 2014



Funded by:

Federal Ministry of Education and Research
Région de Bruxelles-Capitale
GOBIERNO DE ESPAÑA
MINISTERIO DE INDUSTRIA, ENERGÍA Y TURISMO

This page is intentionally left blank

Work-Package 5: “Demonstrator”

OETCS/WP5/M5.X
August 2014

ERSA Demonstrator 1st release API

Document approbation

Lead author:	Technical assessor:	Quality assessor:	Project lead:
location / date	location / date	location / date	location / date
signature Nicolas Van Landeghem (ERSA)	signature Nicolas Van Landeghem (ERSA)	signature Ainhoa Gracia (SQS)	signature Klaus-Rüdiger Hase (DB Netz)

Nicolas Van Landeghem, Didier Weckmann, Alexis Julin, Stéphane Chenevoy

ERSA
5 Rue Maurice Blin
67500 Haguenau, France

Description of work

Prepared for openETCS@ITEA2 Project

Modification History

Version	Section	Modification / Description	Author
1.0	All Parts	Document creation	Nicolas Van Landeghem
1.1	All Parts	Integrate GE comments	Nicolas Van Landeghem

Table of Contents

1 Introduction

The purpose of this document is to describe ERSA's API of its EVC simulator: its breakdown into modules, the exchanged data and the interfaces with external devices. The EVC simulator behaves according to the SRS Class 1 version 3.3.0.

2 API content

The API definition is split between its public types, structures, methods and members. Methods are divided into families, e.g. check consistency and basic configuration, control EVC simulation, set EVC configuration data, store data, handle train data, handle internal events and retrieve internal data.
While ERSA's API has private types and its interface to the EVC kernel is predefined, openETCS' API (currently under definition) may have similar interfaces.

