LEONARDO J. VALDIVIA, Ph.D. Candidate.[1]
(Corresponding author)
E-mail: lvaldivia@ceit.es
GONZALO SOLAS, Ph.D.[1]
E-mail: gsolas@ceit.es
JAVIER AÑORGA, Ph.D.[1]
E-mail: jabenito@ceit.es
SAIOA ARRIZABALAGA, Ph.D.[1]
E-mail: sarrizabalaga@ceit.es
IÑIGO ADIN, Ph.D.[1]
E-mail: iadin@ceit.es
JAIZKI MENDIZABAL, Ph.D.[1]
E-mail: jmendizabal@ceit.es
[1] CEIT and Tecnun, University of Navarra
Manuel de Lardizabal 15, 20018 San Sebastian, Spain

Intelligent Transport Systems (ITS)
Preliminary Communication
Submitted: 18 May 2016
Accepted: 16 Nov. 2016

ETCS ON-BOARD UNIT SAFETY TESTING: SABOTEURS, TESTING STRATEGY AND RESULTS

ABSTRACT

It is necessary to verify the faults tolerance of the European Train Control System (ETCS) on-board unit even if these faults are uncommon. Traditional test methods defined and used in ETCS do not allow to check this, so it is necessary to develop a new mechanism of tests. This paper presents the design and implementation of a saboteur applied to the railway sector. The main purpose of the saboteur is the fault injection in the communication interfaces. By means of a virtual laboratory it is possible to simulate actual train journeys to test the ETCS on-board unit. Making use of the saboteurs and the virtual laboratory it is possible to analyse the behaviour of the train in the presence of unexpected faults, and to verify that the decisions taken are correct to ensure the required safety level. Therefore, this work shows a testing strategy based on different kinds of train journeys when faults are injected, and the analysis of the results.

KEY WORDS

saboteur; fault injection; virtual laboratory; ETCS; on-board unit; testing strategy;

1. INTRODUCTION

Currently, all the railway signalling systems manufacturers have to be compatible with the European Train Control System (ETCS), i.e. the development of any railway product has to meet the standards and tests specified by ETCS. All the devices designed following the ETCS norms must be safety critical, which means that errors must not occur and they must be fault tolerant. To achieve this, standards like EN-50129 [1] should be used. This norm specifies that safety testing is required, but there is no technical specification in the standards on how to implement these tests. Moreover, the ETCS standards SUBSET-076 [2] and SUBSET-094 [3] that are used as reference to implement the tests of the ETCS On-Board Unit (OBU) do not list any safety testing.

With the aim of accomplishing safety testing, fault injection techniques can be employed. Currently, the behaviour of the train modules in the presence of faults is not tested according to ETCS standards. In real operation when an unexpected fault appears, the system faces a non-tested situation. This is why the systems should be tested in worse case scenarios including faults.

Therefore, to test the modules including safety assessment according to EN-50129 [1] it is necessary to inject faults in the inputs of different modules. For the case of the ETCS OBU it is possible to simulate that wrong information is received due to different factors that could be the missing of a balise, wrong speed information, wrong balise position, etc. This allows testing of the train in the presence of unexpected faults and observing the behaviour of the on-board computer. Therefore, fault injection techniques can be employed by means of saboteurs in communications between physically independent parts.

This paper is structured as follows: Firstly, the state of the art describes ETCS and the fault injection techniques. Afterwards, the fault list that might cause a core hazard and type of voters are detailed. The next sections deal with the features of the virtual laboratory used to carry out the test. Then, in the fifth section, the implementation of the saboteurs is described. The following section shows the strategy of performing the tests. The seventh section deals with the results of the carried out tests. And finally, the conclusions of the research work are presented.

2. STATE OF THE ART

The state of the art is divided in two subsections: firstly, the descriptions of the European Rail Traffic Management System (ERTMS) and ETCS are presented, and later the fault injection techniques are detailed.

2.1 ERTMS

One of the main components of ERTMS is the ETCS. It was designed to replace different national safety systems used in railway that are incompatible among them. This allows to have the same signalling and control system along multiple countries regardless of the train manufacturer [4]. ETCS is specified at four different levels [5]:
Level 0: there is no communication between the train and the track; the driver needs to observe the track signals.
Level 1: the communication between the train and the track is via ETCS balises. It allows communication up to 300 km/h.
Level 2: ETCS data transmission is continuous by means of wireless communications.
Level 3: similar to level 2 but with on-board positioning and train integrity allowing the moving block technology.

On the other hand, ETCS is divided into two parts [6]:
- ETCS on-board equipment: composed of an EVC (European Vital Computer) computer and its peripherals.
- ETCS trackside equipment: this always includes balises. It can also include Radio Block Centres (RBC), Euroloops and Radio In-fill Units.

The on-board equipment is safety critical and according to the specifications SUBSET-091 [7] and SUBSET-088 [10] it must has a Tolerable Hazard Rate (THR) less than 10^{-9} F/h, or in others words, it is allowed to have maximum one fault every 10^9 hours. For each function on the train a THR is assigned to identify how it contributes to the core hazard. This is described in SUBSET-091 [7]. Therefore, in order to reach this THR the on-board system should be designed against failures.

