Research project

Andrea Bombarda

June 10, 2019

The research project submitted in view of admission to Phd course in **Engineering and applied sciences** (XXXV Session) should include all the information listed below. The maximum length is given in characters with spaces and has a purely indicative value.

Candidate's forename and surname: Andrea BOMBARDA

Degree option chosen: PhD course in Engineering and applied sciences - Positions covered by scholarships financed by the University

Title of research project: Quality of medical software and systems

Abstract

Software plays a role of increasing importance in the medical and health fields since it has a critical function in many healthcare processes: from the software included in embedded medical systems to the applications installed on the PCs of the doctors, from the mobile applications on smartphones to the protocols used to make the communication between medical devices possible. Health depends on the correct functioning of this software, and their malfunctioning or fail can cause serious damage. For this reason, in recent years there has been a growing attention to validation, verification and testing of medical software. Recent software and medical devices also include many components that use artificial intelligence (AI) to process data and produce results. Also this aspect has to be considered when one wants to verify a medical software, so it is crucial to use methods able to deal with AI for its validation.

The general objectives of this research project are those related to the use of rigorous methods to test and develop medical software, ensuring the quality of the products and to comply to the main regulation in developing medical systems [8, 9]. In particular, the project will be mainly focused on the activities of validation and verification (V&V) that are challenging when executed over a medical devices.

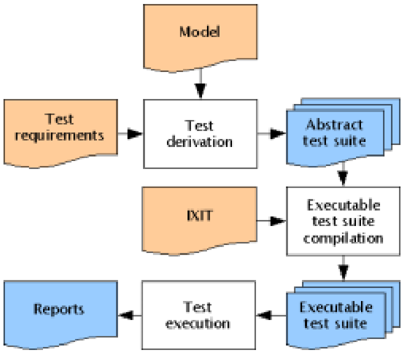

Figure 1: An example of the MBT workflow

State of the art in the field

The main regulations concerning the development of medical systems classify as medical devices the followings:

- Software embedded in medical hardware
- Accessories of any medical device
- Software that are executed on a PC during each medical activity

Historically, medical device software has typically been verified using code reviews, static analysis, and dynamic testing. However, these methods are not a good choice when the system to be tested is complex. That's why a different approach should be used.

Current regulations require that the software life cycle also includes activities of software V&V. The results of these activities, the used methods and all the information that can be useful to prove the correctness of the software have to be documented for the certification of these devices. The best way to maintain a complete traceability and obtain a good quality is the use of rigorous methods, since the demonstration of all the executed activities is straightforward.

As shown by [6], formal methods are increasingly used in the development of medical software and devices because they can lead to a better quality of the product, which is a must for the human safety. The most frequently used testing method for medical devices is the Model Based Testing (MBT), in which the tester builds a model, i.e. an abstract partial representation of the System Under Test (SUT) and of its behaviour (see Figure 1). From this model, one can derive several test cases that can be executed over the abstract model or over the real system. One of the best known MBT formal method used to develop medical devices is the one which uses Abstract State Machines [4], since it allows to design the system at an high level and then, after V&V activities, to automatically generate the code that can be embedded on the actual device.

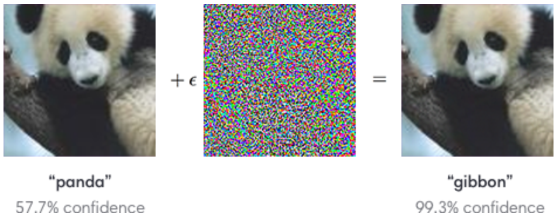

Figure 2: Example of adversarial example: an adversarial input, overlaid on a typical image, can cause a classifier to miscategorize a panda as a gibbon

Another similar approach is the one who exploits Event-B, which adopt a multi-formal development paradigm: the requirements are modelled by using UML-B and then the verification is made with theorems proving [1].

