On repeatable measurements when testing dependable systems: a discussion and experiences

Abstract
The scientific literature and the industrial practice agree since several years on the fundamental role of experimental evaluation (testing) in the assessment of the dependability attributes in critical systems. This paper analyzes the key issue of achieving repeatable measurements in such systems, which is a major milestone albeit often difficult to achieve if not to approach appropriately. To improve the repeatability of the experiments and consequently improve our confidence in the results, this paper presents the support of metrological assessment of instruments and results when testing critical systems. The paper identifies the current status of the research and existing gaps, and presents three case studies developed by the authors where measurement theory is applied to aim to trustworthy, repeatable experiments.

Keywords
Dependability, testing, metrology, repeatability, GPS, SAFEDMI, ALARP, R&SAClock

1 Introduction

Experimental evaluation (*testing* [1]) is an attractive option for evaluating an existing system or prototype, because it allows to observe the real execution of the system to obtain (hopefully, highly accurate) measurements of the system in its usage environment [7], [29], [25].

A fundamental requirement at the basis of experimental evaluation is that the experiments are repeatable [2], [3], [7], such that the collected results can be claimed trustworthy and successfully compared [26], [28], [27]. To guarantee a high confidence in the results provided and in the repeatability of the experiments, it is necessary to perform an accurate design of the measuring system (the instruments and features used to perform the measurements), and appropriately characterize the target system. This implies that the measuring system and all factors that may influence the results of the experiments (e.g., the environment) need to be investigated and that possible sources of uncertainty or bias in the results need to be addressed.

This paper analyzes the state of the art on the *experimental evaluation of dependable systems* to identify the general approach towards metrological assessment of tools and results, especially focusing on repeatability (as defined by metrology). The analysis shows that while there is generally a widespread consciousness of the relevance of the topic, few solutions or approaches exist to address it systematically. Three case studies developed by the authors in the context of different research projects are shown, that describe possible approaches to achieve a metrological assessment of the system to judge on measurements quality and repeatability. The three case studies present the evaluation of different systems, that are (in ascending order of complexity) i) a middleware service to achieve resilient clock synchronization, ii) OTS (Off-the-Shelf) GPS (Global Positioning System) devices, iii) a safety-critical embedded system for railway train-borne equipment (specifically, a Driver Machine Interface).

The rest of the paper is organized as follows. Section 2 introduces basic notions on measurement theory. Section 3 investigates the application of principles of measurement theory in the experimental evaluation of dependable [1] systems. Section 4 introduces the case studies, identifying why they were specifically selected for this work, Section 5 to Section 7 present the three case studies, and Section 8 concludes the paper.

2 Measurement theory (metrology)

A few fundamental concepts related to characterize measurement systems and methods according to metrological criteria are introduced. A complete digest of metrological terms and concepts can be found in [2], [3], .

Measuring a quantity (namely the *measurand*) consists in quantitatively characterizing it. The procedure adopted to associate quantitative information to the measurand is called *measurement*. The *measurement result* is expressed in terms of a *measured quantity value* and a related (*measurement) uncertainty*.

Uncertainty provides quantitative information on the dispersion of the quantity values that could be reasonably attributed to the measurand. Uncertainty has to be included as part of the measurement result and represents an estimate of the degree of knowledge of the measurand. It is usually expressed in terms of a confidence interval, that is a range of values where the measurand value is likely to fall. The probability that the measurand value falls inside the confidence interval is named confidence level. Two different ways to compute the *standard uncertainty* (uncertainty expressed as a standard deviation) are described in [2] and are respectively called Type A and Type B uncertainty. *Type A* uncertainty is

computed statistically, as an estimate of the standard deviation of the mean of a set of independent observations. *Type B* uncertainty is computed on the basis of a scientific judgement using all the relevant information available, as previous measurement data and knowledge of the behavior of relevant materials and instruments. Type B uncertainty does not depend on the amount of observations collected and consequently it is especially relevant when the number of independent observations is low.

Resolution is the ability of a measuring system to resolve among different states of a measurand. It is the smallest variation of the measurand that can be appreciated, i.e., that determines a perceptible variation of the instrument's output.

It is well known that any measurement system perturbs the measurand, determining a modification of its value. Minimizing such perturbation, that is minimizing the system's *intrusiveness*, is therefore desirable when designing a measurement system.

Repeatability is the property of a measuring system to provide closely similar indications in the short period, for replicated measurements performed i) independently on the same measurand through the same measurement procedure, ii) by the same operator, and iii) in the same place and environmental conditions. In practice being able to achieve repeatability (and having trusted results which e.g., are not characterized by the same bias) requires to investigate on the intrusiveness and resolution of the measuring system, and on measurement uncertainty.

Finally, as measurement results are expressed in terms of ranges of values, intervals measured through different instruments ought to be compared rather than single values. Specifically, if results are expressed with the same confidence level, they are said to be *compatible* if the related intervals overlap.