However, when it comes to testing and assessing the safety level, the ERTMS normative does not mention how to test this capability. Previous European projects such as EMSET [8] and INESS [9] only defined functional testing as the ETCS norms SUBSET-076 [2] and SUBSET-094 [3]. For trackside equipment norms like SUBSET-085 [11] for balises and SUBSET-103 [12] for Euroloop describe the tests needed to approve these devices, but again these standards only address functional tests. However, the normative for safety critical systems, such as ETCS, EN-50129 [1] for railways and EN-61508 [13] generic, specify that fault injection is required to ensure safety in any electronic device used in safety critical devices like railway.

2.2 Fault injection techniques

Fault Injection Techniques (FIT) are considered as a very useful approach to evaluate the dependability of a system, and also to reduce costs of field testing [14, 15]. Different FIT types include:
- Hardware-based techniques: this technique allows testing of the circuit resistance against external elements, for example, electromagnetic waves, radiation or signals injected in the I/O pins. An advantage of this technique is that it is possible to test real environment conditions. On the other hand, a disadvantage is that expensive external hardware is required [16].
- Software-based techniques: it is only necessary to change the source code to inject a fault in the device under test, compile the software and run it to see the behaviour of the device. An advantage of this technique is that it is low complexity and cost. A disadvantage is that it is not possible to test all used cases [17, 18].
- Emulation-based techniques: with the emergence of the hardware description languages this technique became popular. The main advantage is that it allows testing of complex circuits with a better performance than simulation, and hence the development time can be reduced. On the other hand, a disadvantage is the additional hardware to emulate, which can be very expensive [19].
- Simulation-based techniques: the device under test is modelled in a simulation tool to test it in different situations. An advantage is the possibility to test the system in any step of the development process or even when it is already implemented. The main drawback is the complexity of certain scenarios; sometimes it is not possible to simulate all the cases [20].
- Hybrid techniques: combining two or more of the above ones [21].

When fault injection is performed during the execution of a simulation, an external device to introduce the fault is required. This device is usually called saboteur. Using saboteurs allows testing of a system or device in a non-invasive way, i.e. no changes on the software or hardware are needed.

3. FAULT LIST AND TYPES OF SABOTEURS

The saboteur is a device that performs the fault injection. This section details the faults that can cause

a core hazard on trains. For each fault, the on-board module involved is detected. Then, it is possible to define all the saboteurs required to simulate the faults. Also an overview of the saboteur's functionality is presented.

3.1 Fault list

A dangerous failure for the ETCS on-board equipment is defined as: "*failure to provide on-board supervision and protection according to the information provided to the ETCS on-board from external entities*" [7], and only failures that lead to the ETCS Core Hazard (exceeding the safe speed or distance information provided to ETCS) need to be considered.

Table 1 shows a list of the events that might cause an ETCS Core Hazard, either alone or in combination with other failures. Additionally, it contains a description of each fault and the on-board element that is involved [7]. There were 27 different events that were identified. Each of these events has to be tested to verify the safety capabilities of the OBU.

3.2 Type of saboteurs

Table 1 shows that five on-board elements can cause the core hazard, hence one saboteur per element should be implemented to test all the events. The list of the saboteurs includes: BTM, LTM, ODO, TIU and DMI.

The saboteur is connected in the communications interface of two devices, and its objective is to alter the information of the messages. This is why this kind of devices is also called interface saboteurs [22, 23].

When the saboteur is not injecting faults it acts as a bridge between the two devices, i.e. the communication is not altered, so the devices can send messages between them without any problem. *Figure 1* shows the communication between two devices when an interface saboteur is connected and injects a fault to

Table 1 – Faults list that might cause an ETCS Core Hazard

Event Id	Event Description	On-board module
MMI-1a	False acknowledgment of mode change from full supervision	Driver Machine Interface [DMI]
MMI-1b	False command to enter non-leading mode	DMI
MMI-1c	False command of override End of Authority (EoA) request	DMI
MMI-1d	False acknowledgment of level transition	DMI
MMI-1e	False acknowledgment of train trip	DMI
MMI-1f	False acknowledgment of track ahead free	DMI
MMI-2a	False presentation of speed or distance on the DMI	DMI
MMI-2b	False presentation of mode on the DMI	DMI
MMI-3	Falsification of driver's train data input	DMI
MMI-4	Frozen or delayed DMI display	DMI
ODO-1	Incorrect standstill indication	Odometer (ODO)
ODO-2	Speed measurement underestimates trains actual speed	ODO
ODO-3	Incorrect actual physical speed direction	ODO
ODO-4	Distance measurement is incorrect	ODO
TI-1	Service brake/emergency brake not commanded when required	Train Interface Unit (TIU)
TI-2	Service brake/emergency brake release commanded when not required	TIU
TI-3	Inappropriate sleeping request	TIU
TI-4	Incorrect brake status (TIU failure)	TIU
TI-5	Incorrect direction controller position report (TIU failure)	TIU
TI-6a	Loss of cabin active signal	TIU
TI-6b	Wrong cabin considered as Active	TIU
BTM-H1	A balise group is not detected, due to failure at the on-board Balise Transmission Module (BTM) function	BTM
BTM-H4	Transmission of an erroneous telegram, interpretable as correct	BTM
BTM-H7	Erroneous localization of a balise Group, with reception of valid telegrams	BTM
BTM-H8	The order of reported balises, with reception of valid telegrams	BTM
BTM-H9	Erroneous reporting of a balise group in a different track, with reception of valid telegrams	BTM
LTM-H4	Transmission of an erroneous telegram, interpretable as correct	Loop Transmission Module (LTM)

a message sent from Device A to Device B. The saboteurs operate as follows: firstly, the saboteur intercepts the message, while the Device A (device which sent the message) is waiting for a confirmation of reception. The saboteur responds with the expected message to Device A. After that, the saboteur alters the message information and sends it to Device B.