All these solutions require the creation of a model of the SUT. Sometimes the developer wants to be sure about the correctness of the written model, in terms of safety properties and conformance to a given specification. This activity is known as Formal Verification and can be performed, for instance, through Model Checking with the use of Linear Temporal Logic (LTL) or Computational Tree Logic (CTL). The use of property verification is quite common when working with medical software [3].

Regarding the use of artificial intelligence and machine learning in medical software, the most used tools are the Convolutional Neural Networks (CNNs) that allow the user to classify input images in different classes. They are a very effective tool but their main problem is that we cannot be sure about their performances (in terms of accuracy of the results) when an alteration of the input occurs. This can be a con of the use of ML because for a system, to be useful in solving medical diagnostic tasks, the following features are desired:

- Good performance
- Ability to appropriately deal with missing data and/or noisy data (errors in data)
- Transparency of diagnostic knowledge and ability to explain decisions

While the first feature is quite easy to obtain, the second and the third are difficult to prove. The robustness w.r.t. errors in data is not as easy to assure, because it is not possible to test the network with all the inputs of the domain. The transparency of diagnosis and ability to explain decisions can be hard to prove because the internal behaviour of a neural network can be not very clear and predictable.

Currently, the main results about the robustness of CNNs concern only the robustness w.r.t. adversarial examples [14] that exploit the internal structure of a network to modify the classification of an input image (see Figure 2).

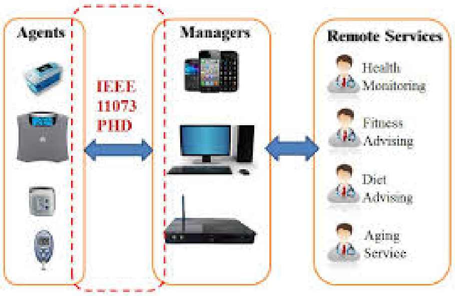

Figure 3: IEEE 11073 PHD's communication model between two entities: agent and manager

Aims of the project

The main goal of the project is to define and test different methodologies to help the developer to create or test a medical software or device, assuring a better quality and safeness of the final product.

There are many well-known techniques that are not used for medical system development. In this project these techniques will be tested to see whether they are suitable to comply the regulation in force and also if they can lead to a better product. For example, when executing testing activities which involves the generation of test sequences, one has to assure that each sequence is acceptable: in actual systems, it is possible that some event `A` can be fired only after an event `B` has been already fired. This is not considered by the techniques that are currently used in medical software development field, causing an overhead of negative testing and a lack of focus on positive testing, i.e. the kind of testing in which the input data assume valid values.

The same methodologies will be also applied on protocols that are used by medical systems to communicate with others, since it is really important not only the behaviour of the single device but also the behaviour of the whole system, when a communication between components occurs. This approach has been already applied on the IEEE 11073 protocol (Figure 3), and in particular on its C implementation called Antidote, in a previous paper (that is attached to this application), but still a lot of work can be conducted over this and other protocols.

In a world which is becoming more and more AI-Centric [7], also machine learning techniques have to be tested, in particular when they are used in medical devices. As previously stated in the "State of the art", the main references of robustness analysis of the neural networks are only versus the adversarial examples. However, adversarial examples are unlikely to occur when a medical software is used by a caregiver. Thus, one of the main aspects of this research project includes the analysis of the robustness of CNNs w.r.t. the most common input alterations, such as compression, noise addition, rotation, translation and image crop. Moreover, like the traditional testing, also in the artificial intelligence world, it is impossible to perform an exhaustive test on all the parts of the systems, because even in the most simple case the input domain is too big to test each possible input; in addition to it, in this particular field, there is not a solid state of the art in testing, but just few guidelines in aid to the tester. Considering that, in Neural Networks, is not possible the application of the classical white-box approach, the testing activity will be only performed with black-box approach, by applying known inputs and observing the results provided.

Many medical software use not only CNNs but also other types of Neural Networks to produce a diagnosis or to generate treatment rules [2, 15]. Testing activities will be also performed over these kinds of Neural Networks to define a standard framework for their validation, since they have to be safe as well as the CNNs.

