BedreFlyt Artifact: A Self-Adaptive Digital Twin Software System for Hospital Resource Allocation

Riccardo Sieve
Department of Informatics
University of Oslo
Oslo, Norway
riccasi@ifi.uio.no

Nelly Bencomo
Computer Science Department
Durham University
Durham, United Kingdom
nelly.bencomo@durham.ac.uk

Einar Broch Johnsen
Department of Informatics
University of Oslo
Oslo, Norway
einarj@ifi.uio.no

Silvia Lizeth Tapia Tarifa
Department of Informatics
University of Oslo
Oslo, Norway
sltarifa@ifi.uio.no

***Abstract*—Socio-technical systems can benefit from self-adaptive digital twins that combine runtime models, simulation, formal reasoning, and operational data to support continuous monitoring, analysis, and runtime decision making. This paper presents the BEDREFLYT *Artifact*, a reusable software system implementing a self-adaptive digital twin for hospital resource allocation. The artifact integrates semantic reflection, lifecycle management, simulation, optimisation, and a human-in-the-loop dashboard within a modular architecture. Together, these components maintain a runtime representation of the hospital and generate bed bay allocation recommendations under changing operational conditions.**

The paper describes the architecture, deployment, configuration, and execution of the artifact, and illustrates its use for hospital bed bay allocation across multiple wards through both allocation and simulation workflows. The BEDREFLYT *Artifact* provides an executable and reusable software exemplar for reproducing, evaluating, and extending research on self-adaptive digital twins and runtime decision support in hospital resource allocation.

***Index Terms*—Self-Adaptive Systems, Digital Twins, Software Artifacts, Runtime Models, Hospital Resource Allocation**

I. MOTIVATION AND SIGNIFICANCE

A digital twin (DT) can be seen as a dynamic virtual representation of a target system that integrates models and real-time data continuously updated from, e.g., sensors, operational systems, or historical records to drive decision making and provide decision support to humans [1], [2]. This enables organisations to monitor, analyse, and plan operations before decisions are manually or autonomously executed.

While DTs stem from engineering disciplines [3], they are now increasingly used in other domains such as manufacturing [4], transportation [5], and healthcare [6]. In hospital management, *bed bay allocation* directly affects patient waiting times and resource usage at the hospital [7]. However, the allocation process is complex due to factors such as patient needs, bed availability, and hospital policies. Moreover, this process is still largely performed manually by admitting nurses, which can lead to inefficiencies and errors.

This paper presents the BEDREFLYT *Artifact*, a reusable software system implementing a self-adaptive DT. This work builds upon previous research on the BEDREFLYT DT [8]–[10] for hospital resource allocation. The artifact leverages semantic reflection [11] and formal models to generate bed bay allocation recommendations while adapting to changing operational conditions, such as changes in the layout of the wards in the hospital, e.g., due to maintenance in rooms, or fluctuations in patient admissions and discharges. By utilising a DT approach, BEDREFLYT can simulate different operational scenarios and predict the outcomes of alternative allocation strategies, allowing hospital staff to make informed decisions. The artifact is publicly available and is intended to support the evaluation, reuse, and extension of self-adaptive DTs for hospital resource allocation.

The BEDREFLYT *Artifact* provides decision support to hospital staff by allowing a human-in-the-loop to interact with the self-adaptive DT, review recommendations, and make informed decisions based on the DT's analysis. The DT continuously adapts to new data and feedback from the human-in-the-loop, ensuring that the bed allocation process remains efficient and responsive to changing conditions. In BEDREFLYT, self-adaptation stems from the input provided by the hospital staff, which allows the DT to adjust its recommendations and strategies. This integration of human expertise and automated decision-making enhances the overall effectiveness of the bed allocation process in the DT. Thus, the DT can be seen as an integral part of a self-adaptive system that encompasses the hospital's operations and decision-making processes, whereas the hospital is the self-adaptive system that benefits from the DT's capabilities. The main contributions of this paper are:

- The BEDREFLYT *Artifact*, a reusable software system implementing a self-adaptive DT through an architecture integrating semantic reflection, formal models, simulation, optimisation, and a human-in-the-loop dashboard.
- A reusable evaluation setup, including hospital configurations and simulation scenarios for demonstrating and evaluating configurations of the BEDREFLYT *Artifact*.

II. RELATED WORK

Within the healthcare domain, DTs have proven effective in improving patient care and predicting diseases [12]. Supported by the increasing availability of data and advancements in machine learning and AI, DTs have a significant potential to improve personalised medicine and healthcare [13], [14]. Furthermore, DTs can be used to simulate and predict the

progression of diseases and provide strategies for allocation of resources for personal healthcare [15].