Figure 1 – Saboteur actions between devices

4. VIRTUAL LABORATORY

This section describes the virtual laboratory used to perform the tests. Firstly, the architecture of the laboratory including the saboteurs is detailed. Then, an explanation of all the applications that make up the virtual laboratory is presented. And finally, the steps required to perform a simulation and the messages format are described.

4.1 Virtual laboratory architecture

A virtual laboratory is a platform where physical equipment is replaced with computational models, in the field of railway, current tools simulate the behaviour of the telecommunication and control systems [24, 25]. The virtual lab presented in the project "ETCS Advanced Testing and Smart Train Positioning System" (EATS) [26], allows simulation of the train OBU during a trip, and thanks to the saboteurs it is possible to inject faults that would not occur otherwise during normal operation. With this the behaviour of the OBU under unexpected conditions can be analysed.

Figure 2 shows the on-board ERTMS testing laboratory architecture described in [3] but the interface saboteurs are added. The virtual lab follows this architecture, simulating all the elements.

4.2 Virtual laboratory applications

The virtual laboratory has five applications that allow the simulation of a journey [26]:

– Even feeder: this application reads an XML file with the events of a journey and sends them to the

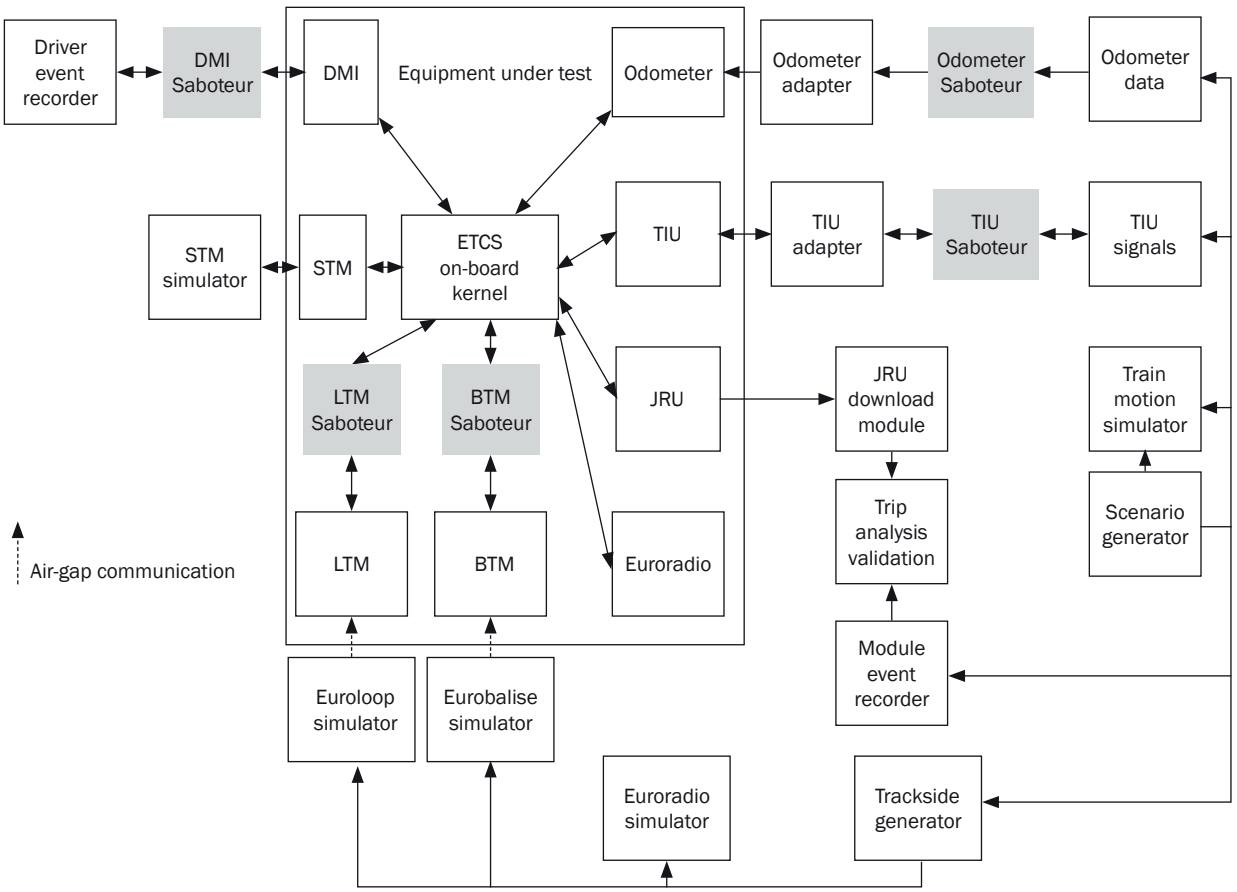

Figure 2 – ETCS on-board simulation lab [26]

corresponding element. The XML files are created with information taken from the real train trips and they contain odometer, balise, Euroloop, train interface and euro radio events.
- Juridical Recording Unit (JRU) application: it creates an XML file when the simulation ends with all the trip information.
- Controller: this application configures and starts all the other elements.
- ETCS kernel: the kernel consists of two applications, the EVC and the ERTMS Formal Specs (EFS) [27]. The EVC receives all the information or messages from all the other lab elements (odometer, balise, Euroloop, etc.). On the other hand, when all the events are received the EFS simulates the train journey.
- Driver Machine Interface (DMI): this application simulates the driver's interface of a real train.

