Requirements for a Trace Property Language for Medical Devices

Yves Ledru, Yoann Blein, Lydie Du Bousquet, Roland Groz, Arnaud Clere, Fabrice Bertrand

Requirements for a trace property language for medical devices

Short Paper

Yves Ledru
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG
Grenoble, France
yves.ledru@imag.fr

Yoann Blein
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG
Grenoble, France
yoann.blein@imag.fr

Lydie du-Bousquet
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG
Grenoble, France
lydie.du-bousquet@imag.fr

Roland Groz
Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG
Grenoble, France
roland.groz@imag.fr

Arnaud Clère
MinMaxMedical
Grenoble, France
arnaud.clere@minmaxmedical.com

Fabrice Bertrand
BlueOrtho
Grenoble, France
fabrice.bertrand@blue-ortho.com

ABSTRACT

The verification of software intensive medical devices can largely benefit from the analysis of their execution traces. Trace points can easily be added to the software, and traces can be used at several stages of the development and maintenance process. In this paper we focus on the TKA system and identify 15 representative properties that should be fulfilled by its traces. We also identify several stages in the product lifecycle where these properties should be evaluated. These properties put requirements on what should be expressible in a trace property language for medical devices.

CCS CONCEPTS

• **Software and its engineering → Specification languages**; *Formal software verification*;

KEYWORDS

Monitoring, Parametric Events, Temporal Specification, Trace analysis, Medical devices

ACM Reference Format:
Yves Ledru, Yoann Blein, Lydie du-Bousquet, Roland Groz, Arnaud Clère, and Fabrice Bertrand. 2018. Requirements for a trace property language for medical devices: Short Paper. In *SEHS'18: SEHS'18:IEEE/ACM International Workshop on Software Engineering in Healthcare Systems , May 27, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, Article 4, 4 pages. https://doi.org/10.1145/3194696.3194699

1 INTRODUCTION

A new generation of software-intensive medical devices (MD) arises, which benefit from various sensors to help physicians perform difficult surgeries with increasing precision. The development of such systems must follow national rules, based on standards such as ISO IEC 62304 [1]. These standards don't prescribe the use of formal verification methods, as required for the railways or the air and space industries. To favour a transition towards more rigorous development processes, the formal methods community has advocated for the advent of "lightweight formal methods"[6]. The MODMED project, funded by the French National Research Agency (ANR), gathers a University team with a MD company, BlueOrtho (BO), and a company specialized in software for MD, MinMaxMedical. This project adopts the monitoring of traces of surgeries as a lightweight approach to the verification of their properties.

There are several reasons to adopt a verification approach based on traces. First, software is usually instrumented with trace points which are easily improved to record sensor data, interactions with the surgeon and the medical team, and important events in the execution of the software. Also, traces can be produced during the development, but also during exploitation of the MD in real surgeries. Exploitation traces play an important role in post-market surveillance of the product.

Producing such traces was standard practice for BO, before the MODMED project started. But effective exploitation of the traces requires adequate tools to monitor properties of the traces. Therefore, the main goal of the MODMED project is to design a Trace Property language, named ParTraP (Parametric Trace Property language), to express properties of medical devices.

In this paper, we report on our process to identify requirements for a trace property language adapted to the TKA product, and explain how trace analyses can be used by BO at several stages of software development. Although our study is based on a single medical device, we rely on the experience of our industrial partners to check that the requirements for a trace property language apply to a larger range of medical devices.

Section 2 presents the TKA Medical device. Section 3 identifies 15 representative properties for MD traces. Section 4 extracts their main characteristics. Section 5 provides classification of properties based on their mandatory character. Section 6 studies when these properties should be checked in the development and maintenance lifecycle. Finally section 7 draws the conclusions and perspectives of this study.

SEHS'18, May 27, 2018, Gothenburg, Sweden

[1] https://www.iso.org/standard/38421.html

2 TKA

TKA is a system to guide Total Knee Arthroplasty surgeries, i.e. the replacement of both tibial and femoral cartilages with implants. TKA is composed of software, mechanical and electronical components. TKA supports the following sequence of operations:

(1) Digitization (acquiring a digital model of patient's anatomy) using sensors (trackers and pointer)
(2) Decision (adjusting the preoperative planning of implant sizes and position, according to the surgeon preferences)
(3) Action (positioning tracked cutting guides to cut bones)

The sensors include a set of trackers, firmly attached to the bones of the patient, whose position and orientation can be detected by a 3D camera. The camera also detects a pointer device, which is used to acquire the position of some anatomical points. The pointer can be used to acquire a single point, or it can be tracked to acquire several points (i.e. a cloud of points). The surgeon interacts with the product through a touch screen, and using the pointer device. When the system has acquired all anatomic information, the surgeon can plan several cuts in order to place the prosthesis. The system will then help them to precisely position cutting guides.