However, the use of DTs in healthcare can be challenged by the need for large amounts of data, which may be unavailable or of poor quality. Additionally, DTs need to process data in a meaningful way to provide accurate predictions and insights [16]. Moreover, the use of DTs in healthcare needs to be transparent, accountable, and respect patients' rights and privacy [17]. Self-adaptation can be used to enable DTs to adapt and evolve over time, allowing them to better meet these needs [18], [19]. Self-adaptive systems can also help to identify faults and anomalies [20], thereby enhancing the safety and reliability of DTs in healthcare.

DTs have been applied to patient care and healthcare resource allocation. However, supporting continuous adaptation to changing operational conditions remains comparatively underexplored. The BEDREFLYT *Artifact* provides an executable implementation of a self-adaptive DT that combines semantic reflection and penalty-based optimisation to support runtime decision making for hospital resource allocation.

III. THE SOFTWARE DESCRIPTION OF BEDREFLYT

The BEDREFLYT *Artifact* consists of several software components that together implement a self-adaptive DT for hospital resource allocation. The artifact provides runtime decision support for bed bay allocation based on the current hospital conditions and patient needs. The real system modelled by the artifact is a hospital consisting of multiple wards, treatment rooms, and bed bays. The DT maintains a runtime representation of this information together with dynamic data, including patient admissions and discharges, bed bay availability, and similar relevant operational information.

Figure 1 depicts the MAPE-K-inspired runtime workflow implemented by the BEDREFLYT *Artifact*. Its main software components interact during bed bay allocation, from monitoring the hospital environment to generating allocation recommendations that support decision-making for the hospital staff. The process is initiated by the admitting nurse, who interacts with the DT through a *Dashboard*, and requests bed bay allocations for incoming patients. The DT processes these requests by collecting data from the hospital environment through *SMOL* [21], [22], which constructs a runtime knowledge base (KB) from the domain ontology by applying semantic lifting [22], [23] to DT's runtime state. Using this enriched KB, the *Lifecycle Manager* checks that the wards align with the current hospital conditions. Otherwise, the *Z3* optimiser [24] computes new room configurations, and the structure of the hospital is updated accordingly. The DT retrieves patient information from the *Database* and computes patient needs using the actor-based simulation language ABS [25]. Finally, the *Z3* optimiser solves the bed bay allocation problem, and the recommended allocation is presented through the *Dashboard*, where the admitting nurse makes the final decision. The MAPE-K workflow consists of the following overall elements (where the associated container name is given in parenthesis):

a) Orchestrator (bf-api): The *Orchestrator* coordinates the interactions between the main components of the BEDREFLYT Artifact. It manages the flow of information during the allocation process, from the request initiated through the *Dashboard* to the execution of the reasoning components. It is responsible for initiating the allocation process, collecting the required data, and coordinating the execution of the other components. It is further composed of following subcomponents:

- The *SMOL* component is responsible for constructing the runtime KB from the ontology. It aligns the KB with the current hospital conditions by applying semantic lifting [11], thereby enriching the runtime data used by the other components of the artifact.
- The *ABS simulation* component is responsible for computing expected patient room requirements from patient diagnosis information. This information includes the list of tasks that each patient has to undergo and the room requirements associated with those tasks, so that the hospital has a collective estimation of the required rooms along the timeline of the expected stay of each patient.

The computed patient needs are then enriched with constraints such as isolation of a patient or rooms with special equipment, which influence the bed bay allocation decisions.

b) Lifecycle Manager (bf-lm): The *Lifecycle Manager* analyses the information captured by the *Orchestrator* and identifies changes in the state of the system [26], [27]. In BEDREFLYT, this corresponds to the state of the wards in the hospital. Based on such information, the DT is kept automatically aligned with the real system. It checks whether the current hospital conditions require changes in the ward structure; e.g., when fluctuations in patient admissions or discharges require additional rooms to be opened or closed, the DT gets updated accordingly.

c) Z3 Optimiser (bf-solver): The *Z3 Optimiser* solves the allocation problem by taking into account the input from the orchestrator, which includes current hospital conditions, patient room requirements, and allocation constraints. It plans the bed bay allocation strategy that aims to maximise resource utilisation while preserving patient care quality, e.g., avoiding unnecessary movement of patients.

d) Dashboard (bf-app): The *Dashboard* is the point of contact for the human-in-the-loop to accept or reject the bed bay allocations proposed by BEDREFLYT. The dashboard triggers the allocation process depending on the input from the admitting nurse (it supports the execution step in the MAPE-K loop).

