SYNTHESIS-BASED ENGINEERING OF SUPERVISORY CONTROLLERS

Synthesis-based engineering (SBE) provides assistance needed to manage the complexity of guaranteeing safe control, by combining model-based engineering with computer-aided design. It builds on the synthesis procedure computing a correct-by-construction controller from models of the to-be-controlled system and associated requirements. This article explains how SBE has recently been applied in the context of metal 3D printers, helping to make their control software more robust. Additionally, recent developments in SBE are highlighted, showing it can be used successfully in the development of complex, industrial-scale systems.

WAN FOKKINK, MARTIJN GOORDEN, DENNIS HENDRIKS, ASIA VAN DE MORTEL-FRONCZAK, WYTSE OORTWIJN AND KOOS ROODA

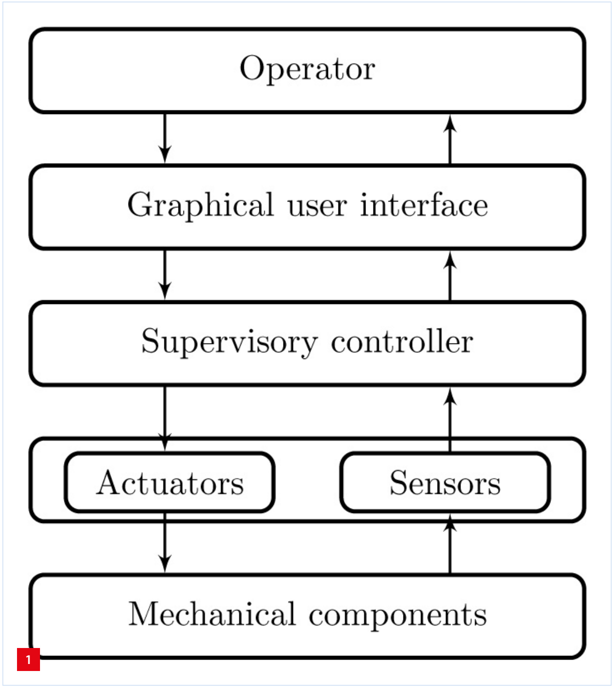
AUTHORS' NOTE

Wan Fokkink is professor of Theoretical Computer Science at Vrije Universiteit Amsterdam (NL), Martijn Goorden is assistant professor at Aalborg University (DK) and expert Industrial Automation at Rijkswaterstaat (the executive agency of the Dutch Ministry of Infrastructure and Water Management), Dennis Hendriks is senior research fellow at TNO-ESI in Eindhoven (NL) and researcher at Radboud University Nijmegen (NL), Asia van de Mortel-Fronczak is assistant professor at Eindhoven University of Technology (TU/e), Wytse Oortwijn is research fellow at TNO-ESI, and Koos Rooda is professor emeritus of Systems Engineering at TU/e.

j.m.v.d.mortel@tue.nl
www.cs.vu.nl
www.cs.aau.dk
www.rijkswaterstaat.nl
www.esi.nl
www.ru.nl/icis
www.tue.nl/cst

Introduction

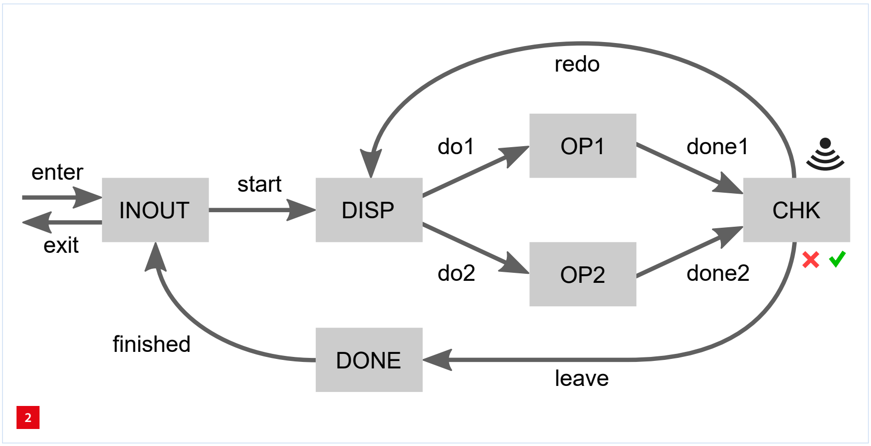
Cyber-physical systems (CPSs) consist of mechatronic components that are controlled by software. This control software typically includes a supervisory controller, which ensures that the various system components interact correctly. For example, a movable bridge for road and water traffic is a CPS consisting of a bridge deck, barriers and traffic lights, among other components. Even though all these components can in principle operate independently, the supervisory controller ensures that the bridge deck can only be opened to allow water traffic to pass through when the road barriers are closed and the traffic lights are red. Such supervisory control software is ubiquitous in our daily lives and is a key component in elevators, traffic, printers, and assembly lines in factories, as well as in metal 3D printers, lithography systems, MRI scanners, etc.



