INTERNATIONAL DESIGN CONFERENCE – DESIGN 2022
https://doi.org/10.1017/pds.2022.171

Concept of a Multi-Agent System for Optimised and Automated Engineering Change Implementation

O. Radisic-Aberger ✉, T. Weisser, T. Saßmannshausen, J. Wagner and P. Burggräf

University of Siegen, Germany
✉ ognjen.radisic@uni-siegen.de

Abstract

Engineering changes are necessary to stay competitive, unavoidable and occur more frequently with increased product complexity. Currently, scheduling of engineering changes into production and supply chain is a manual process. With new possibilities in the field of artificial intelligence, this publication presents the vision of a flexible multi-agent system with four agents and a single shared database. By autonomously scheduling changes and predicting KPI impacts of implementation dates, the agent-system provides additional capacity and decision-making support to the organisation.

Keywords: engineering change, artificial intelligence (AI), change management, multi-agent systems

1. Introduction

To remain competitive, to remove quality issues, or due to legislative reasons, continuous changes to a product are inevitable. Although companies try to minimise or fully avoid these engineering changes (EC), there is an observable increase in frequency and amount due to changed customer behaviour. A therefore necessary change to a component of a product can propagate to other components, possibly resulting in an avalanche of further changes (Eckert *et al.*, 2004). When implementing ECs into the production environment, multiple process partners need to ensure that a variety of changes are introduced simultaneously at the assembly line. With increasing complexity of the product, in the aeronautical or automotive industry for example, coordinating multiple supply streams becomes pivotal to maintain a steady production process output.

To cope with the increasing complexity, frequency, and volume of EC, automating EC and its associated management is seen as an opportunity (Sharp *et al.*, 2021). While research on automated problem assessment (Weißer *et al.*, 2021), solution finding (Beroule *et al.*, 2014) and impact assessment (Ma *et al.*, 2017) is available, the EC implementation process proves difficult as it remains a communication-intensive process. With the increasing size of global production networks, the optimal implementation of changes becomes increasingly complex. Besides the problem to define an optimal implementation date (Barzizza *et al.*, 2001) and multiple changes being introduced simultaneously (Bhuiyan *et al.*, 2006), the workload during peaks leads to prioritisation losses (Wänström *et al.*, 2006), failing some changes. However, to prevent malfunction in digitalised products for software and hardware compatibility it is necessary to have full match of actual and planned bill of materials. Hence, to improve scheduling of changes a digitalisation of the EC process while retaining a degree of flexibility for handling variety is necessary.

With advances in artificial intelligence (AI), this paper provides a concept, key requirements, and logic for an automated EC implementation control via a multi-agent system (MAS). Our system consists of four agents that build upon a single shared database. The interplay between these agents optimises and partly automates the EC process. The remainder of this contribution is structured as

follows: chapter 2 provides the theoretical background on EC implementation and insight on related publications. Chapter 3 introduces the MAS, whose components are subsequently discussed in chapter 4. Chapter 5 concludes with an outlook on future work and research activities.

2. Theoretical Background

2.1. Engineering Change and Engineering Change Implementation

ECs are any change or modification to the function, behaviour, or structure of a technical artefact (Hamraz *et al.*, 2013). As such, they can occur at any point in a product's lifecycle. The handling of these ECs is further defined as engineering change management, with its goals defined as *Less, Earlier, More Effective, More Efficient*, and *Better* (Fricke *et al.*, 2000). Depending on the product's lifecycle phase, the focus shifts. During early product development, changes are frequent and on short notice to enable earlier experience with the modified product. During series production effectiveness and efficiency come into focus.

From a process view, EC handling can be described in six distinct process steps (Jarratt *et al.*, 2005). As seen in Figure 1, following the emergence of a change trigger an EC request is raised and resolved accordingly through the process steps. Upon approval of a solution, the EC request is transformed into an EC notice. Though the process itself is not linear, as there are iteration loops, once a definitive solution is decided on, the implementation phase is mostly linear. Multiple process partners are tasked with activities such as material planning and homologation. To enable coordination, most companies use a form of change coordination board or committee (Huang *et al.*, 2003). As automating other process steps is already being researched, (e.g. (Sharp *et al.*, 2021; Arnarsson *et al.*, 2021)), this paper focuses on an automated implementation process.

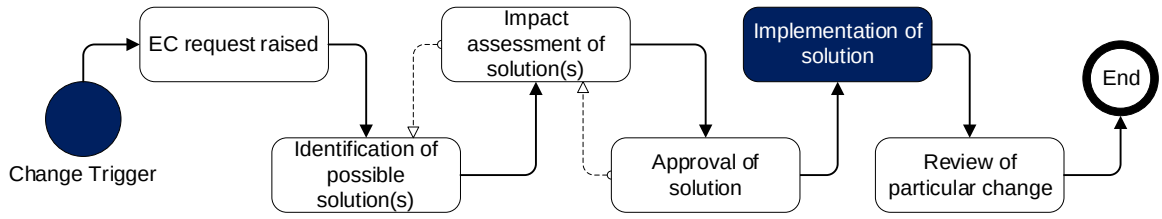

Figure 1. The generic EC process as defined by Jarratt et. al (2005), with the focus area highlighted

