Journal of Electrical Engineering and Automation (ISSN: 2582-3051)
www.irojournals.com/iroeea/

Future Robotics System Design Challenges including Software Engineering Automation – A Report

P. V. Arul Kumar

Principal and Associate Professor, Department of Mechanical Engineering, Bharath Niketan Engineering College, Theni, India

E-mail: arulkumarveera@gmail.com

Abstract

Recently, there is a meteoric rise of automated robots all around the globe to aid people in carrying out useful jobs. However, unlike industrial robots, service robots pose significant software engineering issues due to the fact that they must function in very varied situations. As a field with serious safety concerns, service robots also need safe and reliable methods of software development. These are evaluated in this research for the first time using empirical data. This article encompasses the details of both robotics by industry professionals and academic researchers, as well as the unique qualities of robotics' software engineering. Additionally, the common challenges encountered during automation, as well as the solutions that have been implemented to address them have been summarized. Moreover, what researchers and practitioners should do, has been discussed in the paper's conclusion.

Keywords: Robotics, software engineering automation, mobile robot, automated design selection, robotics simulation challenges, robotic process automation

1. Introduction

The competition between factories is heating up as a result of the difficult worldwide market. Finding and implementing novel approaches to product development that reduce costs without sacrificing quality or the ability to meet the needs of the market is becoming more important. Despite the general downward trend in economic activity, computer capacity is a resource that is both stable and expanding. Time-consuming calculations and regular manufacturing procedures are two examples of how computers and automation have supplanted human labor. Once a job has been thoroughly specified, computers and gadgets are unsurpassed in their ability to carry it out repeatedly at high speed and continuous

DOI: https://doi.org/10.36548/jeea.2022.3.005
Received: 28.07.2022, received in revised form: 19.08.2022, accepted: 05.09.2022, published: 15.09.2022

precision. Automation in manufacturing has effectively enhanced both output and quality, prompting much research into the machines' speed and precision. The developed world has automated its industries to take advantage of the low cost of labor in the poor world [1-5].

However, the documented downsides of automation in production must be taken into account. The industry has learnt the hard way that it is counterproductive to try to make people obsolete in their own industry. Because of the fundamental distinctions between people and robots, these efforts have failed. Because of their inadequacy for creative and intuitive readings, robots will never fully replace people and must be limited to certain tasks. The primary feature of machines is their ability to do tasks that have been carefully specified. This is what allows them to get so much done, but it's also the major reason they struggle to adapt to ambiguous change [6-10].

From now on, machines should be used for what they were originally intended: to do completely specified, non-creative, iterative tasks across all disciplines, including design. There has been a lot of focus on the fact that engineers see a significant chunk of the design as mundane and repetitive. The design process can be sped up and engineers may devote more time to innovative, intuitive design if these jobs are automated.

This is despite the fact that software is becoming more vital in robotics. Cyber-Physical Systems (CPSs) like robotic systems are complex hybrids of computers and the physical world. Software Engineering (SE) has generally been seen as an ancillary problem of mechanical system building [8], despite its helpful function in other CPS fields (such as automotive and aeronautics). One possible explanation is that robots used in industrial automation come equipped with their own unique controllers for the activities they do on a regular basis. Due to this, programming is simplified. The mechanical, electrical, and automated control components are the workhorses [9]. This allows them to function independently in settings that are otherwise very dissimilar.

1.1 Research Questions

Question 1: What methods are used in SE while designing a service robot?

In this analysis, SE concepts like procedures and quality assurance methodologies are used. Moreover, it looks at the ecosystem of related technologies, such as frameworks, middleware, programming languages, and tools. Finally, why and how frequently individuals recycle existing solutions rather than developing brand new ones have been investigated.

Question 2: To what extent does robotics SE vary from SE in other fields?

The features, hardware, autonomy needs, and deployment context diversity of robots are of particular interest and it reveals that robotics SE is apart from SE in other fields.

Question 3: What difficulties do professionals have when using SE for service robots?

This work draws out the most typical SE issues dealt with by service robotics professionals, and the methods used to fix them.

2. Literature Survey

Several recent survey articles record recent progress in creating standard solutions, reference designs, and model-driven methods to design reusable robotic systems; these publications' findings are briefly summarized here.