4.3 Virtual laboratory simulation

The virtual laboratory uses a custom messages format for each interface. *Figure 3* shows the format for the odometer, balise and Euroloop events. This means that the journey events (XML file) are converted to this format and they are sent to the other elements by the event feeder.

To simulate a journey in the virtual laboratory the next steps are performed:
1) All the applications must be configured with the controller.
2) Choose journey to simulate.
3) The event feeder sends the events to the corresponding elements (BTM, LTM, etc.).
4) All the elements (BTM, LTM, etc.) process the messages received from the event feeder and they send the new messages to the EVC.
5) The EVC receives the messages from all the elements and saves them.
6) When all the events are received the EVC sends the events to the EFS and the simulation starts.
7) The DMI receives information from the EFS to show it.
8) When the simulation ends a JRU file is created with the simulation information.

5. IMPLEMENTATION OF THE SABOTEURS

The next subsections describe how the saboteurs are implemented and the process to detect the network packages in order to be able to inject the faults defined in *Table 1*.

5.1 Features of the interface saboteur

Interface saboteurs for virtual lab test bench are implemented through a commercial Linux board. The decision of choosing a Linux board is based on the features that this operating system (OS) offers, which enables the use of all routing characteristics like IP-tables.

The saboteur's features are explained below:
- Database: each saboteur contains a database with the information of the actions that can be performed.
- Listening controller: this software applies all the IP-tables rules in order to redirect the messages to the saboteur.
- Actuator: this application receives the redirected messages and performs the sabotage based on the information stored in the database.

Figure 4 shows a configuration and running software execution order for interface saboteurs. First the external controller configures the database and makes a TCP/IP connection with the listening controller. After that, the listening controller interprets filtering information from database and applies IP-tables. Finally, the listening controller runs as many actuators at proper ports based on the filtering rules as specified in the database.

EuroLoop Event

HEADER	event_time	seq	bytes
11 bytes	8 bytes	4 bytes	n bytes

HEADER

eventType	time	length
1 byte	8 bytes	2 bytes

Balise event

HEADER	event_time	nid_big	n_pig	t_report	seq	bytes
11 bytes	8 bytes	4 bytes	4 bytes	8 bytes	4 bytes	n bytes

Odometry Event

HEADER	event_time	speed	position	acceleration	seq
11 bytes	8 bytes	8 bytes	8 bytes	8 bytes	4 bytes

Figure 3 – Message format for odometer, balise and Euroloop events in the virtual lab

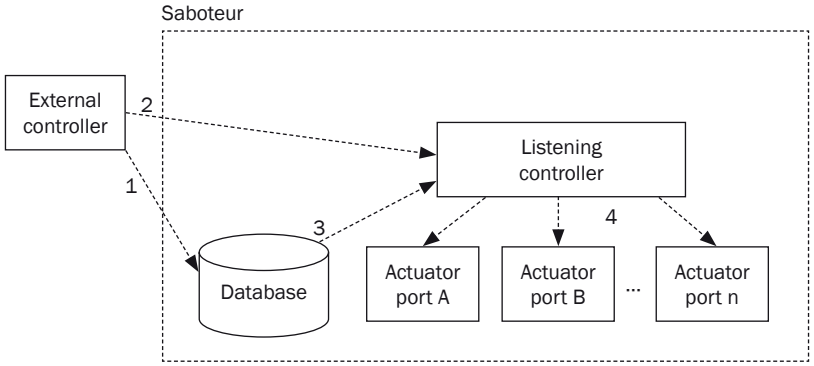

Figure 4 – Configuration and running software execution order for saboteur

5.2 Saboteur actions

This section describes the process of how the network messages are sabotaged. To achieve this, the saboteurs must be configured, they must capture the network traffic and modify the information at the communication protocol layer.

The listening controller receives all the TCP messages and when one message matches the filter criteria it is sent to the actuator. On the other hand, if the packet does not match the criteria the saboteur acts as a bridge and the message is sent to the original end-point. The filter criteria are stored in the database and are based on these parameters: source IP, destination IP, source port, destination port and event type (this is inside the packet payload, see *Figure 3*).

At this point, the actuator at the interface saboteur elaborates a reply to the source IP / source Port of the redirected packet, since communication protocol (TCP/IP) defines an echo response of the end device for each packet received containing data. The interface saboteur does not need to spoof any IP or MAC address since the sending or replying IP or MAC address has no impact on the logic functionality of the lab testbed software.

Table 2 – Fault chains related to faults defined for BTM module

Module	Fault	Chain id	Fault function
BTM	BTM-H1	IS-1	Suppression
	BTM-H4	IS-2	Bytes random
	BTM-H8	IS-4	Flip
	BTM-H9	IS-5	Creation

When the response is sent it is time to perform the sabotage. To do this it is necessary to get the information from the database. The database has two main tables: the first one has the information about all faults that can be injected and the other table has the time in which each fault is injected. *Table 2* shows the information about BTM faults. The “Chain id” indicates the table name that contains the information about the times to inject the faults.