IV. CONFIGURING AND USING BEDREFLYT

We now discuss how to configure and use the BEDREFLYT *Artifact*, and describe how to instantiate the artifact, its required input data, and the mechanisms to deploy and reuse the artifact across different hospital configurations.

A. *Instantiating* BEDREFLYT

BEDREFLYT can be instantiated by means of a provided Docker compose file. This file contains all the necessary

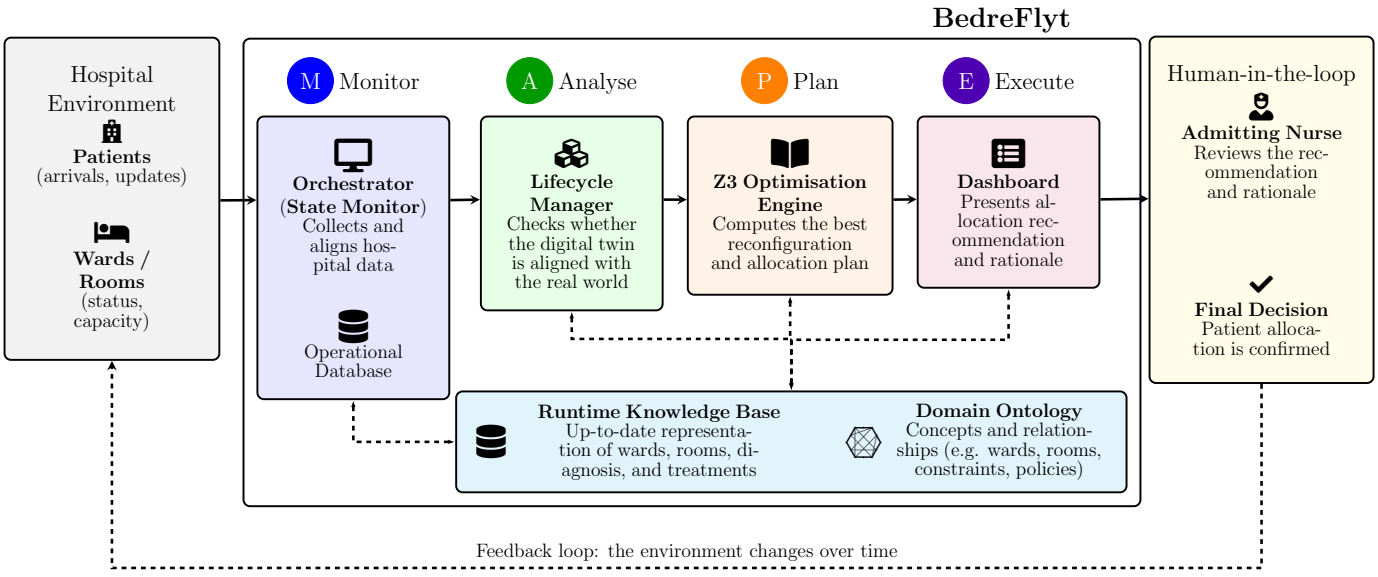

Fig. 1: BEDREFLYT runtime workflow with self-adaptive decision support for patient allocation

components to run the artifact, including the web server, database, and any other required services. By running the Docker compose command, users can quickly set up a local instance of BEDREFLYT. The different components of the artifact are configured to work together through Dockerfiles, allowing users to focus on using the artifact rather than worrying about the underlying infrastructure.

The required information to instantiate BEDREFLYT is provided in the artifact's documentation, including step-by-step instructions for setting up the Docker environment and running the artifact. The documentation also explains how to configure the artifact for different hospital settings, including how to input data and adjust system parameters. Moreover, the compose and environment files allow users to easily modify the configuration of the artifact for their specific needs, such as changing the database connection settings, adjusting the web server configuration, and modifying other system parameters.

The artifact includes a sample ontology file that can be used to test the artifact's functionality. This ontology includes information about a hypothetical hospital, including its structure and policies. The artifact further includes a sample dataset with synthetic patient data that can be used to test the bed bay allocation process. This dataset is designed to be representative of a typical hospital environment, allowing users to try the artifact's features and understand its functionality before inputting their own data.