1 Supervisory control structure.

Supervisory controllers often have high reliability demands and need to strictly adhere to safety requirements, in every situation. For example, the bridge should never open when the traffic lights are green, otherwise the consequences could be disastrous. Developing supervisory control software (by hand) in a way that always guarantees safety is notoriously difficult. This is because the various system components can operate independently of each other, meaning that the control software must consider not only the operational behaviours of individual components, but also all possible combinations of their behaviours, which is typically exponential in the number of components.

For illustration, a recent case shows that the supervisory controller of the movable Algera bridge in Krimpen a/d IJssel (NL), needs to consider $8.4 \cdot 10^{25}$ different situations [1]. Manually reasoning about safety in such massive numbers



2

A simple pick & place manufacturing system.

of situations is infeasible, yet any of them being unsafe may materialise into dangerous real-world situations.

In contrast to manual development, supervisory controller synthesis is an algorithmic method that enables automatic generation of supervisory controllers from formal specifications of the system components and the functional system requirements. With formal specifications we mean unambiguous, mathematically precise specifications that describe what the supervisory controller must do and under which assumptions, without specifying how it should do that. Given such a formal system specification, the synthesis algorithm guarantees that the synthesised supervisory controller is correct by construction with respect to this specification. This means that the synthesised controller will supervise the system components in such a way that they can never reach an unsafe or undesired situation.

Moreover, synthesised supervisory controllers are guaranteed to be minimally restrictive. To illustrate this being a desirable property, considering the movable bridge example again, a trivially safe supervisory controller would be one that lets the barriers be closed and the traffic lights be red forever. Although safe, this controller is so strict that the bridge cannot be used in any meaningful way: no road traffic can ever pass over it. Instead, supervisory controller synthesis guarantees that synthesised controllers are correct in the least restrictive way.
Synthesis-based engineering (SBE) is a methodology for engineering control software from formal models of the to-be-controlled system and associated requirements, that uses supervisory controller synthesis to compute a correct-by-construction controller. Since SBE automates to a large degree the design, realisation and verification of controllers, engineers can direct most of their focus to specifying the requirements, and validating them, for instance via simulation. SBE thereby combines model-based engineering with computer-aided design.

SBE has been successfully applied in industry, for example in the development of control software for MRI scanners [2], for wafer logistics in lithography machines [3], and for metal 3D printers [4], which this article later elaborates on. Notably, SBE has reached a level of maturity that allows real-world applications, such as a bridge from Rijkswaterstaat with PLC code generated from a synthesised controller that passed a regular site acceptance test [5], and a car with a synthesised cruise control controller that drove in real-life traffic [6]. These industrial applications demonstrate the maturity of SBE as a methodology that is able to lead to higher-quality controllers at lower cost and effort compared to traditional (manual) engineering.

All mentioned industrial applications were developed using the Eclipse Supervisory Control Engineering Toolkit (ESCET™) [7][8]. Large systems can conveniently be modelled in the CIF language that comes with ESCET. With the help of a simulator, the controlled system behaviour can be explored in an interactive, visual way. ESCET provides an open-source SBE environment allowing interested industrial and academic parties to collaborate and profit from further developments.