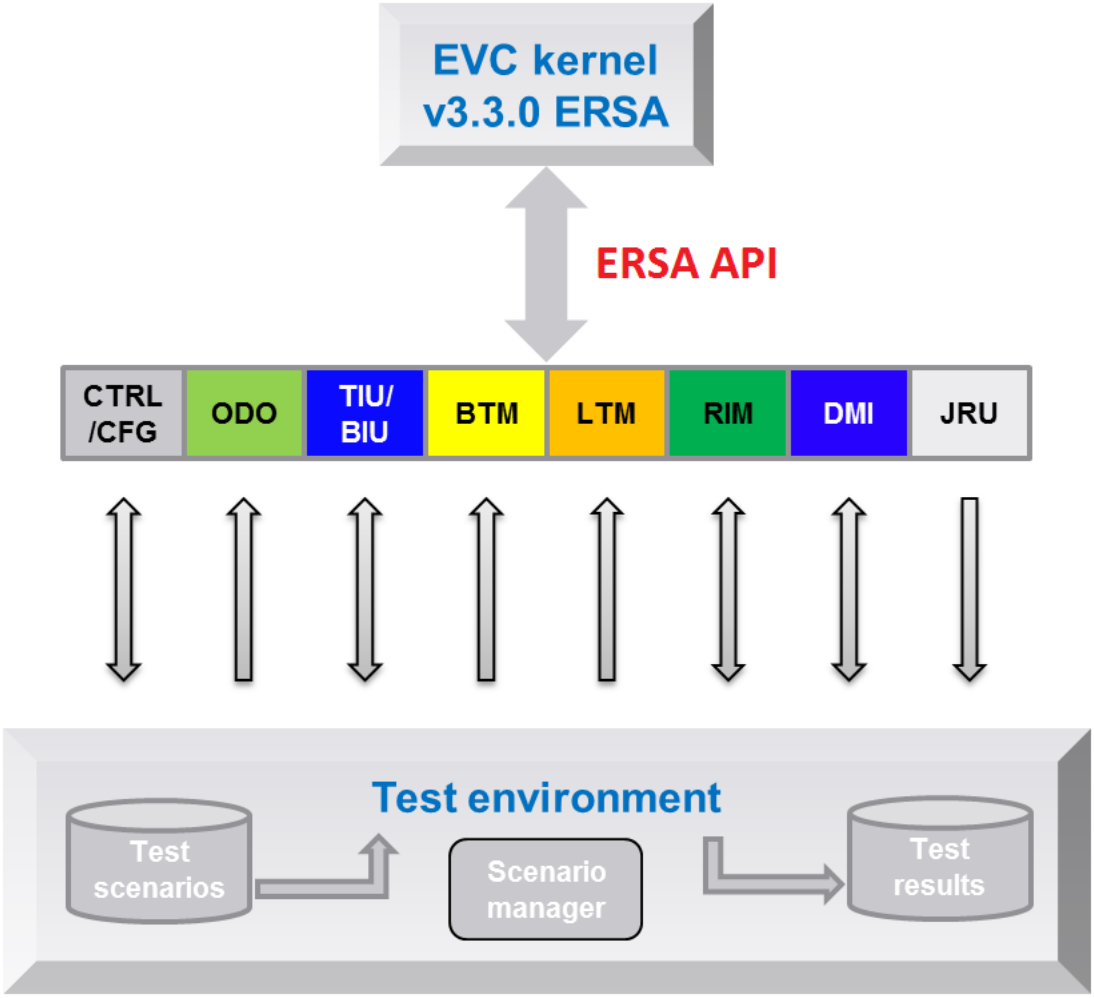


Figure 1. ERSA API to ERSA EVC

ERSA's API has interfaces between its proprietary simulated peripherals and its simulated EVC kernel. Thus, data exchanged, their format, as well as communication between the peripherals and the kernel itself are only compliant with ERSA EVC.