Outline of the project (methodology, different stages, tools employed)

Methodology: The whole research project will be conducted by the study of several practical cases, exploiting each time a specific methodology to execute V&V activities. Moreover many technical and theoretical aspect will be analysed to find out whether is possible to apply novel methodologies in the field of study.

Literature study of the state of the art, and work in different areas that may be relevant to the aims of the project, will be the main methods used to progress with the research, as well as the attendance to relevant courses, workshops and conferences.

Tools employed: As state-of-the-art in the medical software development sector, MBT will be one of the main methodologies used to test the systems. Thanks to all the tools available in the `Asmeta` framework[1], V&V activities can be easily executed by using Abstract State Machines (ASMs) [11]. `Asmeta` framework can facilitate the entire life cycle of software development, i.e., from modelling to automatic code generation (Figure 4). Moreover, by working with Abstract State Machine will be possible to create test scenarios (for example with the `Avalla` language), which are an optimal choice for test execution in the medical field.

The confidence about the correctness of each produced model will be increased using property checking techniques, such as LTL and CTL formulas, with the NuSMV[2] model checker.

To manage the complexity of medical systems, even Combinatorial Interaction Testing (CIT) will be employed, especially 2-wise testing, since it allows not to test all the possible combinations of values that can be assigned to each input, only by testing all the possible discrete combinations for each pair of

[1] http://asmeta.sourceforge.net/

[2] http://nusmv.fbk.eu/

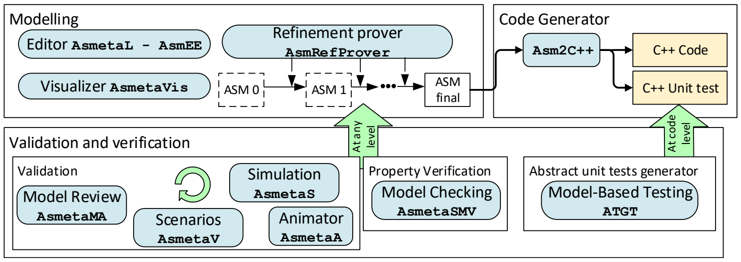

Figure 4: The ASM development process powered by the `Asmeta` framework

input parameters. To reduce even more the complexity, the generated combinations have to be eligible for the studied system. This can be reached by using automata to represent precedence constraints and obtain only test sequences that satisfy them. Three Java libraries that will be used for this purpose are: `Brics`[3], which is the best solution when working with regular expressions, `LTL 2 BA`[4], that can translate LTL properties into automata, and `AutomataLib`[5], which allows automaton learning [5]. CIT can be used not only for combinatorial tests for sequences of events, but also for fault discovery, which is equally important in the development of medical devices.

As state-of-the-art, Neural Networks will be developed using Python and its dedicated libraries, such as Keras[6] and TensorFlow[7]. Considering the unpredictability of the behaviour of Neural Networks, their testing will be mainly executed as black-box testing [13, 12]. An interesting approach to the formal verification of NNs, which will be investigated, is the one related to the use of SMT Solvers [10], with whom it is possible to check their safety.

Different stages: The envisioned different stages of the project are:

- First year:
  - Increase the knowledge in the field of research, of the tools and the methods to be used.
  - Choose one promising rigorous method for investigation on how it can be applied for the purposes of this project, and write at least one conference/journal article(s).
  - Build a tool able to test the robustness of various types of Neural Networks subjected to input unforeseen perturbations or to support the rigorous development of medical software.
- Second year:

[3] https://www.brics.dk/automaton/

[4] http://www.lsv.fr/~gastin/ltl2ba/

[5] https://learnlib.de/projects/automatalib/

[6] https://keras.io/

[7] https://www.tensorflow.org/

  - Improve the tool built during the first year of the project.
  - Choose some case studies to apply the methods developed.
  - Write paper(s) for related topics, such as about model based testing, rigorous methods and formal verification use in medical software.

- Third year:

  - Collect statistical and qualitative data about the usage of the tool(s) created. Review the results and improve the tools/algorithms by submitting them to users.
  - Choose another theme in the research field and write paper(s) related.
  - Prepare thesis and defense.