3 Experiments repeatability in dependable systems

Issues with the way measurement is applied in assessing computer dependability, and the need for giving practice a better theoretical basis, were first raised with respect to software reliability assessment. Problems were identified separately in two communities of research and practice: software reliability [30] and software metrics [31]. There were three sets of inter-related issues: confusion about the meaning of a measure (leading for instance to redefining software "reliability" as a count of bugs in a piece of code, or to seeking scalar measures for inherently multi-dimensional attributes), confusion between problems of measurement and of prediction (leading for instance to naïve methods for inference from

observed failures to future reliability), and insufficient fitness for purpose of the metrics [32].

The objective of this Section is to discuss the vast topic of experimental evaluation of dependable systems, focusing on the assessment of instruments and results according to principles of measurement theory.

The works presented in [5], [6], [29] investigate measurement tools for the evaluation of critical systems in the literature at the light of metrology concepts and rules. To the best of our knowledge, no other similar surveys and investigations exist for dependable systems. We report here the main findings, with a particular focus on repeatability.

In [5], [6], [29] it is suggested that the exploitation of the principles from measurement theory could aid the assessment of results and of measuring systems, consequently increasing the trust in the results achieved. The investigations in [5], [6], [29] show that consciousness about the metrology properties is present, but the approaches are sometimes intuitive and in general non-univocal. It is to be clearly remarked that this *does not mean* that tools, or experiments, are badly designed, nor that results are not correct; but the framework offered by metrology could improve the evidence on the quality of the results achieved.

Amongst the measurement properties mentioned above, in particular, it was rarely identified a real effort to estimate *uncertainty* and to determine solid bounds on the reliability and trustability of the measures collected.

Repeatability is identified as the most critical issue to face, especially when performing time measurements in distributed systems, due to limits on collecting accurate time values (executions of the same run will probably not bring the same exact results [3]). Determinism of the target system is needed to ensure repeatability, including the starting state of the system: for example, in order to completely ensure repeatability of every experiment, a fault injection tool would have to copy the entire state of memory at start-up and restore it in each experiment [4]. Thus it is a matter of fact that repeatability is often *not achievable* when measurements are carried out on computer systems, and especially on highly distributed ones: the same environmental conditions can, in fact, hardly be guaranteed.

It has to be noted that repeatability is a fundamental property in testing dependable systems. In fact, in dependable systems, faulty behaviour may often be not acceptable, and testing is a mean to check the proper behaviour of the system. Thus, repeatability of the experiments - having

the same results for different repetition of an experiment - is easily acknowledgeable as fundamental.

Ultimately, common guidelines are extracted from the measurement standards [2] and [3], to improve confidence in measuring dependable systems:

- the measurand should be clearly and univocally defined;
- all sources of uncertainty should be singled out and evaluated;
- some attributes of major concern for dependability measurements, such as intrusiveness, resolution and repeatability should be evaluated;
- measurement uncertainty should be evaluated;
- comparison of measurement results provided by different tools/experiments should be made in terms of compatibility.

4 Introduction to the case studies

As an exhaustive presentation of the case studies is unfeasible due to space constraints, they are reported summarized in this paper and appropriate references to extended discussions are inserted in the text where appropriate. In this Section, we introduce the three case studies and we motivate their selection as examples of metrological assessment and data comparison. The description of the case studies mainly focuses on such arguments, while steps of the evaluation process as definition of workload, faultload, or presentation of the results are only sketched.

In the first case study, presented in Section 5, a resilient software clock [8], [24] is evaluated. The discussion is centered around the investigation of the measurement system and of the quality of results. Also, reuse of the measuring system is a relevant topics discussed. Additional information can be found in [11], [8], [24].

The second and third case studies were performed in cooperation with the company Ansaldo STS. Such case studies were developed in the context of the ALARP [16] and SAFEDMI [14] projects, respectively. The experiments were executed respectively at the Genova and Torino premises of Ansaldo STS, which made available i) the measuring instrument in the second case study, and ii) the prototype and part of the measuring instrument in the third case study. The time available for running the experiments was limited by the time we had access to the measuring instruments and the prototype. Consequently an attentive planning of the experimental campaign was fundamental together with the metrological assessment of instruments to reduce the risk of unexpected difficulties on-site and of performing useless testing sessions.

The second case study is devoted to the experimental evaluation of GPS devices to understand their measurement error. The analysis presented in Section 6 is intended to provide preliminary results and feedbacks to the designers of the system in which the GPS devices are used, which is presented in Section 6. An attentive investigation of sources of uncertainty was performed to support the evaluation of the GPS devices. Expanded description of this case study can be found in [12].

The third case study, shown in Section 7, is devoted to the experimental evaluation of a prototype of a safety-critical railway embedded system. The measuring instrument is attentively assessed, and results are matched with Type B uncertainty computed with the few measurements available (because of the limited access time to the prototype). We remark that amongst conclusive results of the analysis, an important recommendation was provided to the system designers to notify a potential flaw in the system (a duration slightly exceeding a timeout) and a related correction. Further information are in [10], [15].

