Journal of Mechatronics and Robotics

Original Research Paper

Testing the Interfaces for Mechatronic Systems

[1]Liviu Mihai Sima and [1,2,3]Miron Zapciu

[1]Department of Robots and Production Systems, University Politehnica of Bucharest, Bucharest, Romania
[2]Academy of Romanian Scientists, Bucharest, Romania
[3]Transport Research Institute Incertrans S.A., Bucharest, Romania

Article history
Received: 23-02-2022
Revised: 23-03-2022
Accepted: 28-03-2022

Corresponding Author:
Liviu Mihai Sima
Department of Robots and Production Systems, University Politehnica of Bucharest,
Bucharest, Romania
E-mail: liviusima23@gmail.com

Abstract: The evolution of mechatronic systems had a large impact in recent years. When comparing them in terms of their mechanical, electrical and electronic components with their older variants, responds to a more elevated level of commands and requirements. Their testing has become more complicated and hybrid methods are often used. Hardware and software interfaces are used for this software. Hardware interfaces use a predictive approach method, developed linearly in the form of a "cascade". The software approach includes system requirements, where using a hybrid combination of test methods is preferred. Testing of hardware and software interfaces must be as accurate as possible, as is choosing the right models for the respective component or integrated system. This study discusses the use of "Waterfall" and "V" as a hybrid test method. The purpose of this article is to present such a test method.

Keywords: Hardware and Software Interfaces, “Waterfall-V” Testing, Mechatronic System

Introduction

The mechatronics field has had substantial development in recent years, due to the compaction of the components and varying the functions of a mechatronic system (Carryer *et al*., 2011).

These systems incorporate mechanical, electrical and electronic components. The control of this system is performed by the implemented software code. This allows functions to be properly carried out, making the final product that much more precise (Alciatore, 2007).

A mechatronic system is usually industrial and can be a production line in a factory, which is the system considered in this study. As this is relatively complex, it will be tested using the "V" model. The “Waterfall” test model is also used for simpler systems (Gausemeier and Moehringer, 2003).

It has several components, which, depending on the complexity, use either the “Waterfall” or "V" model. In the presented test, the stages of the “Waterfall-V” model are followed (Graham *et al*., 2012).

Most mechatronic systems involve the same movement or action. This motion or action can be applied to anything- from a single direction to a large articulated structure. Movement is created by a force or torque that results in acceleration and displacement. Actuators are the devices used to produce this movement or action (Ballas *et al*., 2009).

A sensor is a device, module, machine, or subsystem whose purpose is to detect events or changes in its environment and to send information to other electronic components. This is often a computer processor, as a sensor is always used with other electronic components (Sima and Zapciu, 2021).

The computer elements used in mechatronic systems are the hardware and software used.

In addition to obtaining such a system, the manufacturing and management requirements need to be followed (Bishop, 2007).

Materials and Methods

In Fig. 1, the testing of mechatronic systems was studied. Hardware components repeat certain methods, such as "Waterfall" testing, although this depends on the methods. In terms of the software, various testing methods are used. This is mainly due to the complexity of the system and the interfaces. To test the system, an eloquent example was taken from each component with the representative test method, i.e., a sensor and an actuator.

The test methods studied are those related mainly to software testing, which is successfully applied to a mechatronic system. Based on the experience regarding the structure of the components of the mechatronic system, it was considered that a combination of 2 test

Science Publications

methods ("Waterfall" and "V") is a solution that covers very well the testing process during the manufacture of the mechatronic system.

The Components of the Mechatronic System

The mechanical elements that make up the mechanical structure are the mechanism, the thermo-fluid and the hydraulic aspects (Brusa, 2015).

Electromechanical elements refer to sensors and actuators. A servomotor is an element that turns energy into motion. It can also be used to apply force.

The servomotor is a mechanical device that takes energy, electricity, or liquid - and turns it into a motion. This movement can be in any form, such as locking, gripping, or removing.

Actuators are commonly used in manufacturing or industrial applications and can be used in devices such as motors, pumps, switches and valves.

Fig. 1 presents a schematic of a mechatronic system, with its components (Fotso *et al*., 2012).

These types of components are found in a mechatronic system. Certain testing strategies are used for all system testing.

The specification of the system needs to contain the elements with an order execution center (where the software program is executed), with an interface between the software (digital) and the analog. This is needed to control the mechanical parts, which are operated together through a software program and a real-time response of when the system executes the defined commands.