Table 3 illustrates an example of the information stored in IS-2 which relates to BTM-H4 “transmission of an erroneous telegram, interpretable as correct”. The time at which the fault is injected is random, but it has to be within two ranges (t_on and t_off).

Table 3 – Example of four fault windows for IS-2 fault chain

id	t_on	t_off	Offset	nbytes	Res
0	300.56	300.58	45	45	NOT
1	301.66	302	45	45	NOT
2	400.98	401	45	45	NOT

Once fault information is inserted into the packet, it has to be sent to the original communication end-point. For this purpose, the saboteur establishes a TCP/IP connection with the original communication. The saboteur acts in the same way as the original communication, which means that TCP/IP connection from the saboteur with the end-point only is made when the original starting-point is establishing the connection, and it is released when the original starting-point releases the connection.

6. TEST STRATEGY

This section shows the test strategy designed in order to detect the faults with more impact in the system behaviour.

The main objective of the test strategy is to detect the worst case scenarios for the system; therefore, the test was performed following this plan of action:

1) A reference journey is simulated with each fault defined in *Table 1* (Section 3.1), which means that 27 tests are performed.
2) The faults are applied in the following way: for ODO and DMI, time windows for fault injection are included and the faults are injected randomly. On the other hand, for BTM, LTM and TIU faults are injected in all the events.

3) Identify the worst faults according to their effect in the reference journey.
4) Apply the worst faults to other test journeys.

The file used as reference journey was chosen because it contains all types of messages (ODO, TIU, DMI, LTM and BTM). On the other hand, the journeys selected to test the worst faults were created to test particular situations on the train. *Table 4* shows the twelve journeys and the element under test.

To verify the effect of the saboteurs, a simulation without faults is performed and the JRU file is saved, so it is possible to compare this file with the JRUs with faults. Another way to see the effect of the saboteurs is in the DMI; if any change occurs it should be shown in the DMI.

7. RESULTS

This section shows the analysis and the results of the tests made following the strategy proposed in the previous chapter. First, the results obtained using a reference journey are analysed with the aim of identifying the worst cases based on the result of the ETCS OBU behaviour. Then, the results of the worst cases injected to other scenarios are presented.

7.1 Reference journey results

Table 5 shows all the obtained results when the faults are applied to the reference journey.

With the results obtained so far (see *Table 5*) it can be concluded that the DMI faults only have effect in the driver's screen, i.e. the JRU does not detect the changes made by the saboteur, because the faults injected directly to the DMI do not have any effects in the OBU. Moreover, in TIU faults some changes in the JRU file are observed, but this does not affect the OBU. There are two kinds of TIU messages: the messages sent by the TIU that just informs about the status of some elements (brake, cab, etc.) and the messages sent by the EVC to change the status of the elements. The TIU saboteur alters the messages that inform about the status, and these produce changes in the JRU but they do not have any effect on the system.

On the other hand, in the odometer faults it can be observed that if speed is changed and exceeds the limit the train activates the emergency brake. Thus, this fault significantly affects the simulated journey. The same behaviour occurs with balise or Euroloop faults when they are applied to elements that contain movement authority information. If one of these elements is not detected the emergency brake is also activated.

As mentioned in Section 6 to check the effects of the faults the JRU and the DMI are used. *Figure 5* shows the DMI when the fault ODO-2 is injected, and the emergency brake is activated [28].

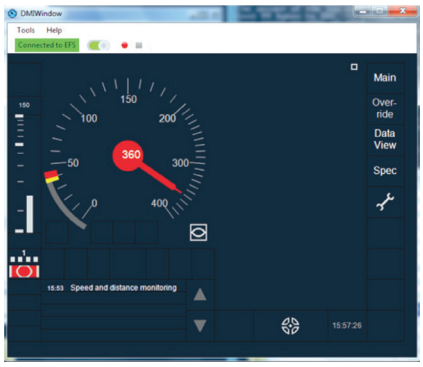

Figure 5 – DMI when the fault ODO-2 is injected

Figure 6a shows the JRU file of the reference journey in a normal operation. On the other hand, *Figure 6b* shows the JRU file in the same point, but fault BTM-H1 is injected. As it can be observed the emergency brake is activated.

```
<StoredJruEvent xsi:type="JruBaliseEvent">
  <HEADER>
  <TELEGRAM>
    <Q_UPDOWN>Messages.Q_UPDOWN.Down_link_t
    <M_VERSION>16</M_VERSION>
    <Q_MEDIA>Messages.Q_MEDIA.Balise</Q_MEDI
    <N_PIG>0</N_PIG>
    <N_TOTAL>1</N_TOTAL>
    <M_DUP>Messages.M_DUP.No_duplicates</M_I
    <M_MCOUNT>14</M_MCOUNT>
    <NID_C>255</NID_C>
    <NID_BG>2081</NID_BG>
    <Q_LINK>Messages.Q_LINK.Linked</Q_LINK>
    <BitField>90-02-07-1F-E4-10-CB-10-18-03-
    <Sequence1>
  </TELEGRAM>
</StoredJruEvent>
<StoredJruEvent xsi:type="JruBaliseEvent">
<StoredJruEvent xsi:type="JruGeneralEvent">
<StoredJruEvent xsi:type="JruGeneralEvent">
```