5 Testing of a software middleware

We describe the experimental evaluation of a middleware component, namely the software clock Reliable and Self Aware Clock (R&SAClock, [8], [24]).

The R&SAClock is designed to be self-aware of its *synchonization uncertainty* [8], [24]: we define synchronization uncertainty as an adaptive and conservative evaluation of the distance of the local clock from the reference time (the *offset*). Such offset is usually computed by the synchronization mechanism in use without offering guarantees on the accuracy of such computation. The synchronization uncertainty computed by the R&SAClock provides such guarantees: when asked to provide the time at any time instant t, R&SAClock replies with an enriched time value [*likelyTime*, *minTime*, *maxTime*], where *likelyTime* is the time value computed reading the local clock c, and the interval [*minTime*; *maxTime*] offers information on the confidence that can be associated to the time value *likelyTime*. The computation of *minTime* and *maxTime* are based on the synchronization uncertainty computed by the internal mechanisms of R&SAClock [8].

It is required that the enriched time value includes the *true time* (i.e., the time provided by the *reference clock*): this means that given the enriched time value computed at any time instant t, such t must fall within the

interval [*minTime*; *maxTime*]. This is the motivation of the experimental evaluation here reported.

The experimental plan covers a relevant set of cases, including i) different values of the software clock parameters, ii) different types of workload, and iii) the possible occurrence of faults in the system under test and/or in the underlying synchronization mechanism (in general, we injected faults whose effect is a failure of the NTP synchronization mechanism in disciplining the software clock).

5.1 The target system and the experiments

The target system consists of an R&SAClock prototype, which is installed as a software component on a computer, that we call PC_R&SAC (Figure 1). The local software clock is synchronized through the Network Time Protocol (NTP). An NTP client (a process daemon) running on PC_R&SAC synchronizes the local clock using information from the NTP server(s). PC_R&SAC is connected to one or more NTP servers through the Internet. In the implementation considered here, the R&SAClock is a C++ middleware service which interacts with the software clock and NTP (see Figure 1).

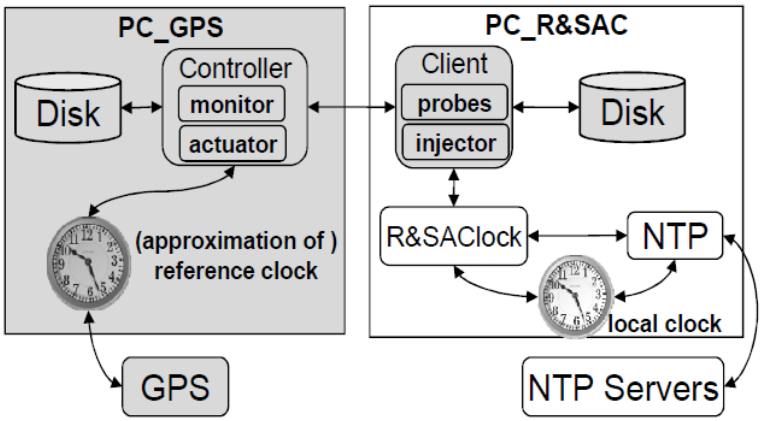

Figure 1. The measuring system developed and the target system [12].

5.2 The measuring instrument

The design and implementation of the validation testbed follow two basic rules: i) grant a time resolution sufficiently lower than that of the system under test, ii) keep the software probes as simple as possible in order to reduce the intrusiveness on the system under test, and consequently reduce uncertainty of the measurement results and improve repeatability.

The measuring system is shown in Figure 1. PC_GPS represents the monitoring system, where a Controller triggers the Client to request the

enriched time value to R&SAClock, and collects the returned values. The Controller can also access the reference clock (it is a clock synchronized using GPS).

The PC_R&SAC includes the target system, already described, and a software Client which can ask the enriched timestamp to the R&SAClock.

Both PC_R&SAC and PC_GPS log data relying on NetLogger [13], a tool for logging data in distributed systems guaranteeing negligible intrusiveness.

The choice of keeping the monitoring system and R&SAClock on different nodes is justified by the need of minimizing the intrusiveness of the monitoring system on the operative system of the node the R&SAClock is installed on. For the same reason, the option of having the reference clock as a second clock on the same node of R&SAClock is not considered. Our choice is to have a node (PC_GPS) including the reference clock and the monitoring system.

Controller and Client interact periodically to collect the enriched timestamps as follows. The Controller sends a message containing a *getTime* request and an identifier ID to the Client at time instant T_1 (T_1 is collected reading the PC_GPS clock) to ask the enriched time value. When the Client receives the message (at time instant T_2, collected reading the PC_R&SAC clock), it forwards such request to R&SAClock. When the Client receives the enriched time value from R&SAClock at time instant T_3 (again T_3 is collected reading the PC_R&SAC clock), it notifies the Controller. The Controller finally receives such notification at time instant T_4 (T_4 is collected reading the PC_GPS clock).