EC implementation is a complex process, with different stakeholders involved, each with their own objective. With case studies from the automotive industry, Potdar and Jonnalagedda (2018), as well as Shivankar *et al.* (2015) provide insights and process flow charts with detailed activities described. Both contributions can be summarised as a change coordinator distributing an approved change to various departments within the plant and waiting for their feedback. Upon approval, a material planner organises the logistics for changed parts into production. A generic process for EC implementation however is difficult to define, due to a high degree of customisation triggered by the EC itself. Additionally, EC implementation has been an overlooked research field in the past (Hamraz *et al.*, 2013). Hence, only a few sources describe standardised processes, as well as tools and methods for this step. Thus, in a first step, the MAS should support two specific functions within an EC process, the material planner and the change coordinator.

The objectives of the agents developed for this task can be derived from theory in combination with the case studies described. Two main research paths were discovered within EC implementation research, namely implementation date optimisation and process optimisation (Radisic-Aberger, 2021). On the one hand, some researchers focused on calculating the optimal change effectivity date defining *when* and *how* to implement the EC. Focusing on theoretical exploration, Barzizza *et al.* (2001) and Wänström *et al.* (2006) for instance built on previous work by Diprima (1982) and calculated the optimal dates for an EC to reduce rework and obsolescence cost respectively. On the other hand, research addressed by Bhuiyan *et al.* (2006) as well as Ouertani (2008) among others, simulated efficiency gains through parallelisation of tasks and batching of multiple ECs into one change

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

occurrence. However, EC implementation remains a manual task. For integrating both theoretical research streams into a potential solution, an MAS is proposed, providing the benefit of automatic EC scheduling, optimising the effectivity date while being adaptable to EC variance.

2.2. Applied Methodologies and Related Work

This contribution is built on two methodologies namely the design science research methodology (Hevner *et al.*, 2004) for the development of artifacts as well as the Gaia methodology (Wooldridge *et al.*, 2000) for detailling of the MAS.

The first methdology is used to define the overarching procedure and develop potential future information systems (IS), which are going to be embedded in the industry. The environment according to Figure 2 has been described in chapter 2.1., the EC implementation process.

The second methodology is used for conceptualisation and development of the MAS. As the methodology demands the requirements on the agents first, these are initially developed by usage of the design science research methodology. The role model and function model is further described in chapter 3.

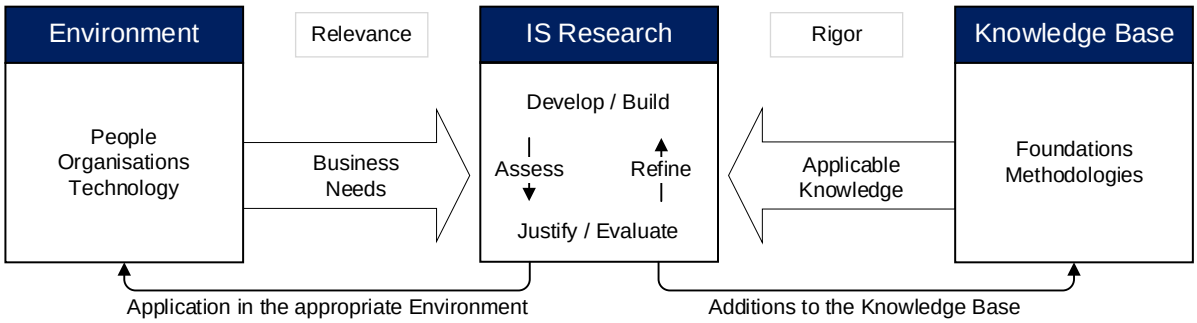

Figure 2. Design science research methodology according to Hevner (2004)

As a proposition for the objectives of the agents the following business needs (BN) are given, according to the tasks described by Potdar and Jonnalagedda (2018) in combination with the theoretical research streams:

- BN1: Define a cross-company optimal EC effectivity date on part level
- BN2: Provide a prediction whether the part will be introduced accordingly, depending on data from other stakeholders
- BN3: Provide an EC schedule, according to the outcome of BN1 and BN2
- BN4: Provide the organisation with feedback

BN1 is a combination of the two theoretical research streams: defining an optimal date for implementation and usage of EC batching to reduce complexity. With the digitalisation of the process, an additional benefit can be raised, as through advanced AI applications, the introduction date of ECs can partly be predicted. This results in BN2, targeted at the industry observations (cf. chapter 2.1) and incorporating data from multiple sources. BN3 is the potential for automating the scheduling of changes, enabling better usage of resources compared to the manual process today. Finally, to remain in control of the process, feedback to the organisation is necessary, resulting in BN4.

After identifying the BN, design science research methodology proposes the establishment of a knowledge base. For this, we performed a systematic literature search (Radisic-Aberger, 2021), and allocated discovered literature in the framework developed by Hamraz *et al.* (2013), with an additional layer of AI usage. Hence, the foundations of the knowledge base are formed by literature on EC and EC management, with supportive literature on MAS. Through the literature search, five related works have been identified, each providing an MAS for solving a distinct EC problem.