Brugali et al. [11] outlined the following five obstacles that are characteristic to robotics:

- Methods for determining the constants in robotic system needs.
- Define abstract models to manage hardware/software differences.
- Deployment of mechanical applications in production settings.
- The smooth transition from prototype testing and debugging to actual system implementation.

Highly disparate technologies are required for the development and integration of software control systems.

In [12], Elkady et al., illustrated a state-of-the-art robotic middleware frameworks literature review and attribute-based bibliography. One such piece of software that automates the hardware abstraction layer and makes it easier to create robot control applications is dynamic. On the other hand, they noted that evaluating requirements for robotic software lacked an emphasis on architecture-specific assessment. Ingrand et al. [13] demonstrated that providing robotic systems with flexible and dependable capabilities is fundamental to the concept of autonomy in robotics.

Developers of the Robot Operating System (ROS) were polled and interviewed in [14]. It discussed the problems that arise from recycling resources in this system. Paper [15] offered a thorough mapping investigation of the current level of mobile robotic system safety.

From a systems engineering vantage point, 58 main researches were organized into categories that include safety solution trends, characteristics, and industry adoption. Existing methods were found to be mostly unfit for usage in the unpredictable and complex settings that are often shared with humans.

Only articles chosen for publication are considered for inclusion in this research as secondary sources. So, a poll has been conducted with robotics experts from industry and academia to dig further into these questions.

3. Future Design Challenges

3.1 Design Iteration Challenges

Vehicles such as trains, cars, and planes, as well as mechatronic machinery such as factory robots, are all examples of goods for design iteration. Multidisciplinary Optimization is a new method with the potential to vastly enhance the design iteration process. This enables designers to automatically explore the design space for optimum solutions after creating a map of the multidisciplinary interactions that exist inside a system. Due to the complex interdependencies across the many product disciplines, multidisciplinary design is a labor-intensive and iterative process. The likelihood of success in finding optimum designs improves as the number of design iterations rises. The assessment period limits the number of design iterations that may be performed. The potential to complete more iteration grows as evaluation time's decrease [16-18].

Low-fidelity models suffer from inaccuracies due to the lack of detail they provide. As a result, it becomes more difficult to settle on optimal design solutions. As new information becomes available throughout the design process, these choices are revisited. However, fixing past errors is costly since it requires working with higher-fidelity models, which necessitates more manual procedures and therefore more engineers from various departments are required. The standard design process may be enhanced by giving designers more leeway to experiment at the later stages of the process. Knowledge may be improved via increased model fidelity and the use of holistic design procedures. However, there are several challenges that must be overcome before such a strategy can be implemented. Problems with holistic MDO procedures have been highlighted by Simpson and Martins (2011). The article was based on an MDO workshop that included 48 participants from universities, corporations, and government organizations across the world. Simpson and

Martins, in brief, provide a unified, modular, and highly re-usable design framework that makes use of a unified and parametric representation of geometry [19].

3.2 Complexities in Design Modelling

The modelling difficulties outlined by Simpson and Martins may be overcome, as shown by a wide variety of established solutions. Furthermore, many suggested methods are applied to design tools that were not originally created for automation.