As said above, the objective of the validation is to check that given the enriched time value computed at any time instant *t*, such *t* must fall within the interval [*minTime*; *maxTime*]. As the R&SAClock provides a synchronization uncertainty sometimes lower than 1 millisecond, an accurate methodology is needed. Consequently, the validation test bed is based on a detailed analysis of which is the most suitable reference time instant to compare with R&SAClock output. The correct way to proceed is not to think at R&SAClock as a (software) device designed to answer the question "*what time is it*?", regardless of the practical use of the output it gives. Performing a meaningful validation of R&SAClock means verifying if R&SAClock works properly and to what extent it is useful. This means taking the time from the reference clock *when R&SAClock provides its answer*, rather than when the question is made. From the description above, this time instant corresponds to the reference time at

which T_3 is collected i.e., $t(T_3)$. Obviously T_3 can be measured only on PC_R&SAC, thus on the node without the reference clock: consequently $t(T_3)$ can not be measured directly. The following solution has consequently being identified to assess that $t(T_3)$ is within [*minTime*; *maxTime*].

The enriched time value should be compared to the reference time $t(T_3)$. For $t(T_3)$, the following relation holds (see also Figure 2):

$$t(T_3) \in (t(T_1) + \delta_1 + \Delta_{23} ; \; t(T_1) + \Delta_{14} - \delta_2 - \mu)$$

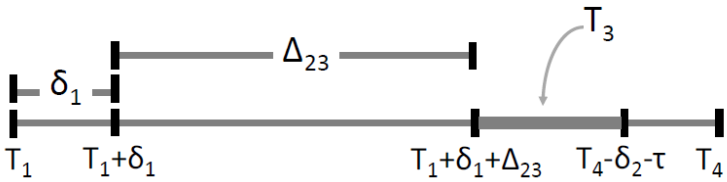

Figure 2. Time interval containing T3.

where $t(T_i)$ is true time when T_i happens; δ_1, δ_2 are the minimum transmission time (respectively, from PC_GPS to PC_R&SAC and vice-versa); μ is the time elapsed between $t(T_3)$ and the beginning of the transmission plus the time elapsed between the reception of the ack at PC_GPS and the actual timestamping instant; Δ_{xy} is the time interval $| t(T_y) - t(T_x) |$.

Given the reasonable hypothesis that $\delta_1 + \delta_2$ is much smaller than Δ_{23}, it is possible to reduce the uncertainty on $t(T_3)$ to a small interval. Thus, by comparing the [*minTime*, *maxTime*] with the interval $(T_1 + \delta_1 + \Delta_{23} ; T_4 - \delta_2 - \mu)$ shown in Figure 2, assuming that T_3 is the median of this interval, we can verify if the interval [*minTime*, *maxTime*] contains the true time.

The main contribution to the uncertainty on T_3 is given by the resolution, that is the amplitude of the interval where T_3 falls. In the experiments such interval has come out to be of the order of 100 microseconds.

A reasonable hypothesis underlying the above equation is true is that the delay between any T_i and the time its corresponding timestamp is taken, is the same for any *i*. Moreover, it should be noted that Δ_{23} and Δ_{14} are measured on different machines and, therefore, the interval in equation above could come out to be empty (due to severely different drifts). In such a case, the monitoring system can estimate $t(T_3)$ by subtracting δ_2 and μ from the time provided by its clock (i.e., the reference clock) at T_4, when it receives the ack; i.e., $t(T_3) = t(T_4) - \delta_2 - \mu$.

5.3 A glimpse on results

From the results shown in [11] and [8], the R&SAClock is able to compute an interval [*minTime*; *maxTime*] which includes the true time most of the time, with a coverage which depends on its configuration parameters (not debated in this paper). An exception is when the communication between the NTP client and its server(s) is of extremely poor quality; in that case the ability of R&SAClock decreases significantly.

6 Testing OTS GPS devices

The primary goal of the ALARP ATWS (Automatic Track-Warning System) [16] is to recall the attention of a working group operating on a railway worksite about the presence of an approaching train. ALARP keeps track of the status and position of the workers relying on *low-cost GPS-based wearable devices*, to identify the workers at risk (i.e., close to the track while a train is approaching) or to suggest escape routes. The localization requirements of ALARP demand that the railway worker is accurately localized for safety reasons.

The objective of the experimental evaluation here described is to assess if and to what extent cheap GPS devices can be successfully applied in the ALARP scenario. In particular, our work aimed to investigate the localization errors, with the goal of quantifying the contribution of systematic and random errors achieved when using low cost GPS receivers and providing feedbacks to the designer of the localization solution in ALARP. Although the sources of localization errors for GPS are well-known [17], the specificity and requirements of ALARP make relevant a detailed investigation of localization errors in this context.