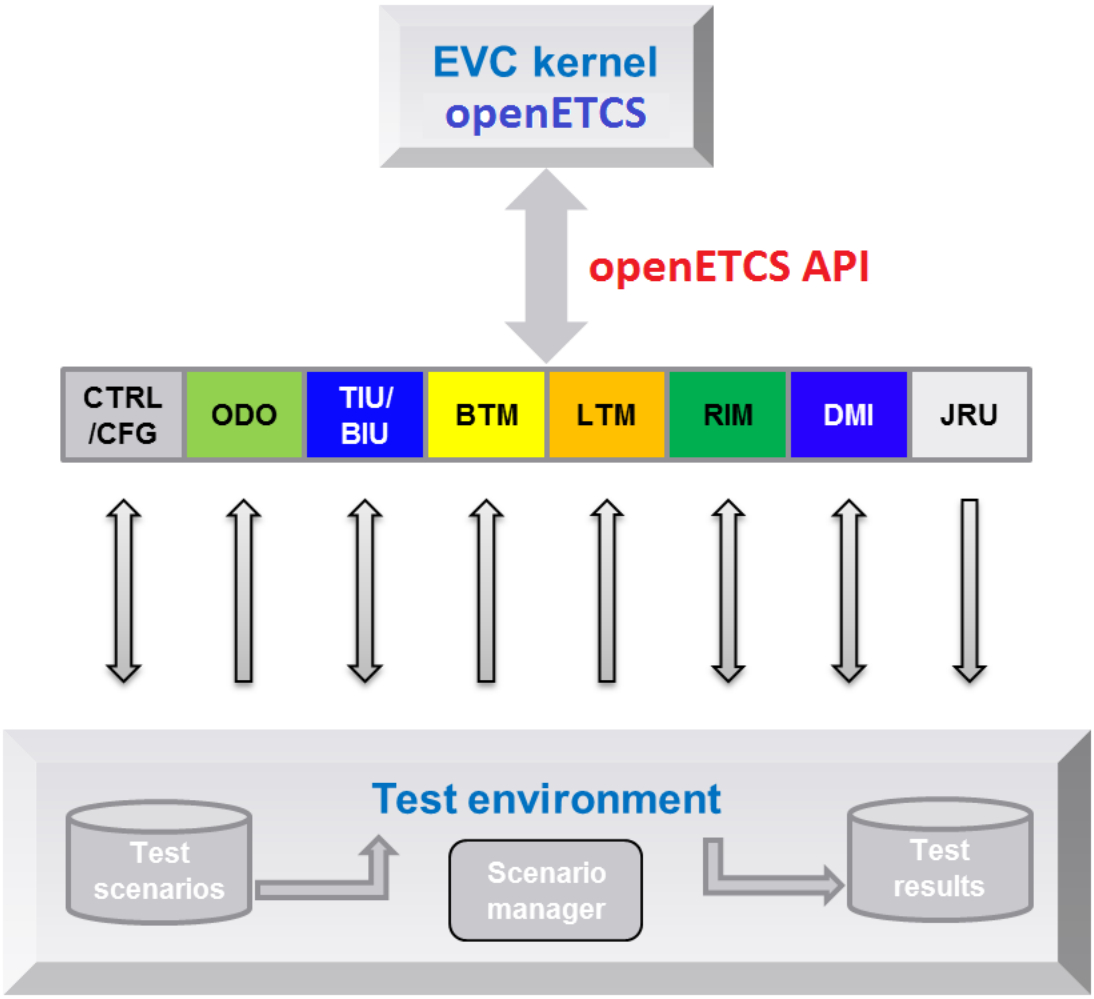


Figure 2. openETCS API to openETCS EVC

openETCS's API interface between the peripherals and the EVC kernel is currently under definition. OpenETCS API may differ from ERSA interface, nevertheless the closer they are, the less adaptation will be required to replace a kernel by the other one. Indeed, for test purposes, it may be interesting to have a common API definition of these interfaces, allowing to define a reference behaviour that can be later compared to a what is under development.

2.1 Function prototypes

Initialise the EVC simulator: creation of EVC data structure, initialisation of interfaces.

Table 1. Simulation functions

Return type	Function prototype	Comment
int32_t, 0 on success	Init(uint32_t ulLogId)	[in] ulLogId, key used as prefix for log files
int32_t, 0 on success	Start_processes(void)	Start the EVC simulator modules
int32_t, 0 on success	Run(void)	Start the EVC simulation
int32_t, 0 on success	Stop(void)	Stop the EVC simulation

void	Modify_EVC_Configuration (bool bSet, eConfigData Config)	Modify the EVC configuration. It should be call just after constructor [in] bSet, indicate if the config should be set or reset [in] Config, flag to identify the configuration

Table 2. Assessor functions "get"

Return type	Function prototype	Comment
double, train speed in km/h	GetTrainSpeed(void)	Get current train speed (km/h)
double, target speed in km/h	GetTargetSpeed(void)	Get target speed (km/h)
t_distance, target location	GetTargetLocation(void)	Get target location
t_distance, estimated front loc.	GetEstimatedFrontLoc(void)	Get estimated front location
double, EBrake speed in km/h	GetEBrakeSpeed(void)	Get EBrake speed (km/h)
double, SBrake speed in km/h	GetSBrakeSpeed(void)	Get SBrake speed (km/h)
double, permitted speed in km/h	GetPermSpeed(void)	Get permitted speed (km/h)
double, warning speed in km/h	GetWarnSpeed(void)	Get warning speed (km/h)
t_distance, EOA location	GetEoaLocation(void)	Get EOA location
double, EOA speed (km/h)	GetEoaSpeed(void)	Get EOA speed (km/h)
int32_t, adhesion factor	GetAdhesionFactor(void)	Get adhesion factor
double, most restrictive speed	GetMostRestrictiveSpeed(void)	Get most restrictive speed (km/h)
eTrainMode, train mode	GetCurrentMode(void)	Get current on-board ETCS mode
eLevel, train level	GetCurrentLevel(void)	Get current on-board ETCS level
eRadSafeConnectStatus, radio safe connection status	GetSafeConnStatus(int32_t lEquipmtIdx)	Get safe connection status for indicated equipment (0 or 1)
eRadSessionState, radio session status	GetSessionStatus(int32_t lEquipmtIdx)	Get session status for indicated equipment (0 or 1)
eSpeedMonitoringStatus, enumerated speed monitoring status	GetSpeedMonitoringStatus(void)	Get speed monitoring status
eBrakePosition, enumerated brake position	GetBrakePosition(void)	Get brake position
bool, true if message is valid, 0 otherwise	GetIhmMessages(char * szMessage, eMSG_TYPE * type)	Get messages from UI [out] pointer to UI message [out] pointer to type of message (balise, loop, radio, other)
bool, true if message is valid, 0 otherwise	GetDriverAction(t_distance & rdLocation, t_time & rdTimeStamp, eDriverActionInfo & rDriverAction, SVarDataList & rVarDataList);	Get driver action produced on DMI [out] location where action was sent [out] timestamp when action was sent [out] action requested by driver [out] list of data if relevant (driver id, etc.)