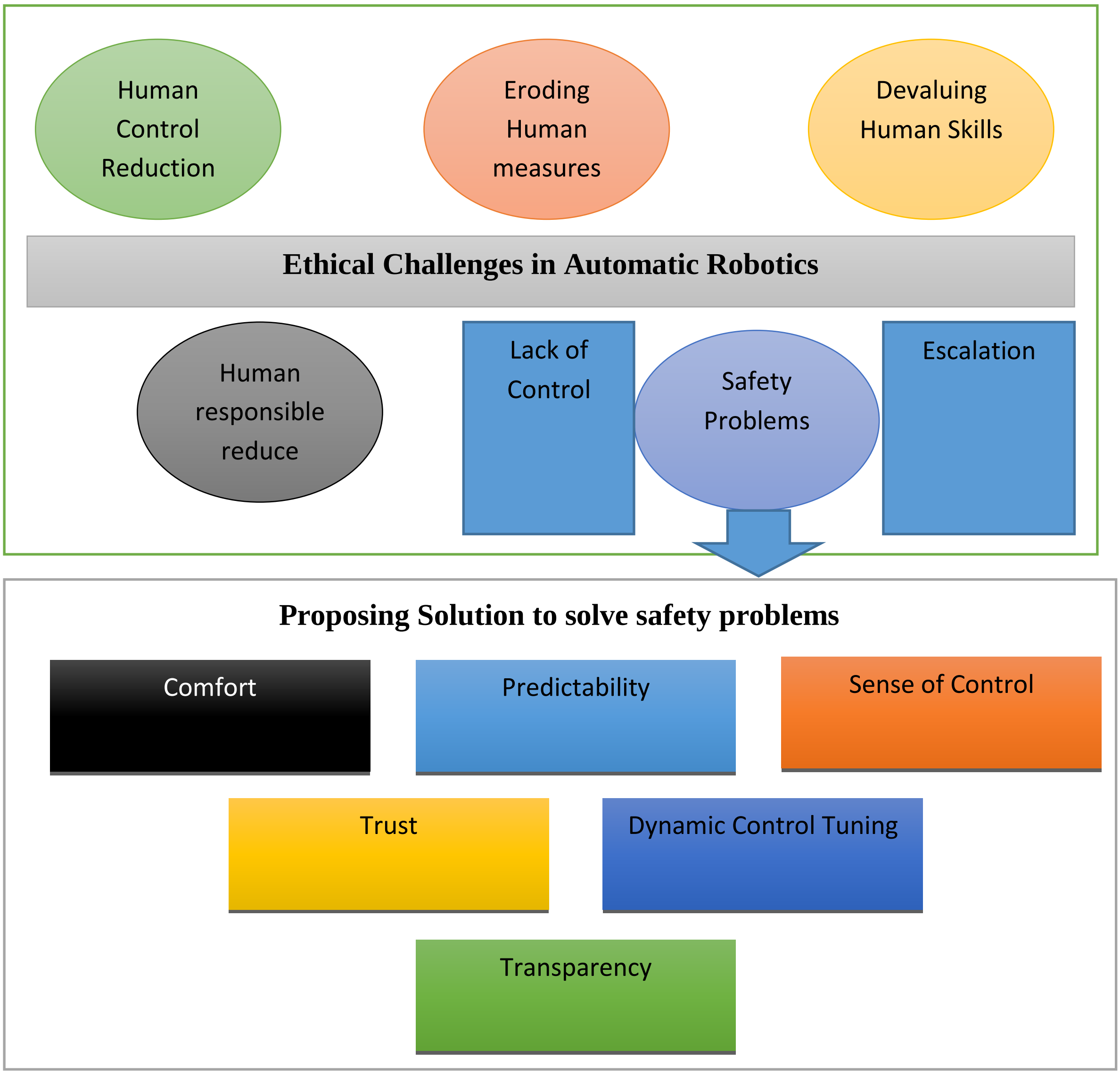

Figure 1. Ethical Challenges and proposing solutions

However, the value of Computer-Aided Design (CAD) has been hotly contested throughout the years. While some predict a more proactive role for CAD, others argue that it's best used as an automated tool for drafting. From a historical perspective, Ullman is true; CAD was first sold as a means to cut down on the overhead of drawing departments. Thus, the lead time was drastically cut by using CAD to create sketches. What was advertised at first as more economical drafting, eventually gave rise to major shifts in practice.

The first shift in practice occurred when semi-automatic sketches were created using previously established geometry models. Second, design engineers began doing their job directly in CAD once CAD departments were combined with the design and production divisions. The next big shift occurred in the late 1980s with the advent of Parametric Associative (PA) CAD, which allowed for minor adjustments to a model's geometry to be made by the adjustment of a few parameters. The software and hardware advancements in the late 1990s made update delays and mistakes far less frequent, allowing PA modeling to have an organic impact on processes [20 - 22].

There is still considerable skepticism about CAD as a design help, despite significant hardware and software advances. In some ways, it concludes that CAD models need too much effort and detail. Due to the time commitment, designers are less likely to scrap subpar concepts.