From a chronological perspective, the first identified agent-based EC management approach by Moon and Wang (2009), models consumers, producers, and suppliers. By simulating the EC process on a macroscopic level, they were able to show the positive impact of an effective EC process on market shares. To support engineers with the challenging task of finding the optimal configuration after an EC

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

trigger, Beroule *et al.* (2014) developed consensus-seeking agents. They were able to provide potential new configurations based on requirements, functions, and manufacturing, built around *robust* and *flexible* components. Automatic mapping of product data was identified by Bender *et al.* (2015) as a potential of improving the EC process. Thus, they designed a concept for an MAS for mapping geometrical and logistical data, discussed different architectures and arrangements of active and passive agents, and proposed developing a running prototype. Similar to the objective of Beroule *et al.* (2014), Camarillo *et al.* (2017) used case-based reasoning agents to identify potential solutions based on past data. Their system was orientated towards problem solution finding and in comparison to human engineers their prototype achieved 80 % solution accuracy. Finally, Ma *et al.* (2017) used an MAS to better predict change propagation. Comparing it to the change prediction model, their system performed better when calculating propagation paths with multiple changes occurring on multiple parts at once. These five contributions show that MAS can be used to model the complex EC process, as well as to support those involved with EC. However, an MAS in support of EC implementation has not been identified in the literature.

3. Multi-Agent System Approach

As introduced, the EC implementation process involves multiple stakeholders and departments across a company, loosely collaborating through the actions of a change coordinator. Representing loose collaboration between autonomous entities becomes possible through MAS (Kehl *et al.*, 2015; Ma *et al.*, 2017). Hence, we propose to model and handle the EC implementation process via an MAS. As introduced, we employ the Gaia methodology (Wooldridge *et al.*, 2000), to detail the design of the agents. As a initial step, we describe the roles and interactions model of the MAS, as shown in Figure 3. Accordingly, neither the roles nor the interactions model describe the actual function, but rather a general description of responsibilites of the agents, which are then further expanded on as the MAS is developed. The roles and general architecutre of the four agents are afterwards discussed in detail in chapter 4.

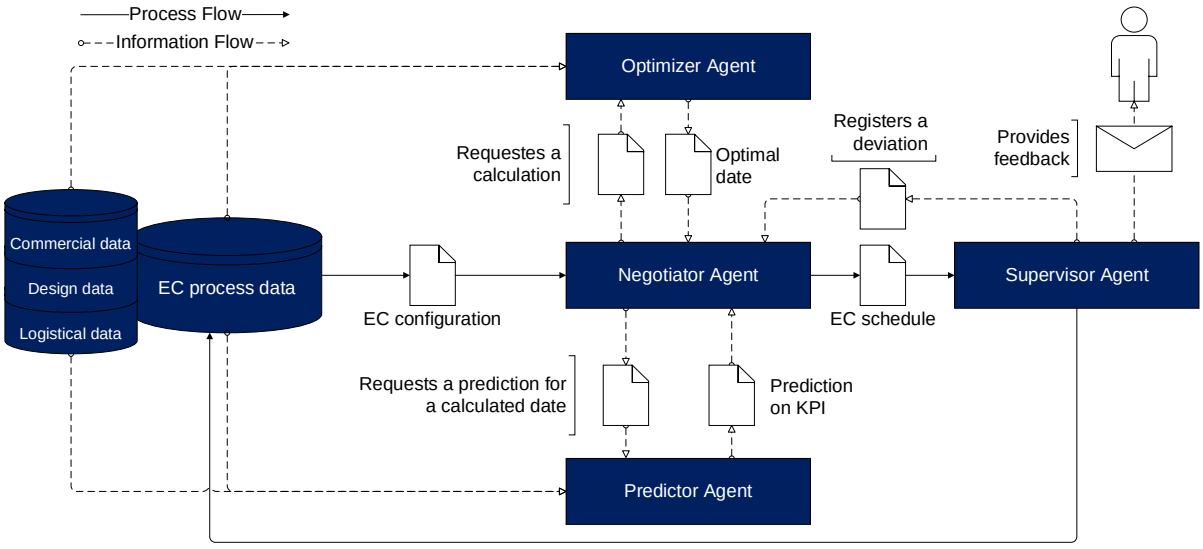

Figure 3. Schematic overview of the role model of the proposed multi-agent system for EC implementation

We envision four agent roles, each responsible for one of the BN in chapter 2.2, and a data source that all agents can access. According to the definition of Russell and Norvig (2016, p. 36ff.) an agent is an autonomous entity, that perceives its environment through sensors, upon which it can act through actuators. Depending on the task, the agent's structure varies. As such, the MAS employs goal-based, utility-based as well as model-based reflex agents (Russell and Norvig, 2016, p. 50ff.). Each agent solves its given task with regard to optimising the performance fulfilling their objective, with their orchestrated procedure defining the MAS. For each agent, a task environment is necessary, defined by

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

performance measure, environment, actuator, and sensor (PEAS) (Russell and Norvig, 2016, p. 40), provided in Table 1.

Within the generic EC process, the proposed MAS starts after the finished approval process and is only tasked with the EC implementation process. The resulting EC configuration document is taken by the 'Negotiator Agent' as the input for its process, starting an optimise-predict-schedule (OPS) sequence.

