AIDA: A Tool for Resiliency in Smart Manufacturing

Giuseppe De Giacomo[1,2], Marco Favorito[3], Francesco Leotta[1], Massimo Mecella[1], Flavia Monti[1](✉), and Luciana Silo[1]

[1] Sapienza Università di Roma, Rome, Italy
{degiacomo,leotta,mecella,monti,silo}@diag.uniroma1.it
[2] University of Oxford, Oxford, UK
[3] Banca d'Italia, Rome, Italy
marco.favorito@bancaditalia.it

Abstract. One of the salient features of Industry 4.0 is that machines and other actors involved in the manufacturing process provide Industrial APIs that allow to inquire their status. In order to provide resilience, the manufacturing process should be able to automatically adapt to new conditions, considering new actors for the fulfillment of the manufacturing goals. As a single manufacturing process may include several of these actors, and their interfaces are often complex, this task cannot be easily accomplished in a completely manual way. In this work, we focus on the orchestration of Industrial APIs using Markov Decision Processes (MDPs). We present a tool implementing stochastic composition of processes and we demonstrate it in an Industry 4.0 scenario.

Keywords: industrial API · smart manufacturing · service composition

1 Introduction

The term Industry 4.0 refers to the emergence and diffusion of new technologies which allow the development of fully automatized production processes [16]. *Smart manufacturing* is nowadays a term highly used in conjunction with the concept of Industry 4.0; it aims at improving the manufacturing processes in order to increase productivity and quality, make workers' lives easier, and define new business opportunities. This is enabled by leveraging on innovative techniques like Artificial Intelligence (AI), big data analytics and Process Mining (PM). The adoption of such techniques enables the advent of AI-augmented Business Process Management Systems (ABPMSs), an emerging class of process-aware information systems [9]. Such a trend has made it possible to create new opportunities for interoperability, modularity, distributed processing, and integration in real-time with other systems for industrial processes.

One of the main characteristics of Industry 4.0 is that actors involved in the manufacturing process (e.g., machines, humans) provide Application Program-

M. Favorito—The author's views are his own, and they do not reflect those of his employer.

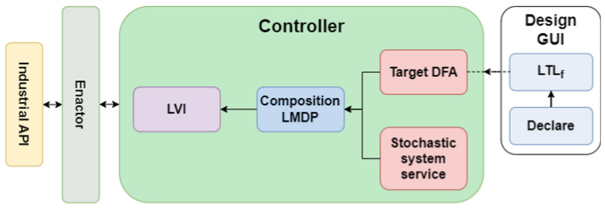

Fig. 1. Architecture of AIDA tool

ming Interfaces (APIs) which allow to collect their status [4] and to operate on them. The term status here does not only encompass the current situation of the actor, but also information resulting from the application of diagnostic, predictive and prescriptive analytics models. A single manufacturing process may include tens of different actors along the supply chain [2] which may suddenly fail or provides bad performance. At any moment, in order to provide resilience, a manufacturing process should be able to automatically adapt to new conditions, considering new actors for the fulfillment of the manufacturing goals. This task cannot be done manually when actors span multiple organizations possibly separated from both the geographical and organizational points of view.

In this paper, we propose a tool that generates a plan for a manufacturing process. Manufacturing actors are depicted as services (Industrial APIs) that reflect their behaviors. Particularly, we employ the generalization of the service composition approach in a stochastic setting [3], in which the services have an unpredictable behavior and are subject to wear. In our tool, instead of representing the manufacturing process as a target stochastic service, we use the well-known formalism DECLARE, widely used in the Business Process Management literature. Another important contribution that we provide is to achieve an optimal solution for the orchestration by solving a probabilistic planning problem formalized as Lexicographic Markov Decision Process (LMDP), which permits to take into account not only the breaking probability of a machine but also a scenario of multiple objectives. In this way, we are able to *autonomously* obtain a production planning which is *adaptive*, as it changes every time that the manufacturing of a new product or batch is started, and *context-aware*, as it depends on the current status of involved actors. Our tool is implemented in a software prototype, which will be showcased in a scenario concerning an electric motor manufacturing process derived from a real industrial project.

2 Tool Architecture

Figure 1 depicts the architecture of our tool AIDA - Adaptive InDustrial APIs[1].