3.3 A Primary Focus on Templates

To hasten the process of coming up with ideas and testing them, new approaches are needed. During the idea creation and assessment phase, engineers should be able to work at low-level and non-creative CAD operations are reduced or eliminated altogether. It stands to reason that fewer manual processes are needed for lower-level and non-creative CAE activities like mesh creation, boundary condition, and load requirements. The thesis [23] provided strategies for the production and automated generation of high-level templates to help get rid of the identified non-creative activity. Principles are comparable to La Rocca's High-Level Primitives (HLP). Parametric LEGO bricks with a predefined set of design and analysis parameters are a good analogy for the fundamentals. They are manufactured and kept in archives, where engineers or AI agents may make topologically informed decisions about which templates to use and then tweak the parametric shape of each template before evaluating the created system [23].

Integrated design frameworks, in which CAD models act as integrators for other CAE models, are made possible with the use of high-level template-driven design. Therefore, parametric DA frameworks are necessary for MDO. The use of DA in mechanical engineering design relies heavily on Geometry Automation (GeA), whereas MDO is dependent on DA.

Conceptual airplane design, load frame design, and Tran’s disciplinary industrial robot design were used to test and validate the design approaches in the studies [24][25]. Most of the rules and regulations that have been established have been inspired by the industrial robot.

3.4 Selection of Automated Designs

The design purposed here is to seek for ideal actuators by considering cost and performance. Cycle Time (CT) serves as the performance metric, while cost is a measure of the robot's overall heft (W).

3.4.1 Static Simulation

In order to test whether or not the actuators are sufficient to counteract the effects of gravity, the static simulation takes into account a number of different robot workspace configurations. The amount of processing time saved by not starting the dynamic simulation if the configuration doesn't fulfil the gravitational loads is called the "penalty value." Following a successful static stimulation, the dynamic model is used to determine the robot's equation of motion.

3.5 Hybrid Approach to Automation

3.5.1 Challenges to the Implementation of Automated Processes

Using robots to do a task, a rising volume of data necessitates more and more sophisticated processing methods, which is why automation is gaining popularity in the Western economy. But ever since its debut, this automated technique has had to overcome a number of obstacles. The accompanying remark demonstrates that their attention is predominantly drawn to three distinct domains: the technological, the organizational, and the socio-economical.

3.5.2 Problems Arising from the Use of Technology

Rules-based robots are notoriously hard to scale since their rules are often authored by humans. Furthermore, they are challenging to keep adaptable, since the data to be processed typically comes in a wide variety of formats and structures, such as emails, online forms, faxes, scans of physical documents, phone conversations, or even financial or sensor data. Some output papers may need actual printing and mailing to customers. Existing system data is often of low quality in either category, making it difficult to automate processes. Machine learning has a hard time training neural networks appropriately if the data is outdated and relies on outmoded rules, checks, and processes.

3.5.3 Problematical Organization

Complex production chains including several approval phases, third-party contractors, end-users, and bewildering internal processes are the basis for modern business operations at many firms. Such organizations often make use of legacy software, which was created prior to the emergence of current business requirements. These bespoke solutions are often rule-based and are managed by a variety of third-parties. When other processes are broken out into separate software solutions, the resulting software might be brittle since it is seldom compatible with modern standards or even with itself, necessitating frequent updates just to keep it running. However, RPA-delivered automation is typically cumbersome and error-prone since it lacks a simple interface and is designed for a specific purpose. However, many companies lack the internal expertise to develop such robots, which is particularly problematic when it comes to managing exceptions and including workarounds.

3.5.4 The Hybrid Method, an All-Inclusive Approach

The development of a software platform and hardware infrastructure that would enable workers to take part in the creation and upkeep software robots powered by various forms of artificial intelligence, such as neural networks and interactive and collaborative machine learning is proposed. Because in-house experts will be in charge of the software robot's training and daily operations, this solution guarantees the company's continued success. In addition, this system would help businesses hold on to their most enthusiastic employees while giving them the opportunity to develop their technical expertise in tandem with other team members as co-maintainers and co-developers.

3.5.5 Barriers to Full Robotics Automation in the Future

The three most difficult goals to achieve are robustness, validation, and dynamic adaptability. The primary methods of dealing with these issues include handling programming mistakes and implementing rigorous development and testing procedures (which should also include testing in the actual world).