a) JRU file in a simulation without faults

```
<StoredJruEvent xsi:type="JruBaliseEvent">
  <HEADER>
  <TELEGRAM>
    <Q_UPDOWN>Messages.Q_UPDOWN.Down_link_telegram</Q_UPD
    <M_VERSION>0</M_VERSION>
    <Q_MEDIA>Messages.Q_MEDIA.Balise</Q_MEDIA>
    <N_PIG>0</N_PIG>
    <N_TOTAL>1</N_TOTAL>
    <M_DUP>Messages.M_DUP.No_duplicates</M_DUP>
    <M_MCOUNT>14</M_MCOUNT>
    <NID_C>255</NID_C>
    <NID_BG>2080</NID_BG>
    <Q_LINK>Messages.Q_LINK.Unlinked</Q_LINK>
    <BitField>90-02-07-1F-E4-10-15-15-15-15-15-15-15-15-1
    <Sequence1>
  </TELEGRAM>
</StoredJruEvent>
<StoredJruEvent xsi:type="JruGeneralEvent">
<StoredJruEvent xsi:type="JruGeneralEvent">
<StoredJruEvent xsi:type="JruEmergencyBrakeEvent">
  <HEADER>
  <M_BRAKE_COMMAND_STATUS>M_BRAKE_COMMAND_STATE.Commanded
</StoredJruEvent>
```

b) JRU file when the fault BTM-H1 is injected

Figure 6 – JRU files of the reference journey simulation

Table 4 – List of the journeys to test the worst cases

Journey name	Element under test	Annotations
Scenario 1	Trip at Movement Authority (MA) end	
Scenario 2	Level Transition	Level 0 to Level 1
Scenario 3	Level Transition	Level 1 to Level 0
Scenario 4	Mode Transition due to Mode Profile 1	Full Supervision (FS) to On Sight
Scenario 5	Mode Transition due to Mode Profile 2	FS to Shunting (SH)
Scenario 6	Trip because missing balise in linked	
Scenario 7	Mode Transition due to MA	Staff Responsible (SR) to FS
Scenario 8	Level Transition 2	Level 1/Level 0 to Level 2
Scenario 9	Level Transition 3	Level 2 to Level 1/Level 0
Scenario 10	Trip because MA not received	
Scenario 11	Stop if in Staff responsible	
Scenario 12	Stop if in SH	

Table 5 – Results obtained in the reference journey

Event Id	Results obtained
MMI-1a	Two extra messages can be observed in DMI: the request to full supervision and the driver's answer
MMI-1b	Two extra messages can be observed in DMI: the non-leading request and the driver's answer
MMI-1c	Two extra messages can be observed in DMI: the override request and the driver's answer
MMI-1d	Two extra messages can be observed in DMI: the acknowledgment of level transition and the driver's answer
MMI-1e	Two extra messages can be observed in DMI: the acknowledgment of train trip and the driver's answer
MMI-1f	Two extra messages can be observed in DMI: the request of track ahead free and the driver's answer
MMI-2a	A different speed is shown in the DMI
MMI-2b	An extra ETCS mode messages is observed
MMI-3	Two extra messages can be observed in DMI: the train data entry request and the driver's answer
MMI-4	The information shown in the DMI is delayed
ODO-1	No changes were observed
ODO-2	The emergency brake is activated
ODO-3	No changes were observed
ODO-4	No changes were observed
TI-1	No changes were observed
TI-2	The JRU file with the injected faults contains just two brake events compared with the six events shown in the JRU without faults.
TI-3	An extra sleeping request message is observed in the JRU
TI-4	The JRU file with the injected faults contains just two brake events compared with the six events shown in the JRU without faults
TI-5	No changes were observed
TI-6a	A message of cab status not active is added to the JRU
TI-6b	A message of cab status not active is added to the JRU
BTM-H1	If the deleted balise contains MA information, the emergency brake is activated. Otherwise no changes were observed
BTM-H4	If the edited balise contains MA information, the emergency brake is activated. Otherwise no changes were observed
BTM-H7	No changes were observed
BTM-H8	If the balise contains MA information, the emergency brake is activated. Otherwise no changes were observed
BTM-H9	No changes were observed
LTM-H4	If the deleted Euroloop contains MA information, the emergency brake is activated. Otherwise no changes were observed

The tests made show that the worst effect on the OBU is when the faults cause the activation of the emergency brake, therefore, two faults were chosen as the worst cases: ODO-2, when the speed exceeds the limit the emergency brake is activated and BTM-H1, when the OBU expects a balise or Euroloop with movement authority and it is not detected the emergency brake is also activated.

7.2 Worst case scenarios

Finally, the worst faults are injected to the whole set of journeys (see *Table 4*). This means that 24 tests are performed (12 with the ODO-2 and 12 with BTM-H1). *Table 6* shows the results obtained when the fault BTM-H1 was injected to each journey.