We implement a *service-oriented* approach for industrial manufacturing to enable the interoperability between the actors. Particularly, we model each actor involved in the manufacturing process as a service, thus creating a service-based supply chain consisting of a composition of services (representing actors). Such

[1] Aida is also the name of a famous opera by Verdi, somehow inspired by the making of Suez Canal: undoubtedly an example of smart manufacturing for the time being.

services realized as *Industrial APIs*, represent the physical actors describing their functionalities (or tasks). Additionally, they allow to monitor the behavior and status information of the actors and allow to invoke commands.

We define the actors as *stochastic services* modeled as *Markov Decision Processes (MDPs)* [17]. An MDP is a discrete-time stochastic control process containing *(i)* a set Σ of states which represent the status of the service, *(ii)* a set A of actions i.e., the set of tasks that the service can perform, *(iii)* a transition function P that returns for every state s and action a a distribution over the next state i.e., the probability of the service to end in a certain status performing a certain task, *(iv)* a reward function R that specifies the reward when transitioning from state s to state s' by executing action a, and *(v)* a discount factor $\lambda \in (0, 1)$ which determines how important future rewards are to the current state. If $\lambda = 0$, the service is "myopic" in being concerned only with maximizing immediate rewards. As λ reaches 1, the method becomes more "farsighted", more strongly considering future rewards.

Particularly we are interested in LMDPs (Lexicographic MDP) [22] in which the reward function is a vector of reward functions. This vector is formed by two objectives: the cost and the quality of the product; and our goal is to minimize the former and maximize the latter. An optimal solution to an LMDP is a policy ρ^* which assigns an action to each state and maximizes the expected cumulative reward, i.e., the sum of discounted rewards when starting at state s and choosing actions based on ρ following a lexicographic preference.

We describe the behavior of each actor as a state machine with a probabilistic behavior represented as an MDP and maintained by the corresponding Industrial API. The latter contains the set of transitions (states, actions, probabilities and costs) that an actor is able to perform and the information relative to the initial, final and current state. Each actor may include a rich set of states (i.e., READY, CONFIGURATION, EXECUTING, BROKEN, REPAIRING) and actions or only a subset. Different actors can offer the same operation. As a consequence, an actor chosen for a specific process instance could be discarded for the later instance. The Industrial APIs expose endpoints to retrieve their information which are combined to construct a community of stochastic services, i.e., a *stochastic system service*. Intuitively, the stochastic system service status includes the current status of all the composing services, and a specific action performed on the system service changes only one component of the current state, corresponding to the service selected to execute that action.

Among others, we define the manufacturing process specification as DECLARE constraints, i.e., LTL$_f$ formula φ [8] over the set of propositions $\mathcal{P}$ that specifies the allowed traces of the process. We allow the potential production process engineer to specify the process (in a canvas as in Fig. 2) via the *design GUI*. Note that the collection of services representing the actors can perform actions in $\mathcal{P}$ and, moreover, to make our model richer we allow services to execute a broader set of actions. Moreover, we put each LTL$_f$ formula in conjunction in order to compute the equivalent deterministic finite automaton (DFA) (made by `Lydia` tool [6]), i.e., *target* DFA.

Given both the stochastic system service and the target DFA, we compute the *composition LMDP* as a function that contains: all the states of the target DFA and the stochastic system service, all the actions of the services, the probability of ending in a certain system state performing an action, and the vector of reward functions formed by the objectives that we want to consider in our application. In practice, it consists of a cartesian product operation of both the system and target services. According to a specific target (manufacturing goal), it computes all the possible executions of the manufacturing process, i.e., by combining together the specifications of all the actors (stochastic services) and the goal, it identifies all the possible status of the actors at any step.

We extract the optimal policy of the composition LMDP by executing the *Lexicographic Value Iteration (LVI)* Algorithm [22]. Such a policy contains the specification of the optimal actions (and related services) to execute from each possible state in order to reach the final goal.

The *enactor* acts as a middleware that interfaces with the Industrial APIs in order to check whether the current status and the transition functions have changed (for instance because of the wearing out during the execution). We distinguish two different resilience scenarios. On the one hand, when only the status of an actor changes, the *controller* is able to choose the next action to be performed by checking the result of the optimal policy from the new state formed. On the other hand, when both the status and the transition function of an actor change, the *controller* re-computes the optimal policy from an up-to-date composition LMDP which includes the latest condition of the service. Through the Industrial APIs, the *enactor* calls the services identified in the optimal policy computed by the *controller*.