The remainder of this article further introduces supervisory controller synthesis and SBE by means of a small example, presents a recent industrial case with Additive Industries regarding control software for metal 3D printers, and high-lights recent advances around ESCET.

Supervisory controller synthesis

A supervisory controller ensures correct cooperation between individual system components [9], and can typically interact with an operator, being for example a person or another system. Figure 1 shows the general structure for supervisory control. At the bottom are the mechanical components to be controlled. Sensors provide the supervisor with the information about the status of the components driven by actuators. Supervisory controllers disable any actuator signals that would violate the requirements. A human operator can typically interact with the supervisor by means of a user interface, for example to manually open a bridge deck to allow ships to pass through, provided that all safety requirements are adhered to (e.g., the traffic lights are red). Otherwise, the operator signal will not be passed to the actuators and feedback will be given to the operator.

In this article, we consider supervisory controllers of discrete-event systems. This means that, although the movements of the mechanical components (e.g., a bridge deck) may be continuous in time, they are initiated by discrete outputs determined by the supervisor (e.g., 'open' and 'close') based on discrete inputs from the system (e.g., from an operator pushing a button, or a limit sensor detecting the desired end point of a movement).

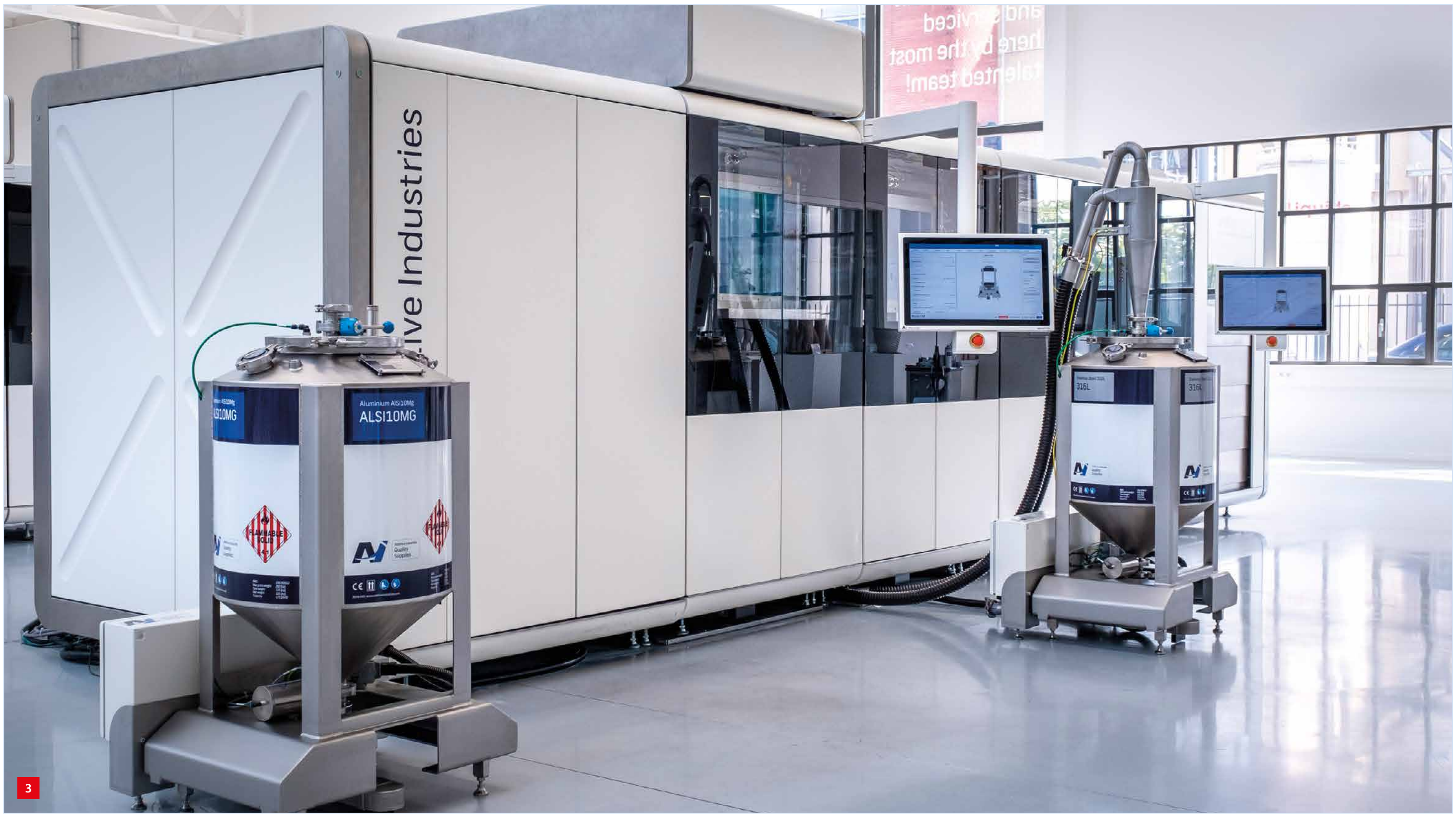
Supervisory controller synthesis automatically computes a correct supervisor based on a formal specification of the mechanical components and the system requirements. The mechanical component specifications together are typically called the plant. The plant model specifies all possible behaviour of the mechanical components as events, and indicates which parts of this behaviour are controllable and which parts are uncontrollable. An example of a controllable event is the action to open a bridge deck, whereas an uncontrollable event may be a signal coming from a sensor. The supervisor can only influence controllable events, by disabling them whenever they would (in)directly lead to requirement violations.