Table 6 – Result obtained when the BTM-H1 is injected

Journey name	Results obtained	Comments
Scenario 1	If the balise contains MA information, the emergency brake is activated. Otherwise no changes are observed	
Scenario 2	Same effect as scenario 1	
Scenario 3	Same effect as scenario 1	
Scenario 4	Same effect as scenario 1	
Scenario 5	Same effect as scenario 1	
Scenario 6	Due to the nature of the scenario, faults cannot be injected	The emergency brake is activated before the injection
Scenario 7	No changes were observed	The journey does not contain balises with MA
Scenario 8	Same effect as scenario 1	
Scenario 9	Same effect as scenario 1	
Scenario 10	Same effect as scenario 1	
Scenario 11	No changes were observed	The journey does not contain balises with MA
Scenario 12	No changes were observed	The journey does not contain balises with MA

On the other hand, when fault ODO-2 is injected the result obtained in all the journeys is the same: the emergency brake is activated. *Figure 7* shows the JRU of the scenario 2 when the fault is injected, the emergency brake is activated and it is possible to observe the speed of the train of 360 km/h (the speed in the JRU file is divided by 5, and that is why it is shown as 72).

Figure 7 – JRU when the fault ODO-2 is injected in the scenario 2

The results obtained in the 12 scenarios are very similar to the reference journey, i.e., when an unexpected situation occurs during the journey, the OBU actives the emergency brake. Since this is a safety critical system and the safe mode is to apply the brakes, it can be stated that this is the correct behaviour of the system [29].

8. CONCLUSION

A new testing platform for the railway sector using saboteurs and a virtual laboratory was created. It allows verifying the behaviour of the train OBU to unexpected faults in different interfaces (BTM, LTM, ODO, DMI and TIU). To check the consequences of faults, they were injected in distinct journeys and situations which proves the flexibility of the created platform.

The testing strategy allowed testing of all the events that can cause an ETCS core hazard, and the faults that have more impact on the OBU were detected. After that, the detected faults were analysed in multiple journeys to verify the safety in all cases.

The worst case is the activation of the emergency brake, which only occurs with BTM and ODO faults. However, it is important to note that all the BTM faults have the same effect on the OBU because if a balise with MA is changed or cannot be decoded for the EVC it is equal to delete it.

The tests made in all the journeys show that the emergency brake is activated in two situations: when

the train does not detect a balise with movement authority information and when the odometer's speed exceeds the limit. This is the correct behaviour, because in unforeseen situations the OBU goes to the safe mode.

Therefore, it is possible to affirm that the platform presented allows testing of the train's OBU against faults, which is a must in safety-critical systems. Moreover, the tests are performed in a virtual laboratory (simulation), reducing cost and time.

ACKNOWLEDGEMENT

This work was supported by the European Community's Framework Program FP7/2007-2013 in the frame of EATS project under the Grant agreement No. 31419.

This work was also supported by the Spanish Ministry of Economy and Competitiveness through the SAREMSIG TEC2013-47012-C2-1-R project (Contribution to a Safe Railway Operation: Evaluating the effect of Electromagnetic Disturbances on Railway Control Signalling Systems) and funded under the call Programa Estatal de Investigación y Desarrollo e Innovación and oriented towards Retos de la Sociedad 2013.

Candidato a Dr. **LEONARDO J. VALDIVIA**[1]
E-mail: lvaldivia@ceit.es
Dr. **GONZALO SOLAS**[1]
Email: gsolas@ceit.es
Dr. **JAVIER AÑORGA**[1]
Email: jabenito@ceit.es
Dr. **SAIOA ARRIZABALAGA**[1]
Email: sarrizabalaga@ceit.es
Dr. **IÑIGO ADIN**[1]
Email: iadin@ceit.es
Dr. **JAIZKI MENDIZABAL**[1]
Email: jmendizabal@ceit.es
[1] CEIT y Tecnun, Universidad de Navarra
Manuel de Lardizábal 15, 20018 San Sebastián, España

PRUEBAS DE SEGURIDAD DE LA UNIDAD DE ABORDO ETCS: SABOTEADORES, ESTRATEGIA DE PRUEBAS Y RESULTADOS

RESUMEN

En el sistema Europeo de control de trenes es necesario verificar la tolerancia a fallos de los sistemas de a bordo, incluso si estos fallos son poco frecuentes. Los métodos tradicionales definidos y utilizados por los trenes Europeos no permiten comprobar esto, por lo que es necesario desarrollar nuevos mecanismos de pruebas. Este trabajo presenta el diseño e implementación de un saboteador aplicado al sector ferroviario, el objetivo principal de dicho saboteador es la inyección de fallos en las interfaces de comunicación. Utilizando un laboratorio virtual es posible simular el recorrido de un tren para poner a prueba la unidad de a bordo. Haciendo uso del laboratorio y de los saboteadores es posible analizar el comportamiento del tren ante fallos inesperados, y además se puede verificar si las decisiones tomadas por el tren son las adecuadas para garantizar el nivel de seguridad. Por lo tanto, este articulo muestra una estrategia de pruebas basada en múltiples recorridos de tren y aplicando fallo, finalmente se presenta el análisis de resultados.