3 Demonstrating the AIDA Tool

A freely available tool[2] has been implemented. The tool can be configured to prioritize either cost or quality. This is particularly helpful in a real industrial context, as a company may prefer to reduce costs (e.g., for timing or pecuniary reasons) or, on the contrary, to maximize quality. Noticeably, the decision of the priority can be possibly seen as a customer decision in the case of so called mass-customization.

To demonstrate the proposed tool, we use the manufacturing process of an electric motor which is depicted in Fig. 2 using the DECLARE formalism [15]. For the sake of brevity, we focus on the main aspects of the process, but the formalization can be easily extended to cover the process much more in detail.

The main components of an electric motor are the stator, the rotor and, in the case of alternate current motors with direct current power, an inverter. These three components are built or retrieved in any order and then eventually assembled to build a motor (*alternate succession* constraint). After the motor is assembled, a Running In test must be performed (*alternate succession* constraint), and, optionally (*alternate precedence* constraint), at most one

[2] See sources at https://github.com/luusi/AIDA.

(*not coexistence* constraint) between an Electric Test and a full Static Test (the latter comprises the former). In addition, optionally (*alternate precedence* constraint), the motor can be painted. The process depicts the manufacturing tasks involved in a production of a *single motor*.

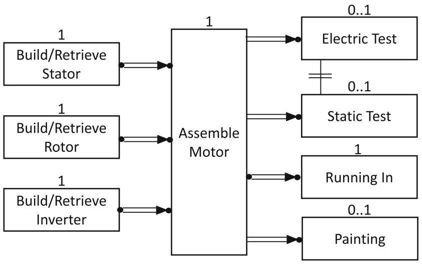

Fig. 2. The electric motor manufacturing process represented using DECLARE.

We want to find a plan that fulfills the manufacturing process, selecting appropriate actors for the actions considering the cost and quality impact of choices.

The repository contains a Python script that reproduces the case study. The script shows how the goal is achieved also in case of malfunctioning and/or exceptions raised by the Industrial APIs. For each of the services we have several instances, each showing a specific wear and reward vector. In the following, we will describe a possible execution obtained by manually choosing the values of wear and reward vectors. In a real application scenario, choices are made accordingly to the real conditions as reported by each actor digital twin.

The example manufacturing process specification allows the three *Build/Retrieve* actions to be executed in any order. As an example, the first *Build Retrieve Stator* action may be executed first, and the tool may decide to choose STATOR BUILDER service to execute the action in case the warehouse machine is more costly.

Particularly interesting is the case when an actor shows a significantly high wear. For example, the execution of the *Build Retrieve Rotor* action, can be performed, in the proposed demo, by two different machines ROTOR BUILDER 1 and ROTOR BUILDER 2, or by the ROTOR WAREHOUSE service. If the first two machines have a higher probability to end in a BROKEN state, the optimal policy will prefer the ROTOR WAREHOUSE SERVICE, because it has no possibility to break. Since the *Build Retrieve Inverter* can be performed only by the INVERTER WAREHOUSE service, in this case the choice of the tool is immediate.

At this point, the *Assemble Motor* action can be performed. Here, the controller may prefer to use the ASSEMBLER MACHINE 1 instead of using the ASSEMBLER MACHINE 2, even if they have the same cost, if the first machine has a higher quality reward.

Concerning the *Painting* action, we suppose it is offered by a machine and by a human, and we may assume that the human is more economic but gives a lower quality compared to a machine. In this case, the controller would prefer using PAINTER HUMAN service because it has a lower economic cost than the PAINTER MACHINE. Finally, for the *Smart Tester* action, the controller may prefer to use SMART TESTER 2 if it has a lower economic cost than SMART TESTER 1.

Noticeably, we have defined a tool that computes not only the right action according to several constraints but is also very resilient. Suppose that some machines end in the BROKEN state; the controller may decide to restore, for example, the STATOR BUILDER SERVICE rather than directly using the STATOR WAREHOUSE which could perform the same action at a much higher cost.

4 Related Works

The tool that we implement proposes a plan for smart manufacturing that is *resilient*. Resilience concerns the ability of a system to cope with unplanned situations in order to keep carrying out its mission. The research on resilient systems is extensively studied [23,24] mainly at a conceptual level, and continuously improved especially at design time [13]. In AIDA, we ensure resilience at run time as, when a service during its job breaks down and/or starts to have a high cost and a low quality level, we take into account all these aspects and the controller is able to adapt the plan avoiding possible not convenient services. The composition is possible thanks to the use of the Industrial APIs representing the manufacturing actors which provide lots of features like accessing the selected services [12], enabling quick integration [10] and monitoring their behaviour [19].