According to the definition of Russell and Norvig (2016, p. 36ff.) an agent is an autonomous entity, that perceives its environment through sensors, upon which it can act through actuators. Depending on the task, the agent's structure varies. As such, the MAS employs goal-based, utility-based as well as model-based reflex agents (Russell and Norvig, 2016, p. 50ff.). Each agent solves its given task with regard to optimising the performance fulfilling their objective, with their orchestrated procedure defining the MAS. For each agent, a task environment is necessary, defined by performance measure, environment, actuator, and sensor (PEAS) (Russell and Norvig, 2016, p. 40), provided in Table 1.

This agent takes over the coordinating tasks, usually performed by the human change coordinator, with the goal to schedule an EC. From the EC configuration, an 'Optimiser Agent' is triggered, replacing the material planner in the process of Potdar and Jonnalagedda (2018), calculating the optimal change effectivity date based on not only EC process data but logistical, design, and commercial data es well. Having identified a global optimum, the date is provided to the 'Negotiator'. Upon receiving this data, the 'Negotiator' requests a prediction from a 'Predictor Agent', how implementing on this effectivity date, in combination with the EC notice and configuration will impact key performance indicators (KPI), e.g. scheduling accuracy. Based on the reply from the 'Predictor', the 'Negotiator' decides whether it is satisfied with this potential outcome and confidence interval, or whether it retriggers the calculations, demanding to search for a different, this time local, optimum. In case the effectivity date is satisfactory, the EC notice is scheduled for production and assembly, upon which a 'Supervisor Agent' watches over the EC schedule and informs the human in the loop and retriggers the 'Negotiator' in case any deviations or reitariton in the process are registered. Once the EC is implemented, the 'Supervisor' adds the EC notice to the historical EC process database.

Each agent has its own objective and performance measure. Accordingly, the actuators and sensors for each agent differ, reacting to its environment. Chapter 4 discusses these agents and their logic in detail.

Table 1. Performance, environment, actuators, and sensors overview

Agent	Performance Measure	Environment	Actuators	Sensors
Negotiator	Scheduling Accuracy	EC Data	Calculation request, prediction request, comparison algorithm, EC scheduling, request manual override	EC configuration, calculation, prediction, implementation information, scheduling check
Optimiser	Implementation cost and time	Current commercial, design, logistical, EC Data	Classifier algorithm, calculation algorithm	EC configuration, calculation requests, supply chain data, product data, production schedule
Predictor	Prediction quality	Past & current logistical, design, EC Data	Prediction algorithm	Prediction requests, EC configuration, EC process data, EC timing data
Supervisor	Manual intervention	EC schedule	Scheduling check request, feedback distribution	EC schedule data, prediction data, EC process data, EC timing data

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

4. Framework and Service Model of the Agents

4.1. Negotiator Agent

The central agent of the MAS, the 'Negotiator', coordinates tasks and providing an EC schedule, fulfilling BN3. Designed as a learning goal-based agent, the logic follows the flowchart in Figure 4. The decisions depicted do not occur in a sequential matter, but rather simultaneously and represent sensors, that enable him to perceive the environment, and act upon it depending on the acutal state.

Upon initialisation, the agent is in an idle state, waiting for a new EC to become available. Once received, the 'Negotiator' requests to calculate optima for EC effectivity date in regards to time and cost. Additionally, a prediction on KPI impact is requested for the EC case. When available, the 'Negotiator' checks whether every optimum has a prediction. If the conditions are met, the agent compares the predicted KPI and the identified date against a rule set. If the resulting comparison returns a result within the acceptability limits of the agent, the EC is scheduled into the production network. In case the result is deemed unsatisfactory, the 'Negotiator' retriggers the 'Optimiser' by requesting to search for the next best local optimum, restarting the OPS sequence.

Unless a request to control the schedule is received from the 'Supervisor', the agent remains in an idle state until receiving the EC implemented confirmation, resulting in its termination. However, if a deviation is recorded, the 'Negotiator' requests a new prediction to reassess the situation, and dependent on the outcome the schedule is confirmed or a new initial optimum calculation required, effectively restarting the OPS sequence. Finally, a limit should be set on how many loops can be re-run, and in case the global optimum never occurs, a manual override or decision is requested.

Main points to take into consideration when developing this agent are check frequency, data storage concepts, and the ruleset. While the first and second are limited by computational resources, the ruleset is unique according to a company's requirements. Also, the scheduling accuracy is suggested as the performance measure (Table 1), as this metric enables the 'Negotiator' to autonomously improve and adapt the rule set over time, increasing the accuracy and efficiency of future EC schedules. At the core of improvement of the agent we envision a reinforcment learning algorithm, as these posses the ability to enhance the rule-set without provision of a detailed one to start with. Initially, this agents task are still to be supervised by a human, until enough data and experience is gathered. Thus, though envisoned as an autonomous scheduling agent in its fully developed system, in its initial state it acts as a recommender.