bool, true if list is available and has changed (according to counter value)	GetTCProfile(STCInfoList & rTCInfoList, int32_t & rlCounter)	[out] list of track conditions information [in/out] counter used to test changes
bool, true if DMI is ready	IsDMIReady(void)	Get DMI status

Table 3. Assessor functions "set"

Return type	Function prototype	Comment
void	SetBTMAlarm(bool bSwitchOn)	Set or reset BTM alarm [in] set or reset value
void	SetIsolation(bool bIsolated)	Set or reset isolation [in] set or reset value
void	SetStoredLevel(eValidity Validity, SLevel Level)	Set stored level data (starting condition) [in] validity of level data [in] stored ETCS level
void	SetStoredRadioNetwork(const char* szRadNetId)	Set stored radio network id [in] radio network id, up to 6 digits
void	SetStoredRbcData(eValidity Validity, uint32ûlRBCId, const char* szPhoneNb)	Set stored RBC data [in] validity of RBC data [in] RBC identity (value in 24 bits) [in] RBC phone nr, up to 16 digits
void	SetStoredTrainPosition(eValidity PosVal, uint32_t ulNID_LRBG, int32_t lLRBGDistance, eDirection LRBGDir)	Set stored train position data [in] validity of position data [in] identifier of the LRBG [in] distance between train front and LRBG [in] orientation of LRBG
void	SetTrainEquipment(bool bBaliseComAvailable, bool bLoopComAvailable, int32_t lNbRadioSessionAvailable, bool bIntegrityDeviceAvailable, bool bServiceBrakeAvailable, bool bTCOAvailable, bool bBrakeFeedBackAvailable, bool bAirtighAvailable, bool bColdMvtDetectorAvailable, const char* szPhoneNb1, const char* szPhoneNb2)	Set available train equipment [in] booleans [in] indicate number of available radio equipments (0, 1 or 2) [in] value of 1st train phone nr, up to 16 digits [in] value of 2nd train phone nr, up to 16 digits
void	SetETCSID(uint32_t ulETCSId)	Set ETCS identitiy [in] value of ETCS ID/NID_ENGINE (stored on 24 bits)
void	SetBaliseAntennaOffsets(t_distance dBalAntennaOffsetCabA, t_distance dBalAntennaOffsetCabB)	Set offsets of balise antennas [in] value of ETCS ID/NID_ENGINE (stored on 24 bits)
void	SetEirenePhoneNumber(const char* szShrtNr)	Set the stored RBC short number [in] RBC short nr, up to 16 digits