A shift from reality to virtual experiences: The state-of-the-art in simulation cannot accurately replicate occurrences in the actual world. As a result, they are only utilized to evaluate the most fundamental aspects of robotics since practitioners don't believe they are accurate enough to test trustworthy robotic systems.

Due to a deficiency in paperwork and long-standing practices: Robotics system development is made more difficult by the absence of standardized approaches, frameworks, libraries, and reference models. Since many already-existing resources aren't well-documented, this complicates the process of making software compatible with others and reusing it. The progress of robotics systems is slowed as a result of this.

Leveraging artificial intelligence methods: Nearly 34% of those who answered the survey found this issue to be difficult. A total of 7 out of 18 interviewees, reported using AI in their work. Five of them use AI-based object and obstacle detection. This work has come up with three strategies to overcome this problem: operating and extending present research to solve key challenges and delivering additional education.

3.6 Proposed Actions

Given the high cost of conducting tests in the real world, strategies that generate realistic test data would be of most use to practitioners as shown in figure 1. This is a suggestion for further study. A robust robotic system requires a repository of real-world, industry-based critical scenarios that can be used to duplicate testing instances and increase system reliability. Professionals in the business world should compile such crucial circumstances. Further, robots' exposure to uncertainty necessitates innovative techniques of instilling resilience in robots. This advocate is used for collaborative efforts between academics and industry professionals to create such tools. While academics are responsible and it comes up with concepts and creates prototypes, it is the expertise of professionals in the field that will turn the prototype into a useful tool.

4. Conclusion

This study has examined the following points in the Robotics system designing:

- Typical methods used by professionals when creating robotic applications,
- The key features of robotic Software Engineering (SE), and
- The usual difficulties professionals experience and the methods they use to overcome them.

In order to successfully interact with (and respond to) their environment and people, robotic systems need to have increased degrees of autonomy in decision-making and planning. The creation of software for such systems is fraught with difficulty and requires methodical SE techniques. Findings from this study highlight the widespread absence of SE best practices in robotics development, despite widespread recognition of the problem. It is believed that these techniques have the potential to address a number of robotics' pressing issues.

References

[1] C. B. Frey and M. A. Osborne, “The future of employment: How susceptible are jobs to computerisation?”, Technological Forecasting and Social Change, vol. 114, pp. 254–280, 2017.

[2] M. C. Lacity and L. P. Willcocks, “A New Approach to Automating Services.”, MIT Sloan Management Review, vol. 58, no. 1, pp. 40–49, 2016. Available: mitsmr.com/2cUzK69.

[3] A. Asatiani and E. Penttinen, “Turning robotic process automation into commercial success - Case OpusCapita”, Journal of Information Technology Teaching Cases, vol. 6, no. 2, pp. 67–74, 2016.

[4] L. Willcocks, M. Lacity, and A. Craig, “Robotic process automation: Strategic transformation lever for global business services?”, Journal of Information Technology Teaching Cases, vol. 7, no. 1, pp. 17–28, 2017.

[5] Hans Utz, Stefan Sablatnog, Stefan Enderle, and Gerhard Kraetzschmar. 2002. Miro-middleware for mobile robot applications. IEEE Transactions on Robotics and Automation 18, 4 (2002), 493–497.

[6] Yentl Van Tendeloo and Hans Vangheluwe. 2017. The Modelverse: a tool for multi-paradigm modelling and simulation. In WSC. IEEE, 944–955.

[7] Hans Vangheluwe, Juan De Lara, and Pieter J Mosterman. 2002. An introduction to multi-paradigm modelling and simulation. In AIS. 9–20.

[8] E. Rocklage, H. Kraft, A. Karatas, and J. Seewig, “Automated scenario generation for regression testing of autonomous vehicles,” in International Conference on Intelligent Transportation Systems, ser. ITSC ’17, 2017, pp. 476–483.

[9] Claes Wohlin, Per Runeson, Martin Höst, Magnus C Ohlsson, Björn Regnell, and Anders Wesslén. 2012. Experimentation in software engineering. Springer Science & Business Media.