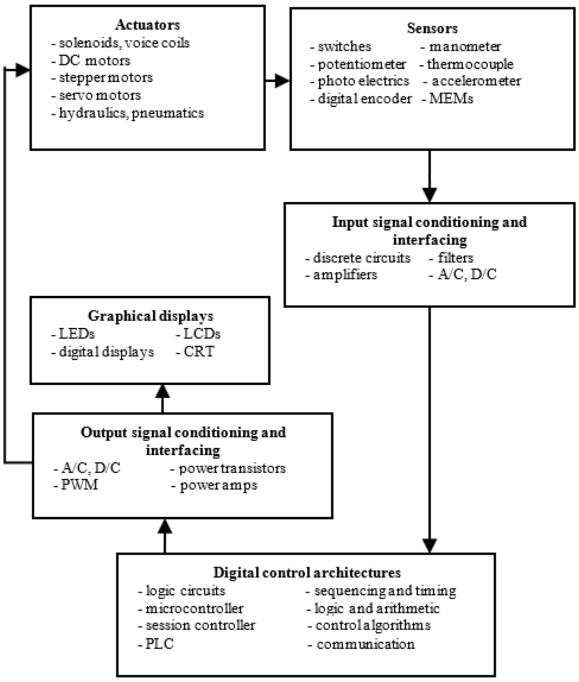

Fig. 1: The components of the mechatronic system

The specifications of the system need to be as clear as possible, mentioned in the contract signed with the client. From test methods, the “Waterfall” model is a breakdown of project activities into linear sequential phases, in which each phase depends on previous results and corresponds to a specialization of tasks. The approach is typical for certain areas of engineering design. It tends to be among the less iterative and flexible approaches, due to the progress flow in one direction ("down" as a cascade) through the phases of design, initiation, analysis, design, construction, testing, implementation and maintenance.

The “Waterfall” development model was originally part of the manufacturing and construction industries. A highly structured physical environment has made the design changes to be more costly much earlier in the development process. There were no recognized alternatives to knowledge-based creative work afterward. Thus, the “Waterfall” model argues that it should move to another phase only when the previous phase is reviewed and verified.

The “V” model is a simple version of the traditional “Waterfall” system or software development model. It is based on the “cascading” model emphasizing verification and validation. Model “V” takes the bottom of the “Waterfall” model and bends it upwards in a “V” shape so that the activities on the right check or validate the work products of the activity on the left. The left side of Model “V” represents the analysis activities that break down users' needs into small, easy-to-manage pieces, while the right side of “V” shows the corresponding synthesis activities that agree (and test) these parts into a system that satisfies user needs. In the “Waterfall” model, the “V”-shaped life cycle is a sequential way of executing processes. Each phase must be completed before the start of the next phase. Product testing is planned in parallel with a corresponding development phase in model “V”.

In the “incremental” model, the whole requirement is divided into various variants. Several development cycles take place, which makes the life cycle "multi-cascade". The cycles are then divided into smaller, easier-to-manage modules. Each module goes through the stages of requirements, design, implementation and testing. A working version of the software is produced during the first module. This creates a faster working version during the software lifecycle. Each later version of the module adds previous versions. The process continues until it is completed.

An “iterative” or “evolutive” life cycle model is also called evolutionary as it does not try to start with a complete specification of the requirements but evolves as the product is developed. Instead, development begins by specifying and implementing only a portion of the software, which can then be revised to identify additional requirements. This process is then repeated, producing a new version of the software for each cycle of the model

(Karnopp *et al*., 2006).

The "agile" development model is a type of incremental model as it is developed in fast, incremental cycles. This leads to small incremental versions with each version of previous features. Each version is thoroughly tested to ensure the quality of the software is maintained. This model is used for time-critical applications. Extreme Programming (XP) is currently one of the most popular "agile" development lifecycle models (Sima and Zapciu, 2020).

The System and its Interfaces

Now, we must insert the proper components and realize their interfaces. The specialized teams are working in parallel. Interfaces are the compatible connections between components. These can be external, which refers to the interaction between the mechatronic system and the external environment and internal, made between the components of the mechatronic system.

A mechatronic system contains a "chain" of interfaces of its components, which are of several types (mechanical, electrical, electronic).

These interfaces are classified according to the type of input and output signal conversion (Dolga, 2011):

- Modifies physical properties (mechanical signal -> electrical)
- Changes the signal encoding mode (analog -> digital)
- Changes the signal transfer mode (parallel transfer -> serial; asynchronous -> synchronous, etc.)