A simple manufacturing system

To demonstrate the value of supervisory controller synthesis, consider Figure 2, which illustrates a simple pick & place manufacturing system. This system (plant) consists of various stations, displayed as rectangles (e.g., INOUT and DISP) that each can contain at most one product. Products can be picked from a station and placed on another unoccupied station as indicated by the arrows between stations. All arrows are labelled with the name of the pick & place action. For example, if INOUT contains a product and DISP is unoccupied, then the 'start' action moves the product from INOUT to DISP. Furthermore, INOUT only allows the 'start' action to be taken for products that just entered the system, and only allows 'exit' to happen for products that are just finished. CHK is a checking station containing a sensor that checks whether some operation has correctly been performed on the product (by OP1 or OP2), and if not, lets the system 'redo' the operation on the product. This loop is repeated until the product is approved by the CHK station.

Even though this pick & place system is simple, it may be difficult to correctly implement its control software in a way that satisfies even straightforward requirements. To illustrate this, suppose that a FIFO requirement is imposed: products must exit the system in the same order as they entered it. Such a requirement is easy to specify: if INOUT labels entered products with an incremental identifier and remembers the identifier of the last product that exited the system, then finished products must always have an identifier that is one higher than the identifier remembered by INOUT.

Although easy to specify, this FIFO requirement is challenging to guarantee, already because products may overtake each other in the OP1 and OP2 stations, or as the result of a 'redo'. To prevent that, the controller must for example only allow 'done1' to happen under the condition that, if there is a product on DISP or OP2, or both, then



The MetalFABG2 metal 3D printer by Additive Industries.

these products must have entered the system later than the product on OP1. Similarly non-trivial conditions are needed for 'start' and 'done1'. Hence, coming up with exactly the right conditions that still guarantee maximal performance, is not trivial. Additionally, the controller must be careful not to let the system deadlock, which may happen when too many products are in the system. To prevent this, 'enter' must only be allowed when at most three products are currently in the system.
This example shows that realising a correct supervisory controller can be non-trivial already in case of relatively simple systems with relatively easy-to-specify requirements. Manually realising supervisory controllers for industrial-scale systems can easily become infeasible, for example due to the sheer depth of speculative reasoning required to cover all corner cases (e.g., 'done1' must sometimes be disallowed to prevent problems many steps later).

Supervisory controller synthesis can automatically generate a correct-by-construction supervisor that is minimally restrictive for this pick & place system and FIFO requirement. Computer assistance is thereby given to engineers, in the sense that they should only formally specify the plant model of the system and the FIFO requirement, which is relatively easy compared to the complexity of the supervisor itself. Supervisory controller synthesis thus can save significant effort and reduce the risk of human error.