B. Inputting Data

To use BEDREFLYT, hospital staff can input new patient data through the *Dashboard*. This includes information such as the patient's name, age, and current condition. Furthermore, while not yet integrated in the *Dashboard*, the system already has the logic needed to adapt to changes in the structure of the hospital, such as the addition of new wards or rooms, or changes in resource availability [10]. It can also accept information regarding changes in treatments and diagnosis, which would prompt an adaptation of the DT to better reflect the current state of the hospital.

V. EXAMPLE USE CASE

We illustrate the use of the BEDREFLYT *Artifact* through a representative hospital scenario with multiple wards spread across different floors of a hospital, as illustrated in Fig. 2. Each ward contains a number of bed bays. Patients are allocated to bed bays based on their needs and the bed bay availability. The hospital structure can accommodate different types of patients, including patients with specific requirements such as isolation rooms or rooms with special equipment.

The bed bay allocation process in this hospital involves several steps. When a patient arrives at the hospital, the admitting nurse interacts with the *Dashboard* to request a bed bay. The DT then collects data from the hospital environment, including current bed bay availability and patient information, and computes a recommended bed bay allocation while accounting for patient needs, bed bay availability, and hospital policies. For the allocation policy, we consider the gender of the patient, the type of care they require, and whether or not they are contagious, to ensure that patients are allocated to appropriate bed bays while also optimising resource utilisation and patient care quality, e.g., by minimizing the movement of patients. Finally, the recommended bed bay allocation is presented to the admitting nurse through the *Dashboard*, allowing the admitting nurse to make the final decision on where to allocate and reallocate the patients in the hospital.

The bed bay allocation process in BEDREFLYT is encoded as a penalty-guided optimisation problem that finds an optimal bed bay allocation for a given set of patients, where rooms have associated penalties, detailed in [8]–[10]. The encoding of the optimisation problem includes constraints related to room capacity, gender of patients, isolated patients, and patients'

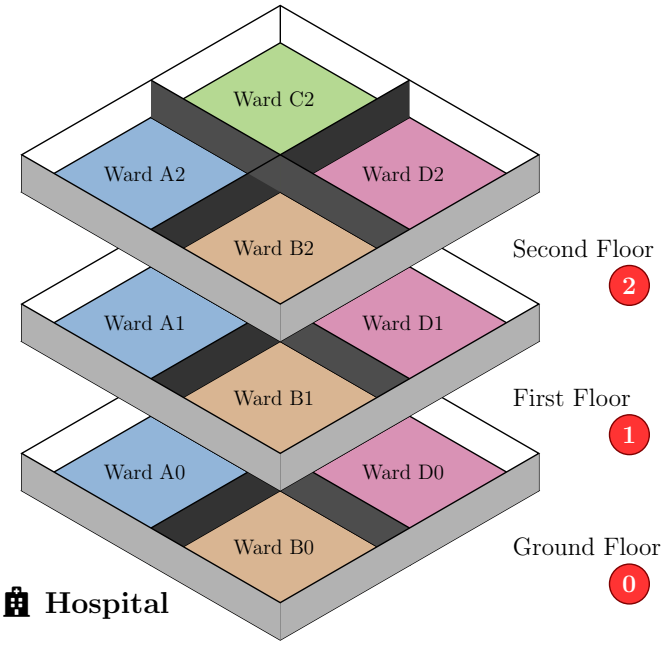

Fig. 2: A hospital structure with multiple wards across the different floors. The wards are mapped to different floors, allowing the system to account for the distance between wards when computing patient allocations.

monitoring needs (related to the equipment required in the rooms). In this paper, we further consider an additional calculated penalty for patient reallocation based on the distance between wards. Let δ_r be the penalty of a room, defined as:

$$\delta_r = \delta_{initial} + \alpha \cdot d(r, initial)$$

where $\delta_{initial}$ is the penalty of the ward where the patient arrived, α is a scaling factor, and $d(r, initial)$ is the distance between the proposed room r for reallocation and the initial ward. This captures an increased penalty for allocating patients to rooms that are further away from their initial ward. By minimising the penalty for allocating patients to rooms that meet their needs, the DT suggests bed-bay allocations that are optimal across different wards.

VI. Artifact Execution

To show the execution of the BedreFlyt *Artifact*, we consider a representative use case of bed bay allocation in a hospital. We model a complete hospital environment and simulate the bed allocation process using the components of BedreFlyt. The objective is to evaluate the artifact's ability to recommend bed bay allocations across different wards while considering the patient needs, bed availability, and resource utilisation. We now describe the experimental setup and the resulting generated bed bay allocation.