Another classification is based on the adaptation of the input and output signals (Dolga, 2007):

- Zero interface - does not involve any conversion
- Passive interface - no power source
- Active interface - implies the existence of an additional source of energy for conversion
- Intelligent interface, which involves programmable signal conversion, in the case of a microprocessor

The complexity of design issues has led to various approaches to construction. This assembles the need for the "V" model, which includes all the steps from idea generation to product creation.

Simple and clearly defined systems are the only ones that can start from functional requirements, leading to system design and development of components in the mechanical, electronic and IT fields, to have a mechatronic system. The communication between the team members helps to achieve this properly. Communication between team members helps achieve a correctly made mechatronic system.

The interface is a "border" between two subsystems. The exchange of information between two components of the system is possible if there is a common concept and a common coding system. The interface refers to all the ways (buttons, graphic display, etc.) to supervise and assist the processes in a system.

The interface of the electronic instruments (multimeter, signal generator, oscilloscope, etc.), computer systems, sensors, actuators, etc. needs to be performed with the centralized computing system when doing an electrical test.

Afterward validating the software from a technological point of view is required. The human-machine interface and the machine-machine interfaces are considered in this process.

Testing the System

The testing of the mechatronic system is performed depending on the field (mechanical, electrical, information theory). For the mechanical part, standard methods are defined for the three fields participating in the mechatronic integration.

For the mechanical field we need to specify the problem, define functions and structures, find solutions and principles, do a division of feasible modules (into sub-activities), model on components, modeling on the final product and draw up the execution plan and the instructions for use.

For the electrical and electronic field, we need to have clear specifications, a description of the system with an overview, a description of the algorithms used, of the interfaces, the logical functioning and integration needs to be done.

For information technology (software code) we need to define the problem, to do a requirements analysis, design and implementation and component testing, system testing, to finally use the product.

One controller that meets the requirements of a mechatronic system is the Arduino. Even if it is not industrial, it can be used on systems where interfaces are tested (Sima, 2022).

It is difficult to distinguish between a mechatronic system and an electronic system.

In terms of structure, an electronic circuit is part of the mechatronic system, but it is not a mechatronic product. Because the mechatronic system encompasses these areas, the finished product is easy to confuse.

Throughout this process, the design must be kept in mind. Design procedures consist of rules to follow; recommended sequences that provide acceptable solutions.

Depending on the product development lifecycle, a specific model may be used. Often, a combination of the "Waterfall" and "V" models is used, represented in Fig. 2.

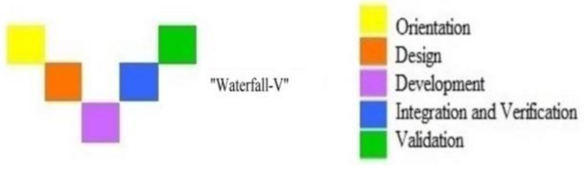

Fig. 2: The "waterfall-V" model

In product development, some steps repeat. These are highlighted in a different colors.

The complexity of the mechatronic system may vary. The “Waterfall” approach is used for deliverables in any field of activity where there are detailed, clear, measurable and demonstrable initial specifications. These are specified in the financing agreement (predictability in execution). This approach is one of the simplest.

The “agile” adaptive approach is used for deliverables in areas where the initial specifications are very general, with no need for a detail.

The hybrid approach combines the predictive and adaptive approaches.

In this case, the following are highlighted some components of the deliverable that are clearly defined from the beginning and for them, the “cascade” method is adopted (predictive) and other components are completed during the project and the “agile” (adaptive) method is adopted for them.

The testing of the mechatronic system is performed depending on the field (mechanical, electrical, information theory). Based on the test methods, using a hybrid method, “incremental”, “iterative” or “agile” give the best results. The production of the product must follow its life cycle, with its phases. It starts with guidance, then designs, development, integration, verification and validation.

Results and Discussion

The results indicate that upon knowing the test methods and how they intertwine, along with the characteristics of the mechatronic system, a hybrid test method can be realized. This test method applies to only one component type. Applying a test method for a mechatronic element that does not require this leads to too little or excessive testing. Both cases would lead to inaccurate results. For linear components, an actuator was chosen as the representative element. For a more complex element, a sensor was chosen.

The Testing of an Actuator

A more predictive approach can be used as the properties of an actuator (for example, a motor, with its type) is known, along with a need for a more precise tool. There are generally no variable functionalities. However, they are known by their specifications. The actuator is only operated according to the technical specifications, while the integration with other components does not have to be done beforehand.