6.1 Localization in ALARP

A typical railway worksite in which the workers need to be localized is an operation area of maximum 700 m length, in which workers typically move on foot (slow movement speed). The worksite can be located in place possibly surrounded by foliage, in canyon, or near buildings (i.e., there is limited visibility of GPS satellites). It is thus reasonable to expect errors due to satellite clocks (errors in the synchronization of the different satellite clocks, typically in the order of 0.8 m to 4 m) and ephemeral satellite orbits (errors in precisely establishing the spacecraft location, typically on the order of 0.8 m) when receivers use the same satellite set. Also errors are expected due to ionospheric and tropospheric signal perturbation and delays (given by the transition of the signal through the troposphere and ionosphere), and due to the receiver's design (errors due to the specific design of the receiver). All these enlisted errors can be

considered as systematic. At the same time, we can expect that errors due the receiver's thermal noise and external interferences exhibits negligible variations from a receiver to the others [17].

Conversely, we expect that multipath (reflection errors, one of the most significant and variable errors incurred in the receiver measurement process) affects randomly the localization error; this is expected as the major issue in localization measurements [17].

6.2 Target systems and measuring system

We selected for our tests two systems: the ND-100S produced by Globalsat [18] and the Garmin 18 LVC [19]. The Garmin 18 LVC is a GPS device of a higher category of price and performance than the Globalsat (it costs around three times the Globalsat ND-100S). Using two receivers of different quality allows to collect information on how the localization error varies depending on the device used and on the tradeoff in performance and costs.

As measuring system, a reference system allows to compare the data collected using the GPS devices previously mentioned. The reference system is the Trimble system [20], [21] composed of a stationary reference station Trimble R7 [20] and a roving device Trimble R8 [21]. This Trimble system is able to calculate the position with accuracy in the order of few centimeters (it is by far more accurate than the other two GPS devices).

During the experiments, common laptops with OS Windows 7 are used to log the NMEA 0183 sentences (the protocol National Marine Electronics Association 0183 defines the information that GPS devices communicate) provided every second on USB ports by all devices involved (Globalsat ND-100S, Garmin 18 LVC, Trimble R7 and R8). Note that the NMEA sentences contain the time instant at which the sentence is generated, so they can be logged directly without the need to investigate possible delays or inaccuracies of the laptops in timestamping.

6.3 Experiments description

We previously discussed the error sources for a railway worksite; amongst those, some error sources are bounded to the characteristics of the environment in which the worksite is placed. Here, we focus on the case when devices are close to the side of a high building, and are consequently subject to interference, multipath and limited satellites visibility. Devices may have partial or no satellites visibility for a short period of time. Each time a receiver loses and re-acquires satellites visibility, it may need to execute a transient phase in which the computation of their position is

particularly unreliable. Additionally, the characteristic of the environment may increase multipath and interference errors.

The experimental campaign consists of experiments involving stationary and rovering measurement, at different distances from the side of the building, and in hybrid scenarios where the planned paths alternate proximity of building and empty areas.

The experiments were executed during two raining days; consequently, the unstable weather conditions compromised part of the tests plan. Also, we must note that weather condition can influence localization results and constitute an additional source of error, thus making the analysis of the systematic and random contributions to the errors in the collected measures more complex.

6.4 A glimpse on results

During motion measurements, encouraging results were given by the Garmin, whose estimated positions always fluctuate around the true values (the values computed by R7-R8). The behavior of the Globalsat was instead more unstable during motion measurements, showing irregular sampling period, where data are not continuously refreshed.

Relying on the results collected, it was identified that GPS alone is not sufficient for the purpose of ALARP. Thus GPS augmentation approaches were deemed necessary and developed in the context of the ALARP project, to complement GPS data. In particular, the selected approach combines the outputs of GPS with the outputs of electronic fences placed in proximity of the tracks [22].

7 Testing of a safe railway system

We briefly present the experimental evaluation via fault injection of a safe train-borne Driver Machine Interface (SAFEDMI, [14]), that we performed as part of the V&V activities planned in the context of the SAFEDMI project [14].

7.1 A brief description of SAFEDMI

The SAFEDMI project aimed to design and validate a safety-critical Driver Machine Interface, called Safe Driver Machine Interface (SAFEDMI), with no hardware redundancy and using as much as possible hardware and software OTS.

In railway train-borne equipment, the SAFEDMI acts like an OTS safety-critical bridge between the operator (the train driver) and the EVC (European Vital Computer: it supervises the train movement). SAFEDMI communicates with the EVC as a slave; it acquires and manipulates

driver's commands (using a keyboard) from the EVC and it transforms EVC commands in graphical and audible information (using an LCD screen and audio devices). The SAFEDMI itself is composed of two components: the Driver Machine Interface (DMI) and the Bridge Device (BD). The DMI is the core of the SAFEDMI: it manages the communication activities with the EVC, with the BD, and with the driver (through a LCD screen and a keyboard). The BD is a wireless access point that allows configuration and diagnostic activities. In this paper we will focus only on the DMI. SAFEDMI target the requirements of Safety Integrity Level 2 (SIL 2 - railway standards [9] propose both qualitative and quantitative classes for the safety of equipments, and SIL 2 quantitatively means that the Tolerable Hazard Rate per hour THR is required to be between $10^{-7} \leq \text{THR} \leq 10^{-6}$).