We support two ways to run the artifact: (1) the *allocation* path, which allocates patients into the wards using the dashboard; and (2) the *simulation* path, which simulates patients arriving to the hospital and automatically allocates them to the wards. The simulation path is useful for testing the DT's performance under different patient arrival rates and evaluating its ability to handle varying levels of demand.

A. Allocation Path

The allocation path allows the user to request bed bay allocations for incoming patients through the *Dashboard* component of the DT. The user can select patient(s) with their associated diagnosis and the ward into which the patient should be allocated. The artifact generates bed bay allocation recommendations based on the current hospital conditions and patient needs. The user can further decide to change the allocation policy to automatic ward reconfiguration and whether extra rooms can be made available in the ward, which can affect the allocation recommendations generated by the artifact. The artifact adapts to the configuration and generates recommendations that align with the hospital's policies and patient care requirements. The user can review the recommendations and make necessary adjustments to ensure that they align with the hospital's policies and patient care requirements. This path is useful for scenarios where patients arrive at a hospital and need to be allocated to appropriate wards.

B. Simulation Path

To explore what-if scenarios in the DT, an event stream generator [28] can be leveraged to simulate a stream of patients with different characteristics arriving at the hospital. This generator can be configured to produce a stream of patients in a modular way, allowing users to test the artifact's functionality under different scenarios, by specifying, e.g., (1) the average arrival rate of patients (the lambda parameter of the Poisson distribution); (2) the peak generation parameter, where every x time steps the amount of patients generated by the Poisson distribution varies to replicate the arrival of patients in an uncertain scenario; (3) the number of time steps to simulate and the number of iterations to run; and (4) the composition of different streams reflecting, e.g., seasonal flu and regular patient traffic. The event stream generator can be used as a CLI tool to test the DT's functionality and evaluate its performance.

C. Results

The allocation and simulation paths allow us to explore the artifact's ability to generate recommendations for bed bay allocation between different wards while considering patient needs, bed bay availability, and resource utilisation. Through the allocation path, we can request bed bay allocations for incoming patients and review the recommendations generated by the artifact via the *Dashboard*. The reviewed recommendations were then used to allocate the patients to the appropriate wards, possibly overriding the DT's suggestions. Figure 3 shows the *Dashboard*, providing the interface with the DT's suggested allocations for the human-in-the-loop approval. The figure shows a bed bay allocation at room level for four patients and single gender rooms, displaying the tab for the "Intensiv" ward. Through the simulation path, we explored patient arrivals to evaluate the artifact's performance under different scenarios, including high patient arrival rates, low

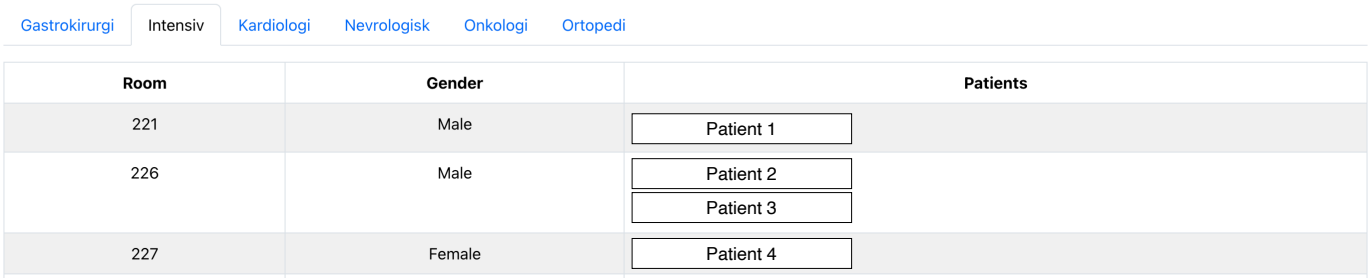

Fig. 3: The BEDREFLYT *Dashboard* component (snippet), showing a suggested bed bay allocation with single gender rooms.

patient arrival rates, and varying levels of demand. By applying different parameters to the event stream generator, we tested the DT's ability to handle varying levels of demand and evaluate its performance under different scenarios.

D. Threats to Validity

While the results of using the artifact for the use case are promising, there are several aspects that should be considered. First, the simulated hospital environment is simplistic and does not capture the complexities of a real hospital setting. This could limit how general the results of using the artifact are with respect to real-world scenarios. Moreover, the performance of BEDREFLYT may be influenced by scalability factors such as the specific patient population or hospital policies that were not explored in the example experiment of this paper.