To test the actuator software, the limitations of the motor versus the functionalities used must be identified. The software code checks whether the motor has reached a certain position after the motor has been switched off.

Regarding the implementation and testing, once the design is complete, the requirements are divided into modules and code implementation begins. There is motion detection by other parts of the mechatronic system. The system detects TRUE: The desired position has been detected for the moving object and the engine must be stopped or FALSE: The engine is moving.

The approach method is the predictive one of the “Waterfall” types, because they can be tracked (in “cascade”) if the functionalities are reached (sub-activity A).

Limit check is performed: The tape is moving and FALSE is maintained; the object is identified and the status changes to TRUE. This is a “black-box” test (or so-called "black box", where only the inputs and outputs are known), the result (detection) is checked for a certain condition.

In the next step (sub-activity B) we can use the “Waterfall-V” method because we have some interactions between the motor and LEDs, for example (to indicate a certain position or status of the system). For simplicity, when the object is not detected (by the sensor), an LED is lit. For this second sub-activity, one actuator is not enough as other components (sensors, LEDs, etc.) need to be integrated. Exploratory testing is thus more effective because more outputs must be verified (object detection, LED lighting, etc.) and the purpose is more ambiguous or vaguely defined - no specific test design, plan, or approach is used. Exploratory testing is an unwritten approach to software testing, where the tester is free to select any possible methodology for testing the software. This is a common practice of software developers who use their personal skills to test developed and/or coded software. Exploratory testing simultaneously tests the functionality and operations of the software, while identifying any functional or technical issues in it. The purpose of exploratory testing is to optimize and improve the software in every way possible. In the development of the software actuator product, testing is performed on several levels: On modules (actuator only, sub-activity A), integration testing (LED, actuator, sensor: Sub-activity B), system testing and acceptance testing (validation). The “V” test model is more suitable in the developed phases of the mechatronic system because the hardware’s integrated parts must be made so the actuator code can be tested. To test the actuator hardware, the connections of the actuator assembly (motors plus adapter) within the mechatronic system need to be followed. In this situation we need to power the actuator (motor with adapter), which must provide the necessary power for the movement; In the mechatronic system created, a separate power source helps the system to operate the actuator better and create the connections to the control module, where the commands in the software code are supplied to the engine (actuation or shutdown). In this type of testing, the “monkey”-testing type can be used, which is a technique in which the user tests the application or system by providing random inputs and verifying behavior or seeing if the application or system will crash. This "monkey” test is implemented as random unit tests and fits these hardware components. During hardware testing, current and resistance testing must be performed with the meter.

The Testing of a Sensor

A more adaptive approach can be used, despite the properties of the sensor being known, as their placement and role can vary. In addition, other dependencies may be added for reporting multiple intermediate states and integration with other components must be done beforehand. For software testing of the sensor, its functions are known. These are state detection and are reported at a defined interval in the. In the software code, only the Boolean state is implemented. From this sensor point of view, testing is easier, shown through the actuator hardware. The method of approach is predictive because the functionalities can be followed. The "black-box" test, i.e., the response is checked for a certain state (object detection). Even if the test is predictive, the code executed and the method of execution can create many problems if a specific command does not result in the desired outcome.

As a test, it is performed at an advanced stage, after most components have already been installed and integrated into the system. The implementation of the sensor in the system is done at the end, as a last "adjustment" of the system functions. The positioning of a sensor is complicated. As such, the experience of the team from other projects and the general manager is very important. Position sensors can vary in number. However, too much implementation of several sensors makes detection difficult. For hardware testing of the sensor, its connections must be tracked, along with the required response (object detection) within the mechatronic system.

The approach method is software testing of the sensor. As a general test procedure, a test method should be considered: "Waterfall", "agile", "incremental", "RAD", "iterative", "spiral", "prototype" or "V" model and variations of "V". The chosen testing strategy needs to cover all components of the system. The "Waterfall" and "V" models would cover all the phases necessary for the test. Only in exceptional cases (complex system functions, one of the general testing procedures should be selected. The test procedure according to the component subsystems (sensor type, actuator, system power supply) must follow the specifications of the specification and the standards for the electronic equipment must be followed. These would be those of electromagnetic compatibility, checking at low voltage currents, heating/operating temperature, etc.

Conclusion

As a test method, the mechatronic system must be divided into parts: Electrical, mechanical and electronic. On each side, there are standard methods for mechatronic integration. It tested two components of the mechatronic system: The actuator and the sensor. They encompass virtually all the testing methods discussed for a mechatronic system, which is why we chose them.