We consider in this work exclusively the Startup and Normal operational modes of the DMI only, that we have tested through software fault injection. In *Start-up mode* the initialization procedures and the thorough testing of all devices are performed. In *Normal mode* the DMI produces graphical and audio information to support train driving, as well as it acquires and processes driver's commands; periodic testing activities are performed and diagnostic functionalities are available. A *safe mode* is entered when a malfunctioning is detected.

7.2 The measuring system

The measuring system built for performing software fault injection and its interactions with the target system are shown in Figure 4 and explained in what follows.

We subdivide the measuring system in two functional blocks (the grey blocks of Figure 4). The first block, that is composed of the software components library, injector and workload generator, deals with the injections and the workload execution: its function is to execute the experiments. The functions of the second block are monitoring, data collection and analysis: this block monitors the target system, collects results and analyzes them.

The workload generator is the tool EVC Packet Generator, a simulator of the EVC that runs on a PC connected to the DMI; the library and injection tool are instead both located on the DMI. The library is the mean to inject the available faultload: it enlists the available faults as well as the methods to inject them in the DMI. The identified faults are implemented adding extra code in genuine DMI functions or developing additional functions that are not part of the DMI genuine software.

The number of instructions needed to inject a fault is always small, and these instructions are fast to execute: the perturbation they introduce on system scheduling and the impact on the overall computational load can be considered negligible. The injection tool allows to perform the run-time software injections in the DMI. It is a cyclic, light and low-priority thread active on the DMI. This thread executes cyclically once every 1000 ms with a deadline of 2000 ms. The injection tool reads from a configuration file the instructions about the experiment to execute (e.g., the fault to inject and the time instant at which it should be injected), and uses the library to select the faults. The injection tool can inject a single fault or a sequence of faults at specific time intervals one from the others.

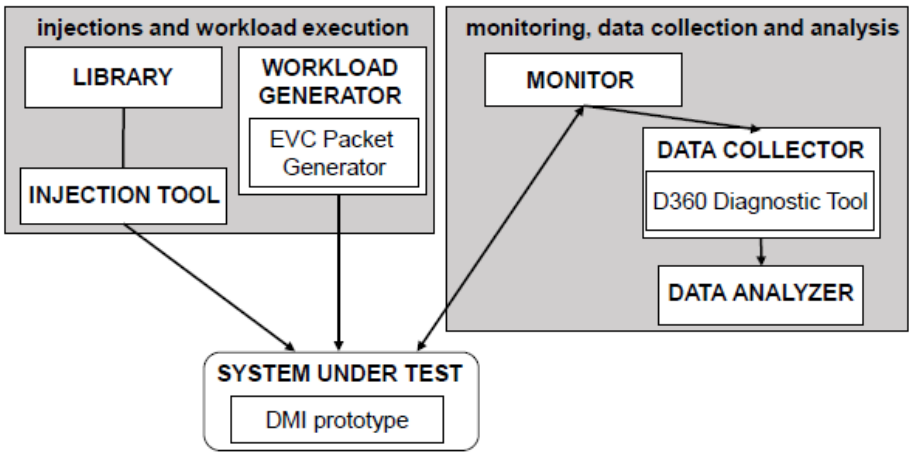

Figure 4. The measuring system developed [32].

Regarding the components of the second functional block, the data collector (or logger) is a diagnostic tool (called D360) located on the PC connected to the DMI. It receives, logs and organizes information received from the monitor, which executes on the DMI to timestamp events and to communicate events and related timestamps to the data collector. The data collector and the monitor communicate using a dedicated serial channel, different from the serial channel for the communication between the EVC Packet Generator and the DMI.

The monitor is an extension of the DMI log manager thread, that is a DMI genuine thread used for diagnostic activities (so we do not introduce a new thread in the system). The log manager thread is the thread with the lowest priority in the DMI, and it has no deadlines: it executes only when other threads are not running. As a consequence, to provide precise

timestamping of events it is necessary to collect each time instant (by invoking the DMI system call *getTime*) as an atomic action with the raising of the event.

7.3 Metrology assessment

We comment on the resolution, intrusiveness, repeatability and uncertainty.

The *resolution* of the measuring system is investigated only for time-related quantities. System resolution for time instants is 2 ms; it is the resolution of the DMI timer used as the base for the activities of the scheduler and of all threads.

Three components of the measuring system may be *intrusive* and perturb the DMI: the library, the injection tool and the monitor. To investigate intrusiveness we need to analyze perturbations in time and memory. Memory perturbation is negligible, since the executable files, the dedicated variables and the dedicated memory areas of library, injection tool and monitor are very small compared to the DMI memory.