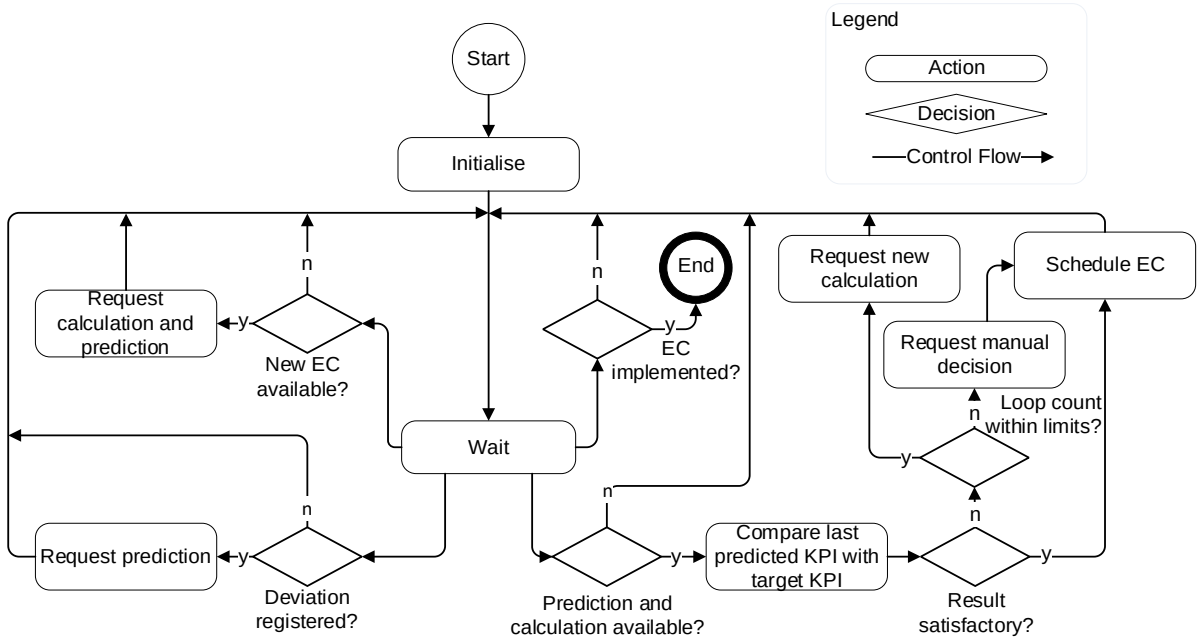

Figure 4. Logic architecture of the 'Negotiator Agent'

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

4.2. Optimiser Agent

The second agent to be introduced is the 'Optimiser Agent' (Figure 5), a model-based reflex agent. It represents the theoretical research streams of EC implementation research and the material planner's task. Accordingly, it addresses BN1 to define a cross-company optimal implementation date. Once initialised, the agent scans through ECs with an open calculation request. Its objective is to search for date optima regarding cost and time. However, before calculating the optima, the change needs to be classified according to its configuration. Afterward, depending on the classification an optimum is calculated. For calculation, a cost function comprised of all costs identified with a certain change type is necessary. For instance, a change due to homologation changes (e.g. resale ban) will have other costs involved than a change due to quality problems (e.g. warranty). Once a global optimum and classification are identified, the information is provided to the 'Negotiator', and the agent returns to an idle state. If an additional calculation request is received, the 'Optimiser' looks for the second-best, or in repeated cases the next best, effectivity date, and returns the information. Upon notice that the EC is scheduled, the 'Optimiser' is terminated, as in case of a full rescheduling of the EC, for the increase of cost and time savings, a full classification-optimisation loop is suggested

On classifying EC, Diprima (1982) and Barzizza *et al.* (2001) each identified three types of changes based on the EC trigger (e.g. immediate and convenient, or rework and scrap). For the classifier we suggest re-evaluating these types, as with more granularity, a better result is possibly achieved. Hence, for improvement of the MAS, additional classifications need to be defined. Furthermore, logistical data, whether parts are delivered in bulk or just-in-sequence, enable different implementation strategies, effectivley increasing the number of change types. Additionally, an investigation of whether rule-based or machine learning (ML) classification proves better is suggested.

For the core of the 'Optimiser' agent, a suitable algorithm is required. This algorithm needs possibilites to test different parameter sets as an actuator. For instance, a change of call-offs, acceptance of obsolescence costs, changing of the amount of goods delivered are possible acutators for the agent to find different optima. To model this search, a clear allocation of product and process specific costs is required. E.g. the above discussed warranty costs are product specific, as the total costs are only dependent on the amount of products built, and not on the change process itself. Furthermore, we suggest developing the actuators as encoded parameters of a genetic algorithm, in combination with a Tabu search to enable the agent to escape previously calculated optima.

Another source of complexity to keep in mind in defining the optimal date are sub-assemblies. For instance, if a quality issue is addressed by changing both the carrier part and the attached part and variants of the attached part exist, achieving zero obsolescence costs is only achieved if all parts have their stock reduced. Contrary, if steered wrongly, production is halted in the worst case due to a mismatch of carrier and attached part.