The implemented approach is influenced by previous works on automated service composition. Authors in [3], for example, propose a solution for the service composition in stochastic settings, defining the non-deterministic behavior of the target service. The authors though do not capture the non-deterministic behaviors of the available services, and do not take into account the rewards of using a certain service.

The problem we addressed also belongs to the area of decision-theoretic planning applied to manufacturing. Such a topic is part of the greater problem of AI planning. Different approaches are employed in planning problems, i.e., classical planning, dealing with deterministic contexts, and decision-theoretic planning, facing with non-deterministic and stochastic scenarios. Multiple works can be found in the literature employing classical planning techniques [14,21], however they do not take into account the stochasticity of manufacturing. The literature presents limited research on the application of MDPs in the manufacturing domain. Authors in [11] propose an MDP-based self-adaptive Automated Guided Vehicles (AGVs) control model that avoids collisions efficiently. The work [5] presents a hierarchical MDP approach for adaptive multi-scale prognostics and health management, maximizing the expected gain. Authors in [1] use an MDP for finding an optimal cost-effective maintenance decision based on the condition revealed at the time of inspection on a single diesel engine.

The proposed tool represents an evolution of what presented in [7]. In the original work though, the target process is represented using a state machine, thus not allowing for the flexibility which is typical of certain manufacturing processes. In addition, the modeling formalism does not allow for multiple objectives. The paper represents a useful reference anyway to deepen some of the concepts behing the AIDA tool.

Another important aspect in manufacturing environments is the maintenance efficiency, where asset maintenance and repair significantly contribute to operation and support costs. In real systems, maintenance is complicated making it necessary to use many optimization criteria. The majority of research works do not adopt a data-driven approach to decision-making, and thus they are limited to specific problems and domains [18,20]. That is why mathematical optimization and rule-based systems are the most common categories of methods.

5 Concluding Remarks

In this work, we proposed a tool implementing a stochastic service composition approach with LTL$_f$ goals, where composing services have a stochastic behavior and are modeled as MDPs. The goal is to obtain an optimal policy with respect to a set of reward measures expressed as a vector with priorities. In particular, we have realized a demo showing how such a tool can be helpful in an industrial manufacturing context where DECLARE models a manufacturing process, while the availability of Industrial APIs enables the collection of information of the involved actors. Especially, rewards and probabilities associated with each service can be continuously updated by applying, for instance, predictive maintenance algorithms to monitor the status of the different involved actors, thus allowing at *each repetition of the process* to choose the most suitable actors to perform actions. This makes the process resilient to failures and optimal with respect to defined reward measures. In our demo tool, we adopt a simulator mimicking the evolution of the single actors (services) from the point of view of rewards and failure probabilities, showing how changes influence proposed execution traces. This paper impacts anyway application scenarios other than Industry 4.0, being applicable in any context where actors can be modeled as services. Finally, in this paper, we do not consider data; adding data introduces new challenges, as specific traces of the process, legal from the point of view of the control flow, might be not doable in practice. Future works include considering conditions expressed on service or process data.

Acknowledgements. This work is partially funded by the ERC project WhiteMech (no. 834228), the PRIN project RIPER (no. 20203FFYLK), the Electrospindle 4.0 project (funded by MISE, Italy, no. F/160038/01-04/X41). This study was carried out within the PE1 - FAIR (Future Artificial Intelligence Research) and PE11 - MICS (Made in Italy - Circular and Sustainable) - European Union Next-Generation-EU (Piano Nazionale di Ripresa e Resilienza - PNRR). The work of Flavia Monti is supported by the MISE agreement on “Promozione del progetto della Scuola europea di industrial engineering and management e il sostegno di progetti innovativi di formazione in industrial engineering e management di impresa”.