Time perturbation needs a deeper investigation. The injections are performed through few, quick instructions that are executed at worst in few microseconds. The injection tool and the monitor are low priority threads that execute mainly when other threads are not running, to be as low intrusive as possible. The monitor sends data to the data collector using a completely dedicated communication channel: thus this communication does not alter the communication between the DMI and the EVC Packet Generator.

To further analyze intrusiveness, a schedulability analysis of DMI threads has been performed using the SchedAnalyzer tool (it provides a pessimistic estimation of the CPU computational load of the overall set of threads on the CPU): it resulted that the set of threads is schedulable (threads deadlines are guaranteed to be met, and there is enough CPU free time to guarantee that the injection tool and monitor threads will execute without influencing other threads execution). Thus, considering that resolution is 2 ms, we can state that intrusiveness is negligible.

Repeatability instead can be guaranteed only in specific cases, for the following limitations. First, there are no guarantees that the injections are performed exactly at due time instants due to the low priority of the injector thread. The EVC Packet Generator severely affects repeatability: despite it is supposed to generate always the same workload with the same exact timing, such exact timing is not guaranteed because of the non-real time OS (Microsoft Windows) in use. Running our experiments, we noted

that experiments performed while the DMI is in start-up mode were highly repeatable, showing compatible measurement results in the various runs of the same experiment [15]. This is mainly due to the fact that in start-up mode the EVC Packet Generator is not active, thus not introducing variability. As can be expected, results compatibility is not identified when the DMI is in normal mode.

Due to time constraints, we have a limited number of observations: consequently we compute a *Type B uncertainty* through an investigation of the system behavior instead of a Type A uncertainty computed through standard deviation. Type B uncertainty is estimated for time-related measurements as follows. When an event is raised, the *getTime* system call is invoked as an atomic action with the event: the contribution to uncertainty of this block of instructions is orders of magnitude smaller than 2 ms (it is at worst microseconds). For each event recorded, the resolution of the target system (2 ms) is the most significant contribution to uncertainty, while other contributions to uncertainty could be considered negligible. According to [2], in such situations the true value is expected equally distributed in an interval given by the measured value and the measured value plus the resolution (e.g., if 10 ms is the measured value and the resolution is 2 ms, the true value is expected within the interval [10; 12] ms). The expected true value should be set as the midpoint of the identified interval with an uncertainty of at most half the interval (e.g., if the interval is [10; 12], the expected true value is 11±1 ms and confidence 1).

However, our purpose is to estimate the safety of a critical system: we preferred to differentiate from the approach proposed in [2] and to report an uncertainty that is conservative, meaning that it must never err on the side of being too small. Consequently, for each event, we pessimistically consider that the corresponding time instant is collected with uncertainty of ±2 ms; thus, the uncertainty of time intervals is set to ±4 ms.

7.4 A glimpse on results

For all the injections performed, the safety mechanisms of the SAFEDMI were able to detect the error and correctly activate the safe state. Only in one case a slight violation of the requirement was observed, in which the reaction time (time to transit to safe state) was slightly exceeding the maximum allowed time of 100 milliseconds. In such case, it was suggested to system designers to shorten the period of the thread devoted to manage transition to safe state [15].

8 Lesson Learnt and Conclusions

This paper discussed the possibility of improving the quality and confidence in the experimental results by applying principles of measurement theory, and especially our work discussed one of the metrological properties that are most difficult to achieve in computer systems, namely *repeatability*.

The ultimate objective of this work is to show possible approaches to design experiments with repeatability in mind. In fact, the paper presented three case studies developed by the authors that include approaches to perform trustworthy, repeatable experiments. The three case studies are (in ascending order of complexity) i) a middleware service to achieve resilient clock synchronization, ii) OTS (Off-the-Shelf) GPS (Global Positioning System) devices, iii) a safety-critical embedded system for railway train-borne equipment (a Driver Machine Interface).

9 ACKNOWLEDGMENTS

10 REFERENCES

[1] Avizienis, A., Laprie, J.C., Randell, B., and Landwehr, C. 2004. Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. on Dependable and Secure Computing, 1(1):11–33.

[2] Joint Committee for Guides in Metrology (JCGM). 2008. Evaluation of measurement data - Guide to the expression of uncertainty in measurement.

[3] Joint Committee for Guides in Metrology (JCGM). 2008. ISO international vocabulary of basic and general terms in metrology (VIM), third edition.

[4] Skarin, D., Barbosa, R., and Karlsson, J. 2010. Comparing and validating measurements of dependability attributes. In EDCC-8, pp. 3–12, Washington, DC, USA. IEEE.

[5] Bondavalli, A., Ceccarelli, A., Falai, L., and Vadursi, M. 2010. A new approach and a related tool for dependability measurements on distributed systems. IEEE Trans. on Instrumentation and Measurement, 59(4):820–831.

[6] Bondavalli, A., Ceccarelli, A., Falai, L., and Vadursi, M. 2007. Foundations of measurement theory applied to the evaluation of dependability attributes. In Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and

Networks (DSN ’07), pages 522–533, Washington, DC, USA. IEEE Computer Society.