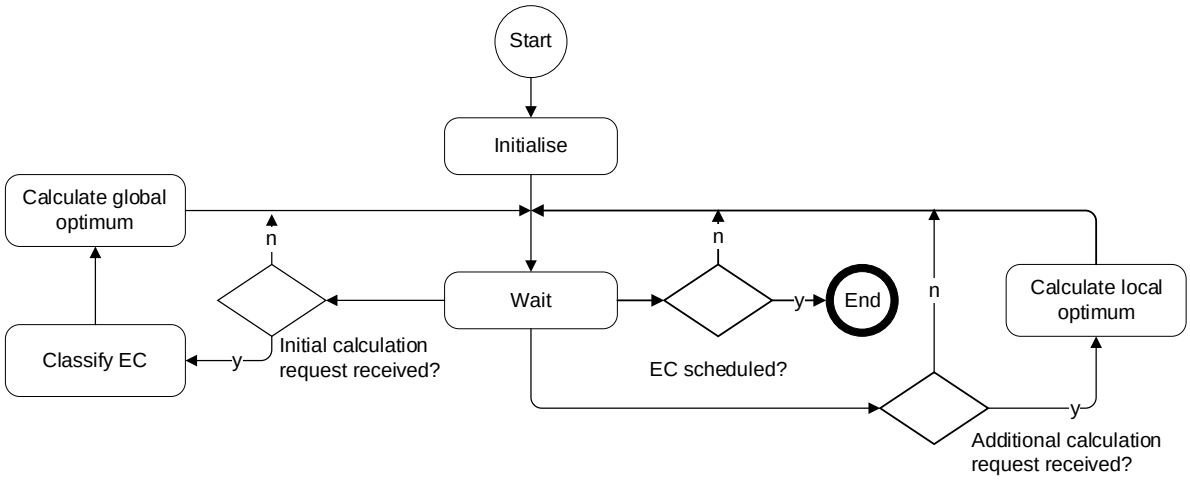

Figure 5. Logic architecture of the 'Optimiser Agent'

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

4.3. Predictor Agent

The 'Predictor' addresses BN2, prediction of EC implementation effects. Therefore, the agent provides the 'Negotiator' with a confidence interval on implementation accuracy and an impact assessment. Its control flow (Figure 6) is rather simple, as upon initialisation the agent is tasked with predicting KPI for all given dates and current EC information. Once the prediction for these KPIs is calculated, the result is returned to be matched against the ruleset of the 'Negotiator'. As an ML-based model, the 'Predictor' is a learning agent, constantly improving its predictions with data from past changes.

From a data perspective, this agent has the highest amount of data consumption. For the prediction algorithm at its core, a supervised ML model is suggested. For this agent to unfold its full potential, data from past changes is necessary as input. To our knowledge, no publication in the past has discussed what data is required for predicting KPI in an EC environment. Additionally, KPI such as 'scheduling accuracy', 'rescheduling amount', 'time to implementation', and others are not defined for the EC process control. Once defined, research is required on what data is best used for predicting these KPIs, and which ML algorithm to use.

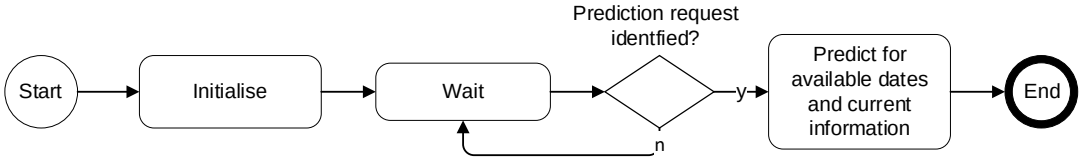

Figure 6. Logic architecture of the 'Predictor Agent'

4.4. Supervisor Agent

The last agent is shown in Figure 7, representing a suggestion for BN4 and thus providing feedback. The 'Supervisor' is a separate entity, with the objective to monitor the EC schedule and inform a human operator and the 'Negotiator' in case any deviations are identified. Separating its objective from the 'Negotiator' enables the strict fulfilment of the respective objectives without interference. Its sensory input includes current and past data of the EC configuration and EC process, giving it the possibility to report the EC implemented notification. Initially, a basic rule set on which deviations are deemed grave enough to necessitate a scheduling check or even a cancellation of the EC should be provided. As it stands, the main difficulty lies in predicting the outcome of a milestone, dependent on the time passed.

It is the only agent that is reliant on a human-machine interface and its performance measure is the number of manual interventions of ECs. The cases with manual intervention can then be used for the agent to learn and improve the deviation ruleset via a reinforcement learning algorithm.