The full pick & place system example, worked out in ESCET, is available online [10], including plant models and FIFO requirement, and instructions for applying controller synthesis to it.

Industrial application

Supervisory controller synthesis has been successfully applied in industry and shown to lead to faster development cycles and reduce the number of human errors. It has recently been applied in a case performed together with Additive Industries [4], a manufacturer of metal 3D printers (see Figure 3). The control software of these printers is specified in a model-based way: control is described in terms of state-machine models from which PLC code is generated.

Apart from the ability to simulate and test these models, there is no method in place to guarantee that the controller always respects desired requirements. In particular, Additive Industries was interested in knowing whether the control software adheres to various Machine and Material Damage Control (MMDC) requirements, and if not, what changes to the models would be needed to guarantee them.

The metal 3D printers contain various moving components that should never collide, to prevent machine damage; for example, components for powder extraction and powder handling. To apply supervisory controller synthesis, the control models for these components, together with all relevant context, were specified as plant models in CIF. A total of six MMDC requirements were specified, and formalised as requirements in CIF. Then controller synthesis was applied to see if any additional control conditions would be synthesised, on top of the restrictions already present in the control models.

If additional conditions are synthesised, then this means that the MMDC requirements are not always adhered to. In that case, the synthesised conditions indicate how the control models should be adjusted to become correct in the least restrictive way. So, apart from the ability to point out issues, controller synthesis additionally gives solutions to these problems, thereby supporting constructive engineering.

With respect to at least two MMDC requirements, the synthesis algorithm found potentially violating behaviour. Additional control conditions were synthesised for both these requirements, which further restrict movement control and possible interaction between various components and their environment. Although for this system such violating situations may only occur very rarely, the synthesis effort led to concrete improvements in the control software, making the metal printers more robust.

The engineers of Additive Industries that were involved in this case expressed that the ability to automatically detect subtle problematic corner cases and synthesise fixes for them is valuable design assistance, and they would like to see synthesis applied also to other parts of the control software.

Recent advances

A number of recent developments have further improved the maturity, scalability and industrial applicability of supervisory controller synthesis.

Notably, multilevel synthesis [11] can drastically improve scalability by splitting up the problem of synthesising one complex monolithic supervisory controller, into synthesising multiple simpler controllers that may individually be significantly smaller [12]. This splitting may be done alongside the hierarchical system decomposition tree, so that local supervisors are synthesised for local parts of the system, which together then form the global supervisor.

A case study with multilevel synthesis [13], involving control software for a ship lock in the Wilhelmina Canal in Tilburg (NL), demonstrated that, while a monolithic supervisor would have $6.0 \cdot 10^{24}$ states, applying multilevel synthesis on its 358 requirements and 51 lock components resulted in a significantly reduced multilevel supervisor with $2.9 \cdot 10^{10}$ states.

Another recent development is fault-tolerant supervisory controller synthesis [1], where the supervisor should continue its operation in case of reduced service, for example when a component fails in the system due to a faulty sensor or actuator. This method was applied to generate control software for the Oisterwijksebaan bridge, a rotating bridge in the Wilhelmina canal, which was then successfully used to actually control the physical bridge [14].

Recently, an 'assembler' was developed for supervisory controller synthesis of ship locks in a configure-to-order manner [15]. Users configure the parameters of the system, such as the number of lock heads and the type of levelling system. The assembler then generates models for that particular configuration, for both controller synthesis and simulation. This approach makes SBE conveniently available to a larger group of non-expert users. The assembler is already being used to develop models for a variety of ship locks at Rijkswaterstaat.

Conclusion

Synthesis-based engineering of supervisory controllers is a methodology for engineering control software by formally specifying the components to control, i.e. the plant model, as well as (safety) requirements over these components, which are expressed over the plant. It allows engineers to focus on what the supervisory controller should do instead of how it should do it, and provides design assistance to help manage the complexity of realising correct supervisors. Supervisory controller synthesis has been brought in recent years to such a level that it can be successfully used in the design and development process of complex, industrial-scale systems.