Moreover, the use case focused on bed bay allocation and did not consider other aspects of hospital operations that could impact patient care quality and resource utilisation like supplies and equipment that are required for the correct trajectory of the treatment of patients. Future research should address these limitations to move the BEDREFLYT *Artifact* closer to real-world hospital settings. Finally, the choice of penalties for allocating patients to different wards and rooms influences the suggested allocations. In our experiment, the penalties were designed to avoid allocating patients to wards that are far away from the ward they arrived at. Thus, the design of the penalties in our example does not capture the complexities of patient needs and hospital operations. Future research should explore different penalty designs to better understand their impact on bed bay allocation outcomes.

VII. REUSABILITY OF BEDREFLYT

The BEDREFLYT[1] Artifact, available for download and experimentation at [29], is intended to support the reproducibility and continued evaluation of the research developed throughout the BedreFlyt project [8]–[10]. Researchers can deploy the artifact, configure different hospital settings by adapting the ontology, input data, and operational parameters, reproduce the workflows presented in this paper, and evaluate alternative allocation scenarios under comparable conditions.

The current release captures the state of the BedreFlyt project at the time of this publication. It consolidates the main capabilities developed across the project, including semantic reflection, lifecycle management [26], simulation, optimisation, and human-in-the-loop runtime decision support [10], [11]. The accompanying documentation explains how to deploy the artifact, configure hospital scenarios, and execute both the allocation and simulation workflows. This allows researchers to reproduce the reported experiments, investigate different hospital configurations, and use the artifact as a common executable baseline for comparison with alternative approaches.

Beyond the current release, we envision the BEDREFLYT Artifact evolving together with the BedreFlyt research roadmap. As new capabilities are developed and experimentally validated through subsequent BedreFlyt publications, they can be progressively incorporated into future releases of the artifact. This may include new runtime reasoning mechanisms, optimisation strategies, simulation capabilities, and other DT functionality. Thus, the artifact serves not only as the software accompanying this paper, but also as a continuously maintained research artifact that reflects the scientific evolution of BEDREFLYT while preserving reproducibility across releases.

VIII. CONCLUSION

In this paper, we presented the BEDREFLYT *Artifact*, an executable software implementation of a Self-Adaptive Digital Twin for hospital resource allocation. The artifact integrates semantic reflection through SMOL, lifecycle management, simulation, optimisation, and a human-in-the-loop dashboard to support runtime decision making in hospital environments.

Through a representative hospital bed bay allocation use case, exercising both the allocation and simulation paths of the artifact, we demonstrated how BEDREFLYT generates meaningful allocation recommendations while supporting admitting nurses in the decision-making process. The artifact further provides a reproducible evaluation setup that researchers can configure to investigate different hospital scenarios and compare alternative approaches under varying operational conditions.

Future work will focus on evaluating BEDREFLYT in real-world settings and further assessing its potential to improve patient outcomes and resource utilisation. We also plan to extend future releases of the artifact with additional operational aspects, such as supplies and equipment required throughout the patient treatment process, and with alternative optimisation

[1]For the maintained sources, see https://github.com/BedreFlyt/bedreflyt.

and runtime reasoning techniques. More broadly, we aim to continue evolving the artifact alongside the BedreFlyt research programme, so that it remains a common executable baseline for future experimentation and comparison.

Acknowledgement

We thank all contributors to the BedreFlyt project; in particular, Paul Kobialka implemented the Z3 Optimiser, Buster S. Rasmussen implemented the Event Stream Generator, and Rudi Schlatte provided feedback on the artifact.

References

[1] K. Willcox and B. Segundo, "The role of computational science in digital twins," *Nat. Comput. Sci.*, vol. 4, no. 3, pp. 147–149, 2024. [Online]. Available: https://doi.org/10.1038/s43588-024-00609-4

[2] National Academies of Sciences, Engineering, and Medicine, "Foundational research gaps and future directions for digital twins," 2023. [Online]. Available: https://doi.org/10.17226/26894

[3] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital twin in manufacturing: A categorical literature review and classification," *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1016–1022, 2018. [Online]. Available: https://doi.org/10.1016/j.ifacol.2018.08.474

[4] A. Billey and T. Wuest, "Energy digital twins in smart manufacturing systems: A case study," *Robotics Comput. Integr. Manuf.*, vol. 88, p. 102729, 2024. [Online]. Available: https://doi.org/10.1016/j.rcim.2024.102729