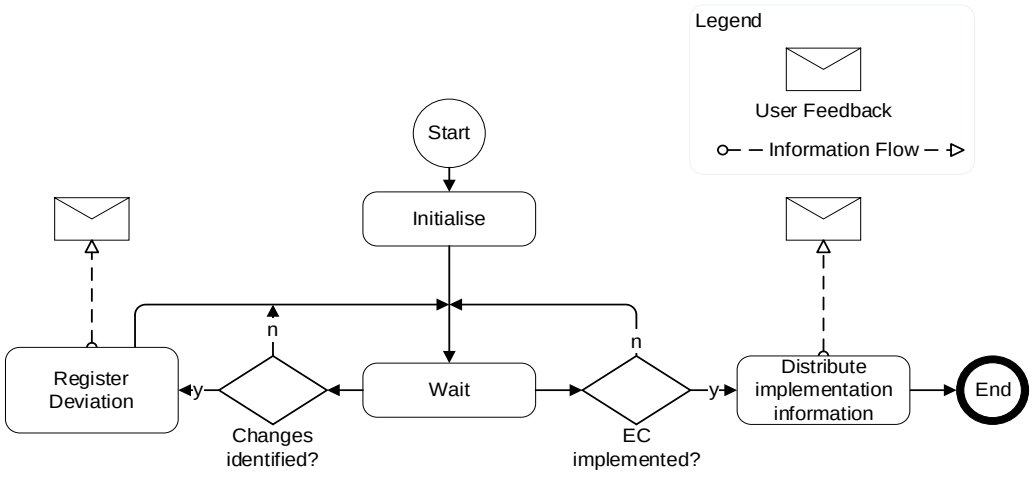

Figure 7. Logic architecture of the 'Supervisor Agent'

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

5. Conclusion and Outlook

We have presented the concept for a potential MAS, designed for automatic EC implementation scheduling. From analysing literature and associated industries, four BN for effective EC scheduling have been defined. These four BN were addressed by introducing four distinct agents. Subsequently, every agent's logic, architecture, and objective were discussed.

As a limitation to the optimisation and automation capabilites of the proposed MAS the focus on the implementation step of the EC process and predefined rules are identified. Due to the late point in the process upon which it becomes active, its optimisation capabilites are within a narrow timeframe. Additionally, within production systems unforseen situations can happen at any time. These result in emergency EC or highly complicated contractual obligations, not represented in data. These changes require fast and flexible decision making and shortcuts within the system, for which the MAS is not designed. Another limitation are responsibilities and accountabilities which need to be decided on a case by case base for the organisation. Especially for legal or safety critical changes, the human should always have the last call.

Once the base MAS is established, further agents could be designed, replicating other tasks. For instance, the 'Negotiator' could also trigger a homologation-specific agent, which is then tasked to check whether any legal testing is necessary. Another issue identified is the product and process approval, as without prior confirmation from the customer, parts should not be sent by the supplier. An agent could thus check whether the change is simple enough to be done without approval checks or even automatically check the documentation provided.

As a research outlook for the MAS, the components of the agents require refinement and investigation. For a starting and testing point, we suggest the tasks of the 'Negotiator' to be performed by human associates and building the other agents first. Research on every actuator is still needed. For instance, the optimising agent needs development regarding classification, as a ruleset-based approach provides limited flexibility to variety. On the goal of prediction, further research regarding learning algorithms and data engineering is needed. As within other research, we suggest investigating which algorithms are suited best for different EC cases and establishing standard datasets for testing. Once a reasonably fit model is trained, it is going to be deployed in a real-world environment to support the human coordinator. In future research, we will test and further develop the logic for each of these agents and deploy the MAS in an industry environment to confirm the applicability through a case study.

References

Arnarsson, I.Ö., Frost, O., Gustavsson, E., Jirstrand, M. and Malmqvist, J. (2021), "Natural language processing methods for knowledge management—Applying document clustering for fast search and grouping of engineering documents", *Concurrent Engineering*, Vol. 29 No. 2, pp. 142–152. https://doi.org/10.1177/1063293X20982973.

Barzizza, R., Caridi, M. and Cigolini, R. (2001), "Engineering change: A theoretical assessment and a case study", *Production Planning & Control*, Vol. 12 No. 7, pp. 717–726. https://doi.org/10.1080/09537280010024054.

Bender, J., Kehl, S. and Müller, J.P. (2015), "A Comparison of Agent-Based Coordination Architecture Variants for Automotive Product Change Management", in Müller, J.P., Ketter, W., Kaminka, G., Wagner, G. and Bulling, N. (Eds.), *Multiagent System Technologies*, *Lecture Notes in Computer Science*, Vol. 9433, Springer International Publishing, Cham, pp. 249–267. https://doi.org/10.1007/978-3-319-27343-3_14.

Beroule, B., Fougeres, A.-J. and Ostrosi, E. (2014), "Engineering change management through consensus seeking by fuzzy agents", in 2014 *Second World Conference on Complex Systems (WCCS), 10.11.2014 - 12.11.2014, Agadir, Morocco*, IEEE, pp. 542–547. https://doi.org/10.1109/icocs.2014.7060920.

Bhuiyan, N., Gatard, G. and Thomson, V. (2006), "Engineering change request management in a new product development process", *European Journal of Innovation Management*, Vol. 9 No. 1, pp. 5–19. https://doi.org/10.1108/14601060610639999.

Camarillo, A., Ríos, J. and Althoff, K.-D. (2017), "Agent Based Framework to Support Manufacturing Problem Solving Integrating Product Lifecycle Management and Case-Based Reasoning", in Ríos, J., Bernard, A., Bouras, A. and Foufou, S. (Eds.), *Product Lifecycle Management and the Industry of the Future*, *IFIP Advances in Information and Communication Technology*, Vol. 517, Springer International Publishing, Cham, pp. 116–128. https://doi.org/10.1007/978-3-319-72905-3_11.