[7] Montgomery, D. C. 2008. Design and analysis of experiments. John Wiley & Sons.

[8] Bondavalli, A., Brancati, F., Ceccarelli, A., Falai, L., and Vadursi, M. 2013. Resilient estimation of synchronisation uncertainty through software clocks. *Int. J. Crit. Comput.-Based Syst.* 4, 4 (February 2013), 301-322.

[9] CENELEC. 2000. EN 50126 - Railway applications - the specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS).

[10] Bondavalli, A., et al. 2009. Design and evaluation of a safe Driver Machine Interface. IJPE, 4(2):153–166.

[11] Bondavalli, A., Brancati, F., Ceccarelli, A., and Vadursi, M.. 2010. Experimental validation of a synchronization uncertainty-aware software clock. In Proceedings of the IEEE Symposium on Reliable Distributed Systems (SRDS ’10), pages 245–254.

[12] Bondavalli, A., Ceccarelli, A., Gogaj, F., Vadursi, M., and Seminatore, A. 2013. Experimental assessment of low-cost GPS-based localization in railway worksite-like scenarios. Measurements, Elsevier, pages 456-466.

[13] Gunter D., and Tierney, B. 2003. Netlogger: a toolkit for distributed system performance tuning and debugging. In Proceedings of the IFIP/IEEE 8th International Symposium on Integrated Network Management, pages 97–100.

[14] SAFEDMI - Safe Driver Machine Interface for ERTMS automatic train control - Project Contract FP6-IST-2006-031413.

[15] Ceccarelli, A., Bondavalli, A., and Iovino, D. 2009. Trustworthy evaluation of a safe driver machine interface through software-implemented fault injection. In PRDC ’09, pp. 234–241.

[16] ALARP - A railway automatic track warning system based on distributed personal mobile terminals - FP7-SST-2010-234088.

[17] Kaplan, E. D., and Hegarty, C. J. (Eds.). 2006. Understanding GPS: principles and applications. Artech House, Boston, MA, second edition.

[18] GLOBALSAT ND 100s, data sheet, http://www.usglobalsat.com.

[19] Garmin 18 LVC, data sheet, http://www.garmin.com.

[20] Trimble R7 GNSS Base Station, data sheet, http://www.trimble.com/.

[21] Trimble R8 Rover, data sheet, http://www.trimble.com.

[22] Figueiras, J., Grønbæk, J., Ceccarelli, A., and Schwefel, H.-P. 2012. GPS and Electronic Fence Data Fusion for Positioning within Railway Worksite Scenarios. In HASE 2012, pp. 17-23, 2012.

[23] Montecchi, L., Nostro, N., Ceccarelli, A., Vella, G., Caruso, A., Bondavalli, A. 2015. Model-based evaluation of scalability and security tradeoffs: A case study on a multi-service platform. Electronic Notes in Theoretical Computer Science, 310, pp. 113-133.

[24] Bondavalli, A., Ceccarelli, A., Falai, L., Assuring resilient time synchronization. 2008. Proceedings of the IEEE Symposium on Reliable Distributed Systems, art. no. 4690795, pp. 3-12.

[25] Duchi, F., Antunes, N., Ceccarelli, A., Vella, G., Rossi, F., Bondavalli, A.. 2014. Cost-effective testing for critical off-the-shelf services. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 8696 LNCS, pp. 231-242.

[26] Durães, J., Fonseca, J., Madeira, H., Vieira, M. 2012. Field Studies on Resilience: Measurements and Repositories. Resilience Assessment and Evaluation of Computing Systems. Pages 213-237.

[27] Angrisani, L., D'Antonio, S., Esposito, M., Vadursi, M. 2006. Techniques for available bandwidth measurement in IP networks: a performance comparison. Computer Networks, 22 febbraio 2006, vol. 50/3 pp 332-349, ISSN 1389-1286, Elsevier North-Holland, Inc., New York

[28] Cinque, M., Cotroneo, D., Russo, S. 2006. Collecting and Analyzing Failure Data of Bluetooth Personal Area Networks. International Conference on Dependable Systems and Networks (DSN '06), Philadelphia, PA, USA, Jun 25-28, pp. 313 - 322, IEEE Computer Society Press, Los Alamitos, CA (USA).

[29] Bondavalli, A., Ceccarelli, A., Falai, L., Vadursi, M. 2012. Foundations of Metrology in the Observation of Critical Systems.Resilience Assessment and Evaluation of Computing Systems. pp. 205-211

[30] Brocklehurst, S. and Littlewood, B. 1992. New Ways to get Accurate Reliability Measures," IEEE Software, vol. 9, no. 4, pp. 34-42.

[31] Fenton, N. 1994. Software measurement: a necessary scientific basis. IEEE Transactions on Software Engineering, vol.20, no.3, pp.199-206.

Fenton, N.E., and Pfleeger, S.L. 1997. Software Metrics - A rigorous and practical Approach. 2nd Edition, PWS Publishing Company.