void	SetDefaultTrainData(t_distance dLength, t_speed dSpeedLimit, int32_t lTrainCategory, t_distance dCantDeficiency, int32_t lLoading-Gauge, eAxleLoadCategory, int32_t lAxleNr, int32_t lBrakePrecentage)	Set the default train data [in] length of train (m) [in] maximum train speed limit(m/s) [in] train category calculated from brake position [in] train cant deficiency (m) [in] train loading gauge [in] train axle category [in] train axle number [in] braking percentage
void	SetSBModel(SSBModelParam SBModelParam, bool bDefault)	Set one Service Brake deceleration model with its corresponding brakes configuration [in] Service Brake deceleration model [in] indicates if model must be applied to all brake configurations
void	SetEBModel(SEBModelParam EBModelParam, bool bDefault)	Set one Emergency Brake deceleration model with its corresponding brakes configuration [in] Emergency Brake deceleration model [in] indicates if model must be applied to all brake configurations
void	SetFactorsKn(SKn Kn_p, SKn Kn_n)	Set the on-board correction factors for gradient on normal service deceleration [in] correction factor for positive gradient [in] correction factor for negative gradient
void	SetTractionModel(t_accel dMaxAccel, t_time dTractionCutOffTime)	Set the traction model [in] maximum train acceleration (m/s^2) [in] time to cut off the traction (s)
void	SetOdoError(t_distance dOdoLinErrCoef, t_distance dOdoFixedErr, double dUnder-ErrCoef, double dOverErrCoef)	Set the odometry error model (to simulate inaccuracy in odometry) [in] linear error (m) [in] fixed error (m) [in] coefficient for under-reading error (%) [in] coefficient for over-reading error (%)
void	SetFixedDataSafeConnectionRepeatTimeout (t_time dSafeConnTimeout)	Modify the value of the fixed data timeout [in] time for repetition of radio connection request
void	SetFixedDataMaxSessionMaintainTime(t_time dMaxSessionTime)	Modify the value of the fixed data timeout [in] maximum time to maintain a communication session in case of failed reconnection

2.2 Types

Table 4. Basic types

Data type	Variable name	Comment
Double	t_time	Time in s
Double	t_distance	Distance in m
Double	t_speed	Speed in m/s
Double	t_accel	Acceleration in m/s^2

2.3 Enumerations

Table 5. eConfigData

Enum member	Enum value	Comment
CFG_RADIO_INTERNAL_TIME_STAMP	0	Internal time stamping (not use T_TRAIN)
CFG_USE_JRU	1	Use JRU (generates JRU data file)
CFG_RECORD_TO_CSV_FILE	2	Record supervision data to CSV file
CFG_LOOP_WITH_SSCODE	3	Loop are received with spread spectrum code
CFG_BAL_WITH_ODO_STAMP	4	Balise are received with odo stamp
CFG_LOCAL_TIME_STAMP	5	Request local time stamp otherwise GMT time in log & JRU record
CFG_RECORDER_LOG_ADD_FULL _TIME_STAMP	6	add time stamp in EuroCabLog.dat like 2009-05-29/08:15:21.29
CONFIG_SIZE	7	Maximum number of config value

Table 6. eTrainMode, ETCS train mode

Enum member	Enum value	Comment
MODE_FULL	0	Full supervision
MODE_ON_SIGHT	1	On sight
MODE_STAFF_RESP	2	Staff responsible
MODE_SHUNTING	3	Shunting
MODE_UNFITTED	4	Unfitted
MODE_SLEEPING	5	Sleeping
MODE_STAND_BY	6	Stand by
MODE_TRIP	7	Trip
MODE_POST_TRIP	8	Post trip
MODE_SYST_FAILURE	9	System failure
MODE_ISOLATION	10	Isolation
MODE_NON_LEADING	11	Non leading
MODE_LIMITED_SUP	12	Limited supervision
MODE_REVERSING	14	Reversing

MODE_PASSIVE_SH	15	Passive shunting
MODE_UNKNOWN	16	Unknown
MODE_NO_POWER	17	No power
MODE_MAX_NB	18	Maximum number of mode value

Table 7. eSpeedMonitoringStatus, speed monitoring status

Enum member	Enum value	Comment
SPEEDMS_CSM	0	Ceiling Speed Monitoring
SPEEDMS_PIM	1	Pre-Indication Monitoring
SPEEDMS_TSM	2	Temporary Speed Monitoring
SPEEDMS_RSM	3	Release Speed Monitoring
SPEEDMS_UNKNOWN	4	Unknown Speed Monitoring

Table 8. eBrakePosition, brake position

Enum member	Enum value	Comment
BRK_POS_R	0	Passenger train in P
BRK_POS_P	1	Freight train in P
BRK_POS_G	2	Freight train in G