Acknowledgements

Rijkswaterstaat has provided financial and technical support over the past ten years for parts of the research discussed here. Special thanks go to Han Vogel, Harry Lammeretz and Maria Angenent. We also thank Marzhan Baubekova, Pascal Etman, Martin Fabian, Lars Moormann, Ferdie Reijnen, Michel Reniers and Jeroen Verbakel for their research contributions.

Additionally, a part of the discussed research is carried out within the framework of the European ITEA4 18030 Machinaide project (aimed at developing innovative concepts for accessing, searching, analysing and using multiple digital twins) [16] under the responsibility of TNO-ESI in cooperation with Cordis and Additive Industries. The research activities are partly supported by the Netherlands Organization for Applied Scientific Research TNO, the Netherlands Ministry of Economic Affairs and Climate Policy, and TKI-HTSM. We thank Benno Beuting and Frank Mertens from Cordis, and Hans Berkers and Roland Smits from Additive Industries, for the collaboration in the Machinaide project and on the metal 3D printer case described in this article.

REFERENCES

[1] F.F.H. Reijnen, *et al.*, "Structured Synthesis of Fault-Tolerant Supervisory Controllers", *Proceedings of the 10th Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS)*, pp. 894-901, 2018.
[2] R.J.M. Theunissen, *et al.*, "Application of Supervisory Control Synthesis to a Patient Support Table of a Magnetic Resonance Imaging Scanner", *IEEE Transactions on Automation Science and Engineering*, vol. 11 (1), pp. 20-32, 2014.
[3] B. van der Sanden, *et al.*, "Modular model-based supervisory controller design for wafer logistics in lithography machines", *Proceedings of the 18th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pp. 416-425, 2015.
[4] W.H.M. Oortwijn, and D. Hendriks, "Description of pilots for enhancing computer-aided design for low-code development by applying synthesis", Deliverable 5.2a of the ITEA4 Machinaide project, 2023.
[5] F.F.H. Reijnen, *et al.*, "Supervisory controller synthesis and implementation for safety PLCs", *Discrete Event Dynamic Systems*, vol. 31 (1), pp. 115-141, 2022.
[6] T. Korssen, *et al.*, "Systematic Model-Based Design and Implementation of Supervisors for Advanced Driver Assistance Systems", *IEEE Transactions on Intelligent Transportation Systems*, vol. 19 (2), pp. 533-544, 2018.
[7] W.J. Fokkink, *et al.*, "Eclipse ESCET™: The Eclipse Supervisory Control Engineering Toolkit", *Proceedings of the 29th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*, pp. 44-52, 2023.
[8] *www.eclipse.dev/escet*; 'Eclipse', 'Eclipse ESCET' and 'ESCET' are trademarks of Eclipse Foundation, Inc.
[9] T.B. Sheridan, and G. Johannsen, *Monitoring Behavior and Supervisory Control*, Springer New York, 1976.
[10] *www.eclipse.dev/escet/cif/synthesis-based-engineering/example.html* (accessed 8 November 2023).
[11] J. Komenda, T. Masopust, and J.H. van Schuppen, "Control of an engineering-structured multilevel discrete-event system", *Proceedings of the 13th International Workshop on Discrete Event Systems (WODES)*, pp. 103-108, 2016.
[12] M.A. Goorden, *et al.*, "Structuring Multilevel Discrete-Event Systems With Dependence Structure Matrices", *IEEE Transactions on Automatic Control*, vol. 65 (4), pp. 1625-1639, 2020.
[13] W.J. Fokkink, *et al.*, "Supervisor Synthesis: Bridging Theory and Practice, *Computer*, vol. 55, pp. 48-54, 2022.
[14] F.F.H. Reijnen, *et al.*, *Synthesized fault-tolerant supervisory controllers, with an application to a rotating bridge, Computers in Industry*, vol. 130, 103473, 2021.
[15] M.M. Baubekova, "Design of a model assembler for synthesis of ship lock supervisors", EngD thesis, Eindhoven University of Technology, 2023.
[16] *www.machinaide.eu*