Diprima, M. (1982), "Engineering Change Control and Implementation Considerations", Vol. 23, pp. 81–87.

https://doi.org/10.1017/pds.2022.171 Published online by Cambridge University Press

Eckert, C., Clarkson, P.J. and Zanker, W. (2004), "Change and customisation in complex engineering domains", *Research in Engineering Design*, Vol. 15 No. 1, pp. 1–21. https://doi.org/10.1007/s00163-003-0031-7.
Fricke, E., Gebhard, B., Negele, H. and Igenbergs, E. (2000), "Coping with changes: Causes, findings, and strategies", *Systems Engineering*, Vol. 3 No. 4, pp. 169–179. https://doi.org/10.1002/1520-6858(2000)3:4<169:AID-SYS1>3.0.CO;2-W.
Hamraz, B., Caldwell, N.H.M. and Clarkson, P.J. (2013), "A Holistic Categorization Framework for Literature on Engineering Change Management", *Systems Engineering*, Vol. 16 No. 4, pp. 473–505. https://doi.org/10.1002/sys.21244.
Hevner, Alan R, March, Salvatore T and Sudha (2004), *Design Science in Information Systems Research*, Vol. 28.
Huang, G., Yee, W. and Mak, K. (2003), "Current practice of engineering change management in Hong Kong manufacturing industries", *Journal of Materials Processing Technology*, Vol. 139 No. 1-3, pp. 481–487. https://doi.org/10.1016/S0924-0136(03)00524-7.
Jarratt, T., Clarkson, J. and Eckert, C. (2005), "Engineering change", in Clarkson, J. and Eckert, C. (Eds.), *Design process improvement: A review of current practice*, Springer, London, pp. 262–285. https://doi.org/10.1007/978-1-84628-061-0_11.
Kehl, S., Stiefel, P. and Mueller, J.P. (2015), "Changes on Changes: Towards an Agent-Based Approach for Managing Complexity in Decentralized Product Development", *DS 80-3 Proceedings of the 20th International Conference on Engineering Design (ICED 15) Vol 3: Organisation and Management, Milan, Italy, 27-30.07.15*, pp. 219–228.
Ma, S., Jiang, Z. and Liu, W. (2017), "Multi-variation propagation prediction based on multi-agent system for complex mechanical product design", *Concurrent Engineering*, Vol. 25 No. 4, pp. 316–330. https://doi.org/10.1177/1063293X17708820.
Moon, Y.B. and Wang, B. (2009), "Agent-Based Modeling and Simulation of Resource Allocation in Engineering Change Management", in *Proceedings of the 11th International Conference on Enterprise Information, 06.05.2009 - 10.05.2009, Milan, Italy*, SCITEPRESS - Science and Technology Publications, pp. 281–284. https://doi.org/10.5220/0001852302810284.
Ouertani, M.Z. (2008), "Supporting conflict management in collaborative design: An approach to assess engineering change impacts", *Computers in Industry*, Vol. 59 No. 9, pp. 882–893. https://doi.org/10.1016/j.compind.2008.07.010.
Potdar, P. and Jonnalagedda, V. (2018), "Design and development of a framework for effective engineering change management in manufacturing industries", *International Journal of Product Lifecycle Management*, Vol. 11 No. 4, p. 368. https://doi.org/10.1504/ijplm.2018.097880.
Radisic-Aberger, O. (2021), Engineering Change Management - Classification *Appendix of* Literature Review. https://doi.org/10.17632/VNS2KP3ZT3.1.
Russell, S.J. and Norvig, P. (2016), *Artificial intelligence: A modern approach, Always learning*, Third edition, Global edition, Pearson, Boston, Columbus, Indianapolis.
Sharp, M.E., Hedberg, T.D., Bernstein, W.Z. and Kwon, S. (2021), "Feasibility study for an automated engineering change process", *International Journal of Production Research*, Vol. 59 No. 16, pp. 4995–5010. https://doi.org/10.1080/00207543.2021.1893900.
Shivankar, S.D., Kakandikar, G.M. and Nandedkar, V.M. (2015), "Implementing engineering change management through product life cycle management in automotive field", *International Journal of Product Lifecycle Management*, Vol. 8 No. 2, p. 132. https://doi.org/10.1504/ijplm.2015.070579.
Wänström, C., Lind, F. and Wintertidh, O. (2006), "Creating a model to facilitate the allocation of materials planning resources in engineering change situations", *International Journal of Production Research*, Vol. 44 No. 18-19, pp. 3775–3796. https://doi.org/10.1080/00207540600622506.
Weißer, T., Wagner, J., Burggräf, P. and Lichtenwalter, D. (2021), "Support-vector classification of downstream problem effects during physical product development and ramp-up", *Procedia CIRP*, Vol. 99, pp. 621–626. https://doi.org/10.1016/j.procir.2021.03.084.
Wooldridge, M., Jennings, N.R. and Kinny, D. (2000), "The Gaia Methodology for Agent-Oriented Analysis and Design", *Autonomous Agents and Multi-Agent Systems*, Vol. 3 No. 3, pp. 285–312. https://doi.org/10.1023/A:1010071910869.