[5] X. Chang, R. Zhang, J. Mao, and Y. Fu, "Digital twins in transportation infrastructure: An investigation of the key enabling technologies, applications, and challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 7, pp. 6449–6471, 2024. [Online]. Available: https://doi.org/10.1109/TITS.2024.3401716

[6] A. Vallée, "Digital twin for healthcare systems," *Frontiers Digit. Health*, vol. 5, 2023. [Online]. Available: https://doi.org/10.3389/FDGTH.2023.1253050

[7] N. Withanachchi, Y. Uchida, S. Nanayakkara, D. Samaranayake, and A. Okitsu, "Resource allocation in public hospitals: Is it effective?" *Health policy*, vol. 80, pp. 308–313, 03 2007. [Online]. Available: https://doi.org/10.1016/j.healthpol.2006.03.014

[8] R. Sieve, P. Kobialka, L. Slaughter, R. Schlatte, E. B. Johnsen, and S. L. Tapia Tarifa, "BedreFlyt: Improving patient flows through hospital wards with digital twins," in *Proc. 1st Intl. Workshop on Autonomous Systems Quality Assurance and Prediction with Digital Twins*, ser. Electronic Proceedings in Theoretical Computer Science, vol. 418. Open Publishing Association, 2025, pp. 1–15. [Online]. Available: https://doi.org/10.4204/EPTCS.418.1

[9] Å. A. A. Kløvstad, P. Kobialka, R. Sieve, A. Pferscher, L. Slaughter, S. L. Tapia Tarifa, and E. B. Johnsen, "What-if scenarios for the BedreFlyt digital twin," in *Principles of formal quantitative analysis — Essays dedicated to Christel Baier on the occasion of her 60th birthday*, ser. LNCS, vol. 15760. Springer, 2026, pp. 360—-381. [Online]. Available: https://doi.org/10.1007/978-3-031-97439-7_18

[10] R. Sieve, P. Kobialka, A. Pferscher, N. Bencomo, S. L. Tapia Tarifa, B. S. Rasmussen, and E. B. Johnsen, "A self-adaptive digital twin architecture for dynamic resource management," in *Proc. 21st Intl. Conf. on Software Engineering for Adaptive and Self-Managing Systems (SEAMS'26)*. ACM, 2026, pp. 92–104. [Online]. Available: https://doi.org/10.1145/3788550.3794872

[11] E. Kamburjan, A. Pferscher, R. Schlatte, R. Sieve, S. L. Tapia Tarifa, and E. B. Johnsen, "Semantic reflection and digital twins: A comprehensive overview," in *The Combined Power of Research, Education, and Dissemination: Essays Dedicated to Tiziana Margaria on the Occasion of Her 60th Birthday*, ser. LNCS. Springer, 2025, vol. 15240, pp. 129–145. [Online]. Available: https://doi.org/10.1007/978-3-031-73887-6_11

[12] H. K. Rudsari, B. Tseng, H. Zhu, L. Song, C. Gu, A. Roy, E. Irajizad, J. Butner, J. P. Long, and K. Do, "Digital twins in healthcare: a comprehensive review and future directions," *Frontiers Digit. Health*, vol. 7, 2025. [Online]. Available: https://doi.org/10.3389/FDGTH.2025.1633539

[13] Z. Elgammal, M. T. Albrijawi, and R. Alhajj, "Digital twins in healthcare: a review of AI-powered practical applications across health domains," *J. Big Data*, vol. 12, no. 1, p. 234, 2025. [Online]. Available: https://doi.org/10.1186/S40537-025-01280-W

[14] A. Carbonaro, A. Marfoglia, L. Quaranta, S. Mellone, and F. Lanubile, "Editorial: Implementing digital twins in healthcare: pathways to person-centric solutions," *Frontiers Digit. Health*, vol. 7, 2025. [Online]. Available: https://doi.org/10.3389/FDGTH.2025.1741466

[15] Y. Lin and P. Tsai, "A multiobjective resource allocation strategy for personal healthcare digital twins in dynamic edge environments," *IEEE Internet Things J.*, vol. 12, no. 23, pp. 51 629–51 639, 2025. [Online]. Available: https://doi.org/10.1109/JIOT.2025.3614647

[16] G. Anyene, C. Schultz, A. Nepomuceno, and I. Kim, "Digital-twin co-simulation framework to support informed decision in healthcare planning and management," *Simul.*, vol. 101, no. 3, pp. 361–375, 2025. [Online]. Available: https://doi.org/10.1177/00375497241283047