References

1. Amari, S.V., McLaughlin, L., Pham, H.: Cost-effective condition-based maintenance using Markov decision processes. In: RAMS, pp. 464–469. IEEE (2006)
2. Bicocchi, N., Cabri, G., Mandreoli, F., Mecella, M.: Dynamic digital factories for agile supply chains: an architectural approach. J. Ind. Inf. Integr. **15**, 111–121 (2019)
3. Brafman, R.I., De Giacomo, G., Mecella, M., Sardina, S.: Service composition in stochastic settings. In: Esposito, F., Basili, R., Ferilli, S., Lisi, F. (eds.) AIxIA 2017. ecture Notes in Computer Science, vol. 10640, pp. 159–171. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-70169-1_12
4. Catarci, T., Firmani, D., Leotta, F., Mandreoli, F., Mecella, M., Sapio, F.: A conceptual architecture and model for smart manufacturing relying on service-based digital twins. In: IEEE ICWS, pp. 229–236 (2019)
5. Choo, B.Y., Adams, S.C., Weiss, B.A., Marvel, J.A., Beling, P.A.: Adaptive multi-scale prognostics and health management for smart manufacturing systems. Int. J. Prognostics Health Manage. **7** (2016)
6. De Giacomo, G., Favorito, M.: Compositional approach to translate LTLf/LDLf into deterministic finite automata. In: ICAPS, pp. 122–130. AAAI Press (2021)
7. De Giacomo, G., Favorito, M., Leotta, F., Mecella, M., Silo, L.: Digital twins composition in smart manufacturing via Markov decision processes. Comput. Ind. **149**, 103916 (2023)
8. De Giacomo, G., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In: IJCAI, pp. 854–860. ACM (2013)
9. Dumas, M., et al.: AI-augmented business process management systems: a research manifesto. ACM Trans. Manage. Inf. Syst. **14**(1), 1–19 (2023)
10. Han, H., Trimi, S.: Towards a data science platform for improving SME collaboration through Industry 4.0 technologies. Technol. Forecast. Soc. Change **174**, 121242 (2022)
11. Hu, H., Jia, X., Liu, K., Sun, B.: Self-adaptive traffic control model with behavior trees and reinforcement learning for AGV in industry 4.0. IEEE Trans. Ind. Inf. **17**(12), 7968–7979 (2021)
12. Liu, Z., et al.: The architectural design and implementation of a digital platform for industry 4.0 SME collaboration. Comput. Ind. **138**, 103623 (2022)
13. Marrella, A., Mecella, M., Pernici, B., Plebani, P.: A design-time data-centric maturity model for assessing resilience in multi-party business processes. Inf. Syst. **86**, 62–78 (2019)
14. Marrella, A., Mecella, M., Sardina, S.: SmartPM: an adaptive process management system through situation calculus, IndiGolog, and classical planning. In: KR (2014)
15. Pesic, M., Schonenberg, H., Van der Aalst, W.M.: Declare: Full support for loosely-structured processes. In: EDOC, pp. 287–287. IEEE (2007)
16. Popkova, E.G., Ragulina, Y.V., Bogoviz, A.V. (eds.): Industry 4.0: Industrial Revolution of the 21st Century. SSDC, vol. 169. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-94310-7
17. Puterman, M.L.: Markov Decision Processes. Wiley, Hoboken (1994)
18. Rocchetta, R., Bellani, L., Compare, M., Zio, E., Patelli, E.: A reinforcement learning framework for optimal operation and maintenance of power grids. Appl. Energy **241**, 291–301 (2019)
19. Sahal, R., Breslin, J.G., Ali, M.I.: Big data and stream processing platforms for Industry 4.0 requirements mapping for a predictive maintenance use case. J. Manuf. Syst. **54**, 138–151 (2020)

20. Terkaj, W., Tolio, T., Urgo, M.: A virtual factory approach for in situ simulation to support production and maintenance planning. CIRP Ann. **64**(1), 451–454 (2015)
21. Wally, B., et al.: Leveraging iterative plan refinement for reactive smart manufacturing systems. IEEE Trans. Autom. Sci. Eng. **18**, 230–243 (2020)
22. Wray, K.H., Zilberstein, S., Mouaddib, A.I.: Multi-objective MDPs with conditional lexicographic reward preferences. In: AAAI (2015)
23. Zahoransky, R.M., Brenig, C., Koslowski, T.: Towards a process-centered resilience framework. In: ARES, pp. 266–273. IEEE (2015)
24. Zahoransky, R.M., Koslowski, T., Accorsi, R.: Toward resilience assessment in business process architectures. In: Bondavalli, A., Ceccarelli, A., Ortmeier, F. (eds.) SAFECOMP 2014. LNCS, vol. 8696, pp. 360–370. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10557-4_39