The final PhD thesis describing the results of this research project should contain:

1. Introduction and outline of the problem, with a particular focus on the motivations.

2. Literature review of subject area (rigorous methods, artificial intelligence techniques and model based testing application during the software life cycle of medical systems).

3. Methodological chapter(s).

4. Results chapters in which the main novelties of the proposed solutions will be analysed as well as their pros.

5. Conclusion and further work.

Results envisaged and how they will be assessed

The expected results are the following:

- Publication of articles about the application of rigorous methods in software development and medical systems.

- Analysis of several software development techniques, to find the best solution for medical software.

- At least one concrete tool that combine one/more of the envisioned methods

Any tools realized should be usable by users familiar with the notion of the research field, but without having a detailed knowledge of the underlying algorithms, i.e. the algorithms should always work as black-box mechanisms.

References

[1] Jean-Raymond Abrial. *Modeling in Event-B: System and Software Engineering*. Cambridge University Press, 2010.

[2] Qeethara Al-Shayea. Artificial neural networks in medical diagnosis. *Int J Comput Sci Issues*, 8:150–154, 02 2011.

[3] Paolo Arcaini, Silvia Bonfanti, Angelo Gargantini, Atif Mashkoor, and Elvinia Riccobene. Formal validation and verification of a medical software critical component. In *2015 ACM/IEEE International Conference on Formal Methods and Models for Codesign (MEMOCODE)*. IEEE, sep 2015.

[4] Paolo Arcaini, Silvia Bonfanti, Angelo Gargantini, Atif Mashkoor, and Elvinia Riccobene. Integrating formal methods into medical software development: The ASM approach. *Science of Computer Programming*, 158:148–167, jun 2018.

[5] Saugat Bhattacharyya, Abhronil Sengupta, Tathagatha Chakraborti, Amit Konar, and D. N. Tibarewala. Automatic feature selection of motor imagery eeg signals using differential evolution and learning automata. *Medical & Biological Engineering & Computing*, 52(2):131–139, Feb 2014.

[6] Silvia Bonfanti, Angelo Gargantini, and Atif Mashkoor. A systematic literature review of the use of formal methods in medical software systems. *Journal of Software: Evolution and Process*, 30(5):e1943, feb 2018.

[7] Kenneth R. Foster, Robert Koprowski, and Joseph D. Skufca. Machine learning, medical diagnosis, and biomedical engineering research - commentary. *BioMedical Engineering OnLine*, 13(1):94, Jul 2014.

[8] P. Jordan. Standard iec 62304 - medical device software - software lifecycle processes. In *2006 IET Seminar on Software for Medical devices*, pages 41–47, Nov 2006.

[9] A. Ohne Autor Fd. General Principles of Software Validation; Final Guidance for Industry and FDA Staff, Version 2.0. FDA document formal, January 2002.

[10] Luca Pulina and Armando Tacchella. Checking safety of neural networks with smt solvers: A comparative evaluation. In Roberto Pirrone and Filippo Sorbello, editors, *AI*IA 2011: Artificial Intelligence Around Man and Beyond*, pages 127–138, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[11] Egon Borger Robert F. Staerk. *Abstract State Machines*. Springer-Verlag GmbH, 2003.

[12] Xiaowu Sun, Haitham Khedr, and Yasser Shoukry. Formal verification of neural network controlled autonomous systems.

[13] Youcheng Sun, Xiaowei Huang, Daniel Kroening, James Sharp, Matthew Hill, and Rob Ashmore. Testing deep neural networks.

[14] Matej Uličný, Jens Lundström, and Stefan Byttner. Robustness of deep convolutional neural networks for image recognition. In *Intelligent Computing Systems*, pages 16–30. Springer International Publishing, 2016.

[15] Zhi-Hua Zhou and Yuan Jiang. Medical diagnosis with c4.5 rule preceded by artificial neural network ensemble. *IEEE Transactions on Information Technology in Biomedicine*, 7(1):37–42, mar 2003.