Every sensor acquisition or interaction with the surgeon is logged by TKA into a large trace (3000 events on average). This trace is initially designed to store sufficient information to understand what went on during a surgery. After each surgery, BO collects the corresponding trace. TKA has been used worldwide for a couple of years and 10 000 surgery traces have been collected by BO.

3 REPRESENTATIVE PROPERTIES

Currently, traces are analysed manually, but the large number of traces requires automating these analyses. In order to identify the kind of properties to check on these traces, we dedicated significant time to a detailed understanding of the traces. Two traces of actual surgeries selected by BO as representative were reviewed in detail, and a few others were provided to illustrate particular requirements. The members of the project also got access to confidential documents: the "FUNctional SPecification" (FUNCSP) document of TKA, which describes the specifications of the whole system, and the "TEChnical SPecification" (TECSP) focusing on the TKA software.

As a result, a list of 43 requirements in 11 functions described in TECSP was determined as relevant based on the interest expressed by the manufacturer. From this list, 15 properties were deemed "representative", based on the temporal relationships featured between events, the type of event data, and the operations performed on event data. They are now listed in arbitrary order:

P1. The trace contains a step "redo acquisitions". This step allows the surgeon to correct their previous acquisition. It is not part of the standard procedure flow and, therefore, interesting to detect.

P2. The temperature of the camera stays within a given interval. If used in proper conditions, the camera temperature should not deviate from the range where its precision is guaranteed.

P3. The distance between pairs of hip centres is less than d. This property asserts that the algorithm computing the hip centre is stable, i.e. gives similar results for consecutive acquisitions.

P4. The distance between the hip centre and the knee centre is greater than d. A violation of this property could reveal an abnormal positioning of the patient or the sensors.

P5. If the medial malleolus is farther from the camera than the lateral one, a warning is issued. A violation of this property may reveal that the 3D camera was installed on the wrong side of the patient.

P6. The user never skips a screen. The surgeon is expected to spend sufficient time to appreciate the information showed on the display before going to the next screen.

P7. The acquisition of a point succeeds if and only if the probe is stable. If the surgeon moves the probe tip during an acquisition, it should not be accepted.

P8. The protocol "redo acquisitions" proposes only already performed acquisitions. The system should not offer the user to redo acquisitions that were never performed.

P9. Detecting a new tracker produces a dialog asking for replacement confirmation.

P10. The state TrackersConnection is unreachable until the camera is connected. The system should not reach a state dependent on the camera until the camera is connected.

P11. A replaced tracker is not used until it is registered again.

P12. The action "previous" cancels the current point cloud acquisition. Acquiring a cloud of points takes a few seconds and can be cancelled. In this case, the current acquisition should not succeed.

P13. All the necessary trackers are seen before entering the state TrackersVisibCheck. To proceed, the system requires a set of trackers depending on the profile in use. All these trackers should be seen at least once before entering the state TrackersVisibCheck.

P14. On the trackers connection screen, a tracker is shown if and only if it is necessary. Only required trackers are shown to the user.

P15. In the state TrackersConnection, not detecting any new tracker for 2 minutes produces an error message.

4 MAIN CHARACTERISTICS

We identified several characteristics of these properties listed here.

- Does it involve events of different types?
- Is it parametric, i.e. does it involve event parameters?
- Is it temporal, i.e. does it constrain the order of two or more occurrences of events,
- Does it apply to a restricted scope of the trace?
- Has it geometric predicates on data?
- Has it GUI predicates on screenshots?
- Does it involve physical-time?
- Is it a required, assumed or usage property (see section 5).

Table 1 synthesizes the classification of the 15 properties presented in the previous section. In the following, we will detail this classification for a couple of properties.

Let us consider property P3, stating that the distance between pairs of hip centres is inferior to a given threshold. It involves several events of a single type, reflecting the acquisition of a new hip centre and that is parametrized with the acquired point. These events parameters must satisfy a geometric constraint. A violation of this property could indicate that the patient was not installed as expected (e.g. the surgeon should not use a "leg holder" which forces the leg position) or, if the patient was correctly installed, that the algorithm computing the hip centre is not stable.