[17] N. Naik *et al.*, "Legal and ethical consideration in artificial intelligence in healthcare: Who takes responsibility?" *Frontiers in Surgery*, vol. 9, 2022. [Online]. Available: https://doi.org/10.3389/fsurg.2022.862322

[18] T. Müller, B. Lindemann, T. Jung, N. Jazdi, and M. Weyrich, "Enhancing an intelligent digital twin with a self-organized reconfiguration management based on adaptive process models," *Procedia CIRP*, vol. 104, pp. 786–791, 2021. [Online]. Available: https://doi.org/10.1016/j.procir.2021.11.132

[19] R. Casadei, A. Placuzzi, M. Viroli, and D. Weyns, "Augmented collective digital twins for self-organising cyber-physical systems," in *Proc. IEEE Intl. Conf. on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*. IEEE, 2021, pp. 160–165. [Online]. Available: https://doi.org/10.1109/ACSOS-C52956.2021.00051

[20] J. R. Lopez, J. de Jesus Camacho, P. Ponce, B. MacCleery, and A. Molina, "A real-time digital twin and neural net cluster-based framework for faults identification in power converters of microgrids, self organized map neural network," *Energies*, vol. 15, no. 19, p. 7306, Oct. 2022.

[21] E. Kamburjan, V. N. Klungre, R. Schlatte, E. B. Johnsen, and M. Giese, "Programming and debugging with semantically lifted states," in *Proc. 18th Intl. Conf. on the Semantic Web (ESWC 2021)*, ser. LNCS, vol. 12731. Springer, 2021, pp. 126–142. [Online]. Available: https://doi.org/10.1007/978-3-030-77385-4_8

[22] E. Kamburjan, V. N. Klungre, R. Schlatte, S. L. Tapia Tarifa, D. Cameron, and E. B. Johnsen, "Digital twin reconfiguration using asset models," in *Proc. ISoLA*, ser. LNCS, vol. 13704. Springer, 2022, pp. 71–88. [Online]. Available: https://doi.org/10.1007/978-3-031-19762-8_6

[23] E. Kamburjan, V. N. Klungre, Y. Qu, R. Schlatte, E. V. Kostylev, M. Giese, and E. B. Johnsen, "Semantically reflected programs," *Trans. Graph Data and Knowledge (TGDK)*, vol. 4, no. 1, pp. 3:1–3:52, 2026. [Online]. Available: https://doi.org/10.4230/TGDK.4.1.3

[24] L. M. de Moura and N. S. Bjørner, "Z3: an efficient SMT solver," in *Proc. Tools and Algorithms for the Construction and Analysis of Systems (TACAS08)*, ser. LNCS, vol. 4963. Springer, 2008, pp. 337–340. [Online]. Available: https://doi.org/10.1007/978-3-540-78800-3_24

[25] E. B. Johnsen, R. Hähnle, J. Schäfer, R. Schlatte, and M. Steffen, "ABS: A core language for abstract behavioral specification," in *Proc. FMCO*, ser. LNCS, vol. 6957. Springer, 2010, pp. 142–164. [Online]. Available: https://doi.org/10.1007/978-3-642-25271-6_8

[26] E. Kamburjan, N. Bencomo, S. L. Tapia Tarifa, and E. B. Johnsen, "Declarative lifecycle management in digital twins," in *Proc. 1st Intl. Conf. on Engineering Digital Twins (EDTconf 2024)*, ser. MODELS Companion'24. ACM, 2024, pp. 353–363. [Online]. Available: https://doi.org/10.1145/3652620.3688248

[27] E. Kamburjan, N. Bencomo, E. B. Johnsen, and S. L. Tapia Tarifa, "Declarative lifecycle management in digital twins," *Software and Systems Modeling*, Feb. 2026, to appear. [Online]. Available: https://doi.org/10.1007/s10270-026-01358-w

[28] R. Sieve, P. Kobialka, A. Pferscher, N. Bencomo, S. L. Tapia Tarifa, B. S. Rasmussen, and E. B. Johnsen, "Additional Material for the paper A Self-Adaptive Digital Twin Architecture for Dynamic Resource Management (Version 3)," Jan. 2026. [Online]. Available: https://doi.org/10.5281/zenodo.18340349

[29] R. Sieve, N. Bencomo, S. L. Tapia Tarifa, and E. B. Johnsen, "Bedreflyt artifact: A self-adaptive digital twin software system for hospital resource allocation," 2026. [Online]. Available: https://zenodo.org/doi/10.5281/zenodo.21456563