Apart from the model presented in Fig. 2, some other models of testing could be used. It could be "incremental", "iterative" or "evolutive", or the "agile" type. Given the constraints of the projects and the complexity of the solution, the direction of the product development is chosen from one of the types enumerated. Regarding the testing of hardware and software interfaces in mechatronic systems, this study brings novelty elements in terms of testing.

Mechatronic systems are gradually evolving and becoming more complex. Solutions are increasingly appearing on a company's digital platform and testing must take these solutions into account. Without this digital platform, a certain product would be common, more ordinary, without the latest elements desired by users.

Once an improved product is created, it needs to be tested correctly as soon as possible and delivered to the customer. As a pure test method, the "Waterfall" or "V" testing can be too simple ("Waterfall") or too complex (in "V"). Therefore, a hybrid method is chosen for testing, so that the team fits within the budget and in time.

The "Waterfall-V" method offers one of the best testing solutions for mechatronic systems, where elements with known status and characteristics are combined with complex elements and integrated software.

Acknowledgment

The authors thank the members of the Department of Robots and Production Systems from the Faculty of Industrial and Robotics Engineering, University POLITEHNICA of Bucharest, Romania for their support and useful suggestions.

Author's Contributions

All authors equally contributed in this work.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues are involved.

References

Alciatore, D. G. (2007). Introduction to mechatronics and measurement systems. McGraw-Hill Education. ISBN: 978-0-070-64814-2.

Ballas, R.G., Pfeiffer, G., & Werthschützky, R. (2009). Elektromechanische Systeme der Mikrotechnik und Mechatronik, 2. Auflage, Springer, Berlin/Heidelberg, Germany,
ISBN: 978-3-540-89320-2.

Bishop, R. H. (2007). The Mechatronics Handbook, CRC Press, Boca Raton, USA, ISBN-10: 978-0-8493-9257-3.

Brusa, E. (2015). Mechatronics: Principles, Technologies and Applications, Nova Science Publishers Inc, New York, USA, ISBN-10: 978-1-6348-2801-7. https://novapublishers.com/shop/mechatronics-principles-technologies-and-applications/

Carryer, J. E., Ohline, R. M., & Kenny, T. W. (2011). Introduction to mechatronic design.

Dolga, V. (2007). Proiectarea sistemelor mecatronice, Ed. Politehnica, ISBN-10: 978-973-625-573-1.

Dolga, V. (2011). Mecatronică. Teoria sistemelor, Ed. Politehnica, ISBN-10: 978-606-554-235-8.

Fotso, A. B., Wasgint, R., & Achim, R., (2012), State of the art for mechatronic design concepts, 8th IEEE/ASME International Conference on Mechatronics and Embedded Systems and Applications (MESA), Suzhou, China.
ISBN-10: 978-1-4673-2347-5.

Gausemeier, J., & Moehringer, S. (2003). New guideline VDI 2206 A flexible procedure model for the design of mechatronic systems. In DS 31: Proceedings of ICED 03, the 14th International Conference on Engineering Design, Stockholm.

Graham D., van Veenendaal, E., Evans, I., & Black, R. (2012). Foundations of Software Testing: ISTQB Certification, Third Edition Cengage Learning, ISBN: 978-1-4080-4405-6.

Karnopp, D. C., Margolis, D. L., & Rosenberg R. C. (2006). System Dynamics: Modeling and Simulation of Mechatronic Systems, Fourth Edition, Wiley, New Jersey, USA, ISBN-10: 978-0-4717-0965-7.

Sima, L. M. (2022). Using Arduino PLC and Robot within Mechatronic Systems, International Journal of Ad hoc, Sensor and Ubiquitous Computing (IJASUC), February 2022, Volume 13, Number 1, Chennai, India, ISSN (print) 0976-2205, (online) 0976-1764. https://airccse.org/journal/ijasuc/vol13.html

Sima, L. M., & Zapciu, M. (2020). Connections and Interfaces of Mechatronic Components on Digital Factory, Conference Proceedings of the Academy of Romanian Scientists, PRODUCTICA Scientific Session, Volume 12, Number 1/2020, Bucharest, Romania, ISSN (print) 2067-2160, (online) 2067-9564. http://aos.ro/editura/proceedings/archive/vol-12-no-1-2020

Sima, L. M., & Zapciu, M. (2021). A Testing Model for the Mechatronic System, U.P.B. Sci. Bull., Series D, Vol. 83, Issue 1, 2021, ISSN (print) 1454-2358, (online) 2286-3699, pg. 113-124.