Property P12 states that action "previous" cancels the ongoing points cloud acquisition. I.e. triggering action "previous" during an

Property	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
No. of event types	1	1	1	2	3	2	2	2	2	2	3	3	4	2	4
Parametric		X	X	X	X	X	X	X	X		X		X	X	X
Temporal						X	X	X	X	X	X	X	X		X
Restricted scope								X	X	X	X		X		X
Geometric predicate			X	X	X	X									
GUI predicate														X	
Physical time						X									X
Required			X		X		X	X	X	X	X	X	X	X	X
Assumed		X	X	X		X									
Usage	X					X									

Table 1: Classification of the 15 properties

acquisition prevents the success of this acquisition. This property involves three different event types, reflecting the action "previous", the beginning of an acquisition, and the success of an acquisition. No event parameter is needed, which makes the property nonparametric. However, it is temporal since the occurrence of the "acquisition success" event is constrained by other event occurrences.

The results of this classification are in accordance with our expectations: the properties are very diverse and rely heavily on data parameters. On the contrary, the physical time is rarely involved in the identified properties despite our expectations.

Based on these properties, we have designed the ParTraP language [3], which is mostly influenced by the specification patterns proposed by Dwyer et al [4]. Although numerous languages exist for runtime verification, e.g. [1, 2, 5, 7], none of them offers the combination of features required by our application context: support of parametric events, capability to express temporal properties and ease of use resulting from a verbose language. The interested reader may refer to [3] for a detailed description of the language and a comparison with related work.

5 CLASSIFICATION OF PROPERTIES

The properties listed in Section 3 belong to three categories: Required, Assumed and Usage properties. Table 1 records the corresponding category for each property.

Required properties. Such properties must be ensured by the TKA system. These properties correspond to requirements on the system. For example, this is the case of property P7 *"The acquisition of a point succeeds if and only if the probe is stable"* which requires TKA to check probe stability before validating the acquisition of a point. 11 out of the 15 properties correspond to this kind of properties.

Checking these properties on the traces should always succeed, otherwise it would reveal a failure of the software or of its associated hardware. Ideally, these properties should be formally proven. This is why we refer to these as "required" properties. In the MD industry, a more pragmatic approach is to provide verified traces as evidences that they are fulfilled by the product, and not full proofs.

Assumed properties. These properties should be ensured by the environment of the TKA system, i.e. the medical team and the surgery room. They appear as assumptions on the behaviour of this environment. If the environment fails to fulfill these properties, the behaviour of the TKA system may be affected. For example, this is the case of property P2 *"The temperature of the camera stays within a given interval"*. The camera temperature is influenced by the surgery room temperature. If the temperature is outside this range, the precision of the camera may be affected.

If the environment does not behave as expected, TKA is designed to stop assisting the surgery in the worst case. In any case, it ensures that the information displayed to the surgical team is correct, or stops displaying information if its correctness is not guaranteed.

Checking these properties on the traces is expected to succeed because the surgeon and their team are expected to use the system in the prescribed conditions. If one of these properties is not satisfied, it may be an explanation for difficulties arising during the surgery and it does not necessarily reveal a defect of the system.

These assumptions on the behaviour of the environment will be referred to as "Assumed" properties.

Please note that property P3 is marked as both required and assumed. P3 states that *"The distance between pairs of hip centres is less than d"*. It can result from the use of a leg holder, which violates assumptions on the environment, or from a wrong calculation which reveals a software failure. Since the trace does not record that a leg holder was used, a violation of this property leaves two possible causes.

Usage properties. TKA can be used in several different ways, depending on the surgeon's choices. Properties can be checked to understand how the system was used.

For example, this is the case of property P1 *"The trace contains a step 'redo acquisitions' "*. The 'redo acquisition' step was triggered by the surgeon and may reveal that it is difficult to have all acquisitions right at the first attempt, or that the surgeon is not trained enough to use the system. Checking these properties helps understand the way the TKA system is used, but does not reveal a particular failure of the system or its environment. It can be exploited to identify potential evolutions of the TKA system (e.g. efforts should be done to facilitate acquisitions). This is why we refer to such properties as "Usage" properties.

Other examples of usage properties include cases where TKA leaves a choice to the surgeon, e.g. several kinds of prostheses may be selected, or the order of several operations is not fixed. Usage properties can check whether a given choice was adopted. Statistics can be computed on the number of traces satisfying a given usage property. At longer term, "quantitative usage properties" might be

considered, reporting quantities instead of Boolean values like a number of occurrences of an event or the duration of a step or the distance between anatomic points.

6 WHEN SHOULD THESE PROPERTIES BE CHECKED?

Traces can be produced in four contexts during the MD lifecycle: development, qualification, manufacturing and exploitation.

The development context. It corresponds to all activities which will create or modify the TKA system. They correspond to the initial development, but also to corrections during the maintenance phase and evolutions of the system. In the development context, traces will be produced during tests. They will be used to evaluate the correctness of the system and hence required properties will be the most important ones at this stage.