[10] I. Dhiah el Diehn, S. Alouneh, R. Obermaisser et al., “Incremental, distributed, and concurrent service coordination for reliable and deterministic systems-of-systems,” IEEE Systems Journal, 2020.

[11] Davide Brugali, Arvin Agah, Bruce MacDonald, Issa A. D. Nesnas, and William D. Smart. 2007. Trends in Robot Software Domain Engineering. Springer Berlin Heidelberg, Berlin, Heidelberg, 3–8.

[12] Ayssam Elkady and Tarek Sobh. 2012. Robotics Middleware: A Comprehensive Literature Survey and Attribute-Based Bibliography. Journal of Robotics (2012).

[13] Félix Ingrand and Malik Ghallab. 2017. Deliberation for autonomous robots: A survey. Artificial Intelligence 247 (2017), 10–44.

[14] Pablo Estefó, Jocelyn Simmonds, Romain Robbes, and Johan Fabry. 2019. The Robot Operating System: Package reuse and community dynamics. Journal of Systems and Software 151 (2019), 226–242.

[15] Darko Bozhinoski, Davide Di Ruscio, Ivano Malavolta, Patrizio Pelliccione, and Ivica Crnkovic. 2019. Safety for mobile robotic systems: A systematic mapping study from a software engineering perspective. Journal of Systems and Software 151 (2019), 150–179.

[16] Aakash Ahmad and Muhammad Ali Babar. 2016. Software Architectures for Robotic Systems: A Systematic Mapping Study. Journal of Systems and Software 122 (2016), 16–39.

[17] Harald Altinger. 2017. State of the Art Software Development in the Automotive Industry and Analysis Upon Applicability of Software Fault Prediction. Cuvillier Verlag.

[18] Cezary Biele and Pawel Kobylinski. 2018. Surface Recalibration as a New Method Improving Gaze-Based Human-Computer Interaction. In International Conference on Intelligent Human Systems Integration. Springer, 197–202.

[19] Andrew T. Duchowski, Krzysztof Krejtz, Izabela Krejtz, Cezary Biele, Anna Niedzielska, Peter Kiefer, Martin Raubal, and Ioannis Giannopoulos. 2018. The Index of Pupillary Activity: Measuring Cognitive Load Vis-à-vis Task Difficulty with Pupil Oscillation. In Proc. of the 2018 CHI Conf. on Human Factors in Computing Systems (CHI ’18). ACM, NY, USA, Article 282.

[20] Carl Benedikt Frey and Michael A Osborne. 2017. The future of employment: how susceptible are jobs to computerisation? Technological forecasting and social change 114 (2017), 254–280.

[21] Wiesław Kope´c, Radosław Nielek, and Adam Wierzbicki. 2018. Guidelines Towards Better Participation of Older Adults in Software Development Processes using a new SPIRAL Method and Participatory Approach. In Proceedings of the CHASE'18: International Workshop on Cooperative and Human Aspects of Software (ICSE '18). ACM, New York, NY, USA. DOI: http://dx.doi.org/10.1145/3195836.3195840.

[22] D. S. Katz, M. Zizyte, C. Hutchison, D. Guttendorf, P. E. Lanigan, E. Sample, P. Koopman, M. Wagner, and C. Le Goues, "Robustness inside out testing," in Dependable Systems and Networks – Industry Track, ser. DSN-I, 2020, pp. 1–4.

[23] A. Juliani, V.-P. Berges, E. Vckay, Y. Gao, H. Henry, M. Mattar, and D. Lange, "Unity: A general platform for intelligent agents," arXiv preprint arXiv:1809.02627, 2018.

[24] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in International Conference on Intelligent Robots and Systems. IEEE, 2012, pp. 5026–5033.

[25] A. Dosovitskiy, G. Ros, F. Codevilla, A. L´opez, and V. Koltun, "CARLA: An open urban driving simulator," in Conference on Robot Learning, ser. CoRL, 2017, pp. 1–16.

Author's biography

P. V. Arul Kumar is currently working as a Principal and Associate Professor, in the Department of Mechanical Engineering, Bharath Niketan Engineering College, Theni, India. His area of research includes CAD, CAM and design optimization.