PALABRAS CLAVE

Saboteador; Inyección de fallos; Laboratorio virtual; ETCS; Unidad de a bordo; estrategia de pruebas;

REFERENCES

[1] CENELEC. EN50129, Railway applications - Communication, signalling and processing systems - Safety related electronic systems for signalling. Brussels: CENELEC; 2005.
[2] UNISIG. SUBSET-076, ERTMS/ETCS Class 1, test plan. Brussels: UNISIG; 2009.
[3] UNISIG. SUBSET-094, Functional Requirements for an on board Reference Test Facility. Brussels: UNISIG; 2009.
[4] International Union Of Railways. ETCS [Internet]. 2015 Aug 08 [cited 2016 Apr 26]. Available from: http://www.uic.org/ETCS
[5] The European Rail Traffic Management System. ERTMS in 10 questions [Internet]. 2014 Jan 14 [cited 2016 Apr 20]. Available from: http://www.ertms.net/?page_id=23
[6] RSSB. GE/GN8605 ETCS System Description. London: RSSB; 2010.
[7] UNISIG. SUBSET-091, Safety Requirements for the Technical Interoperability of ETCS in Levels 1 & 2. Brussels: UNISIG; 2009.
[8] CEDEX. Eurocab Madrid-Seville European tests (EMSET), Cordis FP4. Madrid: CEDEX; 1999.
[9] UIC. Integrated European Signalling System (INESS). Paris: UIC; 2012.
[10] UNISIG. SUBSET-088, ETCS Application Levels 1 & 2 - Safety Analysis. Brussels: UNISIG; 2012.
[11] UNISIG. SUBSET-085, Test Specification for Eurobalise FFFIS no 3. Brussels: UNISIG; 2012.
[12] UNISIG. SUBSET-103, Test Specification for Euroloop no 1. Brussels: UNISIG; 2012.
[13] CENELEC. EN61508, Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 2: Requirements for electrical/electronic/programmable electronic safety-related systems. Brussels: UNISIG; 2000.
[14] Mendizabal J. Methodology & Tools for the Design & Verification of SIL4 SW Based on MDD [PhD thesis]. San Sebastian: University of Navarra; 2012.
[15] Hsueh M-C, Tsai TK, Iyer RK. Fault injection techniques and tools. IEEE Computer Society. 2002 Aug;30(4): 75-82.
[16] Karlsson J, Folkesson P, Arlat J, Crouzet Y, Leber G, Reisinger J. Application of Three Physical Fault Injection Techniques to the Experimental Assessment of the MARS Architecture. IEEE Int'l Working Conference

on Dependable Computing for Critical Applications; 1998; California, USA.

[17] Looker N, Munro M, Xu J. A comparison of network level fault injection with code insertion. 29th Annual International Computer Software and Applications Conference; 2005 Jul 26-28; Edinburgh, Scotland. California: IEEE; 2005.

[18] Arlat J, Crouzet Y. Comparison of physical and software-implemented fault injection techniques. IEEE Transactions on Computers. 2003 Sept;52(9): 1115-1133.

[19] Baraza JC, Gracia J, Gil D, Gil PJ. A prototype of a VHDL-based fault injection tool: Description and application. Journal of Systems Architecture. 2002 Apr;47(10):847-867.

[20] Folkesson P, Svensson S, Karlsson J. A comparison of simulation based and scan chain implemented fault injection. Twenty-Eighth Annual International Symposium on Fault-Tolerant Computing; 1998 Jun 23-25. IEEE; 2002.

[21] Ejlali A, Miremadi SG, Zarandi H, Asadi G, Sarmadi SB. A Hybrid Fault Injection Approach Based on Simulation and Emulation Co-operation. International Conference on Dependable Systems and Networks; 2003 Jun 22-25; San Francisco, USA. IEEE; 2003.

[22] Solas G, Mendizabal J, Valdivia L, Añorga J, Adín I, Podhorski A, et al. Development of an Advanced Laboratory for ETCS applications. Transport Research Arena Conference; 2016 Apr 18-21; Warsaw, Poland. Padova: Elsevier; 2016.

[23] Solas G, Valdivia L, Añorga J, Podhorski A, Mendizabal J, Pinte S, Marcos L. Virtual Laboratory for on-board ETCS equipment. IEEE 18th International Conference on Intelligent Transportation Systems; 2015 Sept 15-18; Canary, Spain. IEEE; 2015.

[24] Sondi P, Berbineau M, Kassab M, Wahl M, Gransart C, Lemaire E, Mariano G, et al. Virtual lab based on co-simulation to include impairments of wireless telecommunication such as GSM-R in the evaluation of ERTMS International Union of Railway. Transport Research Arena Conference; 2014 Apr 14-17; Paris, France.

[25] Aguado M, Pinedo C, Lopez I, Ugalde I, de Las Munecas C, Rodriguez L, Jacob E. Towards zero on-site testing: Advanced traffic management & control systems simulation framework including communication KPIs and response to failure events. IEEE 6th International Symposium on Wireless Vehicular Communications; 2014 Sept 14-15; Vancouver, Canada. IEEE; 2014.

[26] CEIT, ESOL, FRAUNHOFER, NSL, TRIT, UGLA, INTEGRASYS. EATS: ETCS Advanced Testing and Smart Train Positioning System, FP7 agreement nr. 31419. Brussels: CEIT; 2012.

[27] Ferier L, Lukicheva S, Pinte S. Formal Methods Applied to Industrial Complex Systems. 1st ed. Hoboken, USA: John Wiley & Sons, Inc; 2014.

[28] European Railway Agency. ETCS driver machine interface. Brussels: European Railway Agency; 2009.

[29] UNISIG. SUBSET-026, System requirements Specification - Baseline 3. Brussels: UNISIG; 2010.