Table 9. eRadSafeConnectStatus, radio safe connection status

Enum member	Enum value	Comment
CONNECTION_RELEASED	0	Radio safe connection is released
CONNECTION_REQUESTED	1	Radio safe connection is requested
CONNECTION_ESTABLISHED	2	Radio safe connection is established

Table 10. eRadSessionState, radio session state

Enum member	Enum value	Comment
SESSION_RELEASED	0	Session is released
SESSION_INIT	1	Try to establish a session
SESSION_ESTABLISHED	2	Session is established
SESSION_TERMINATED	3	Session is terminated (wait for ack termination)

Table 11. eAxleLoadCategory, train data international category

Enum member	Enum value	Comment
AXLE_CAT_A	0	
AXLE_CAT_HS17	1	
AXLE_CAT_B1	2	
AXLE_CAT_B2	3	
AXLE_CAT_C2	4	
AXLE_CAT_C3	5	
AXLE_CAT_C4	6	
AXLE_CAT_D2	7	
AXLE_CAT_D3	8	
AXLE_CAT_D4	9	
AXLE_CAT_D4XL	10	
AXLE_CAT_E4	11	
AXLE_CAT_E5	12	
MAX_AXLE_CATEGORY	13	

Table 12. eLevel, ETCS level

Enum member	Enum value	Comment
LEVEL_0	0	Level 0
LEVEL_1	2	Level 1
LEVEL_2	3	Level 0
LEVEL_3	4	Level 3
LEVEL_UNKNOWN	5	Level unknown
NR_LEVEL	6	Number of different level

Table 13. eDriverActionInfo, DMI actions from driver

Enum member	Enum value	Comment
DMI_ACTION_DRIVER_ID	0	
DMI_ACTION_TRAIN_RUNNING_NUMBER	1	
DMI_ACTION_TRAIN_DATA	2	
DMI_ACTION_SR_DATA	3	
DMI_ACTION_LEVEL_0	4	
DMI_ACTION_LEVEL_1	5	
DMI_ACTION_LEVEL_2	6	
DMI_ACTION_LEVEL_3	7	
DMI_ACTION_START_OF_MISSION	9	

DMI_ACTION_SLIPPERY_TRACK	10	
DMI_ACTION_NON_SLIPPERY_TRACK	11	
DMI_ACTION_NON_LEADING	12	
DMI_ACTION_NON_LEADING_EXIT	13	
DMI_ACTION_OVERRIDE_EOA	14	
DMI_ACTION_OVERRIDE_RS	15	
DMI_ACTION_SHUNTING	16	
DMI_ACTION_SHUNTING_EXIT	17	
DMI_ACTION_TAF_ACK	18	
DMI_ACTION_ACK_ICON	19	
DMI_ACTION_ACK_TEXT	20	
DMI_ACTION_ACK_BRAKE	21	

Table 14. eMSG_TYPE, type of data message transmitted to MMI

Enum member	Enum value	Comment
MSG_BALISE	0	Balise message
MSG_LOOP	1	Loop message
MSG_RADIO	2	Radio message
MSG_MMI	3	DMI message
MSG_INFO2	4	Informative message

2.4 Structures

Table 15. SVarDataList, contains information about a track condition

Member type	Member name	Comment
t_distance	dStartLocation	Start location of TC (m)
t_distance	dEndLocation	End location of TC (m)
eTCInfo	TCInfo	Type of track condition
SVarDataList	VarList	Data associated to the track condition

Table 16. SVarData, contains variable and its value (used to indicate driver data entry on DMI)

Member type	Member name	Comment
string	VarName	Name of data
string	VarValue	Value of data

Table 17. SKn, contains Kn data

Member type	Member name	Comment
int32_t	lNrPt	Nr of samples in the model
SFactorPerSpeed	Pt[MAX_KN_PT]	data

Table 18. SFactorPerSpeed, contains Kn data

Member type	Member name	Comment
double	dFactor	Correction factor
t_speed	dSpeed	Applicable speed for correction factor