The qualification context. It comes after development activities, it is aimed to demonstrate the correctness of the system and validate assumptions on its environment. Here again the focus will be on required properties, but in the qualification context, these properties are expected to be fulfilled by the system under qualification. The qualification phase also involves "Acceptance tests" typically performed by surgeons on corpses using a successfully verified MD. During these acceptance tests, one can expect that required properties will be satisfied but the focus will be on assumed and usage properties to check that the environment behaves as foreseen.

Manufacturing. Since TKA is composed of software and hardware, it involves a manufacturing phase where the system is assembled and tested. Here the tests check the hardware for manufacturing defects. Properties checked at this stage are required properties because they aim to check the system, and not its environment. Still, while in the development and qualification contexts failure could result from software defects, in the manufacturing phase, the software should be correct and failures only reveal hardware defects, or incorrect execution of the tests by the tester.

The exploitation context. corresponds to the operation of a qualified system during a real surgery. In this context, required properties should not fail. Checking these properties on traces of numerous surgeries brings additional evidence of the quality of the system.

Assumed and usage properties are mostly relevant to the exploitation context. Assumed properties correspond to assumptions on the way the system is operated. Checking the assumed properties on the traces of real surgeries will help detect cases where the environment of the system was not adequate. Detecting such traces may bring explanations on why something did not proceed smoothly, or require for more robustness of the system against the failure of these properties.

This exploitation context corresponds to the activities of post-market surveillance, and misuse surveillance. Once the MD is certified for some market and actually used, BO still performs trace analyses that can be critical to ensure patient safety. But the large number of traces (10 000) requires that automated analyses, motivating the need for a trace property language and its toolset.

Misuse is also an important problem in the exploitation context. As a general usability principle, BO chooses to only block the user when the action would undoubtedly have direct consequences for the patient. Consequently, it is important to study how TKA is used in the market to detect misuses. This allows warning users about potential problems and advising them on how to avoid these problems in next surgeries. On the other hand it may denote usability problems that should be tackled by BO. In any case, BO feels this is a very relevant activity to improve TKA safety and effectiveness.

A typical example is to verify that TKA is used within intended operating temperature range because it affects the camera accuracy. TKA itself checks this prerequisite environment condition and the user is warned about possible accuracy problems but the user is left responsible for using it or waiting for camera warmup.

7 CONCLUSION

The verification of software intensive medical devices can largely benefit from the analysis of traces produced during their execution. Trace points can easily be added to the software, and traces can be used at several stages of the development and maintenance process. During post-market surveillance, they bring empirical evidence that the system exhibits the expected behaviour, and they allow to identify how the system is actually used.

Trace analysis should be automated. In this paper, we have identified a set of properties that should be expressible in a trace property language. These properties result from a thorough study of TKA, its traces, and specification documents. We identified 43 properties, and 15 of them were selected as representative. This paper lists these 15 properties, and performs their classification. This classification identifies expected characteristics of the property language. They constitute requirements for the ParTraP language [3], but can also be used to evaluate the adequacy of existing languages to express such properties. Although this set of properties result from the analysis of a single MD, the experience of our industrial partners with the development of other MD leads them to believe that these properties are representative of a larger set of MD.

ACKNOWLEDGMENTS

This work is funded by the ANR MODMED project (ANR-15-CE25-0010).

REFERENCES

[1] Howard Barringer, Allen Goldberg, Klaus Havelund, and Koushik Sen. 2004. Rule-Based Runtime Verification. In *VMCAI 2004 (LNCS)*, Vol. 2937. Springer.
[2] Andreas Bauer, Martin Leucker, and Jonathan Streit. 2006. SALT - Structured Assertion Language for Temporal Logic. In *ICFEM 2006 (LNCS)*, Vol. 4260. Springer.
[3] Yoann Blein, Yves Ledru, Lydie du Bousquet, and Roland Groz. 2018. Extending Specification Patterns for Verification of Parametric Traces. In *FormaliSE 2018*. IEEE Computer Society.
[4] Matthew B. Dwyer, George S. Avrunin, and James C. Corbett. 1999. Patterns in Property Specifications for Finite-State Verification. In *ICSE' 99*. ACM.
[5] Sylvain Hallé and Roger Villemaire. 2008. Runtime Monitoring of Message-Based Workflows with Data. In *ECOC 2008*. IEEE Computer Society.
[6] Cliff B. Jones. 1996. Formal Methods Light. *ACM Comput. Surv.* 28, 4 (1996), 121.
[7] Rachel L. Smith, George S. Avrunin, Lori A. Clarke, and Leon J. Osterweil. 2002. PROPEL: an approach supporting property elucidation. In *ICSE 2002*. ACM.