UDC 004.94:004.052:004.415 https://doi.org/10.32515/2664-262X.2025.11(42).2.38-44

Olena Shyshatska[1], PhD phys. and math. sci., **Liubomyr Matiichuk**[2], DSc., **Andrii Shyshatskyy**[1]
[1]*Taras Shevchenko National University of Kyiv, Kyiv, Ukraine*
[2]*Ternopil Ivan Puluj National Technical University, Ternopil, Ukraine*
e-mail: shyshatska@knu.ua, mlpstat@gmail.com, ashyshatskyy@knu.ua

Specification and Verification of a Formal Model for a Real-Time System Using TLA+

The paper describes theoretical and applied aspects of modeling real-time information systems using the example of a continuous glucose monitoring (CGM) system. The primary focus is on the application of formal methods, in particular the TLA+ specification language and modal logics, which ensure mathematical precision in describing system behavior. A formal model of the CGM system is constructed, safety invariants are defined, and automated model verification is performed using the TLC mechanism. The obtained results can be used to improve the reliability and justify the correctness of software-hardware solutions in the field of medical IT systems, particularly Closed-Loop Systems (CLS), which currently lack appropriate regulatory certification. The proposed approach contributes to the formalization of the validation process for such systems in accordance with standards for safety-critical software.

real-time systems, CGM, model, formal design methods, specification, verification, TLA+

Problem Statement. Continuous glucose monitoring (CGM) systems belong to the class of real-time software-hardware systems operating in sensitive medical environments. Such systems are characterized by stringent requirements for accuracy, performance, safety, and the handling of large volumes of data, which aligns fully with the typical characteristics of critical IT systems [1]. Ensuring the reliable operation of such systems is a key prerequisite for their use in the medical domain.

Despite the widespread adoption of CGM devices in clinical practice, a number of unresolved technical challenges remain. These include ensuring measurement accuracy, proper real-time event handling, adaptation to individual usage scenarios, and the lack of formal certification for certain components within Closed-Loop Systems (CLS). Many CLS implementations lack official certification, which increases the risks for end users.

Formal methods, particularly the use of specification languages such as TLA+ (Temporal Logic of Actions) [2, 3], make it possible to describe the behavior of IT systems, define invariants, formally verify safety and performance properties, and detect design errors at early development stages, thereby reducing the cost of their correction. However, despite the existence of general theoretical frameworks, there remains a need for applied research that demonstrates the practical application of formal methods across the full development cycle of medical software-hardware IT systems. This necessity determines the relevance of the present study.

The aim of the study is to develop a formal model of a CGM system using the TLA+ specification language and to verify its properties in terms of invariance, safety, and correctness of state transitions.

To achieve this goal, the following tasks must be completed:

1. Analyze the technical requirements for CGM as a real-time system;
2. Justify the choice of TLA+ as a formal means of describing system behavior;
3. Develop a formal specification of the monitoring model, including key invariants;
4. Perform automated verification using the TLC mechanism;
5. Assess the practical effectiveness of the approach based on experimental scenarios.

Review of Recent Research and Publications. Continuous glucose monitoring (CGM) systems belong to the class of safety-critical software-hardware complexes operating

in real-time environments. They are used in medical settings and are characterized by stringent requirements for accuracy, performance, and safety. The design of such systems becomes particularly complex in the context of Closed-Loop Systems (CLS), which automate insulin delivery. While commercial solutions are steadily improving, a significant portion of the market is occupied by DIY systems, which are not subject to certification and therefore pose additional risks.

Despite advances in the hardware components of CGM systems, the issues of formal modeling and verification of their software behavior remain underexplored. In [1], the authors highlight the limitations of traditional testing in the context of safety-critical IT systems, as not all errors can be detected empirically. They propose a multi-level verification strategy based on constructing specifications using formal languages, notably Z.

An early example of applying formal methods to glucose monitoring systems is presented in [2], which proposes a top-down approach to the design of insulin systems using atomic coordinated actions. The development was carried out within the CAA-DRIP (Computer Aided Architecture – Distributed Real-time Implementation Platform) environment, which supports formal system modeling.

In this context, interest in formal development languages is growing, particularly the TLA+ (Temporal Logic of Actions) language developed by L. Lamport [3]. TLA+ allows for the description of parallel, sequential, and reactive behaviors of IT systems, as well as verification of their properties using the TLC Model Checker. Practical aspects of planning and verifying invariants in TLA+ are detailed in [4]. The theoretical foundation of TLA+ lies in temporal and modal logics [5], which enable the formal specification of temporal properties such as obligation, possibility, and event succession. The foundational aspects of these logics are elaborated in [5–6], and their application to the control of dynamic systems is illustrated in [4]. All these approaches contribute to the development of verifiable behavioral models for complex real-time IT systems.

Examples of the effective use of TLA+ are presented in [9], where the language is employed at AWS to verify the consistency and correctness of distributed storage systems, and in [10], which demonstrates the application of formal methods in the design of embedded systems with guaranteed safety properties. A comprehensive overview of the implementation of formal methods in critical IT domains is provided in [11], which examines experiences in the transportation, aviation, and financial sectors.

However, the application of TLA+ to the design of medical IT systems, particularly CGM/CLS, remains limited. This creates an open niche for further research.

Thus, the current literature confirms the existence of powerful formal method toolkits in computer science, including TLA+, yet their application to glucose monitoring systems has not become widespread. This underlines the need for applied developments focused on the formal specification and verification of CGM system behavior in real-time operation.

Problem Definition. This study sets out the task of developing an adaptive formal model of a glucose monitoring system that can be adjusted to meet the specifications of specific CGM solutions. The developed second-generation CGM model is implemented in the TLA+ language and is designed to be scalable for integration into Closed-Loop Systems (CLS).

The proposed approach enables: formal verification of algorithms in DIY-type CLS systems, thereby increasing their reliability; the development of unified criteria for certifying such medical IT solutions; and the optimization of component integration between CGM and CLS, ensuring more effective insulin therapy management for users.

Main Content. Modeling a CGM system requires tools for the formal description of its behavior across different temporal frames, as well as for the specification of functional requirements. Temporal modal logic is an optimal choice for modeling system behavior, as it specializes in working with time-based models [7–10]. Alethic logic is used for the formalization of system requirements, as it provides sufficient expressive power for system

specification and verification. It is worth noting that modern development environments support both alethic and temporal logics, facilitating their use in the modeling process.

The modeling language must meet a number of requirements specific to CGM as a real-time software-hardware system. In particular, the language should:

1. Possess a formal semantics that ensures unambiguous specifications and enables mathematical verification;
2. Support both temporal and alethic logics;
3. Be suitable for modeling behavior within temporal spaces;
4. Provide tools for the formal verification of invariants and properties;
5. Ensure scalability, i.e., the ability of the model to adapt to more complex architectures (e.g., CLS systems);
6. Remain sufficiently simple for development and testing.

Several commonly used specification and modeling languages were considered—TLA+, UML, SysML, and Promela. Each has its advantages; however, TLA+ [2–3] proved to be the most suitable based on the defined criteria.

TLA+ integrates the description of variables, actions, temporal constraints, and invariants within a single formal paradigm. Specifications in TLA+ allow for the definition of system behavior in the form of temporal formulas that express necessary or permitted transitions between system states, while the TLC tool enables automated verification of these properties.

Developing a model in TLA+ involves the formalization of system behavior using temporal logic formulas. The TLA+ syntax supports a precise and structured description of system functionality—incorporating variables, actions, and temporal properties [3].

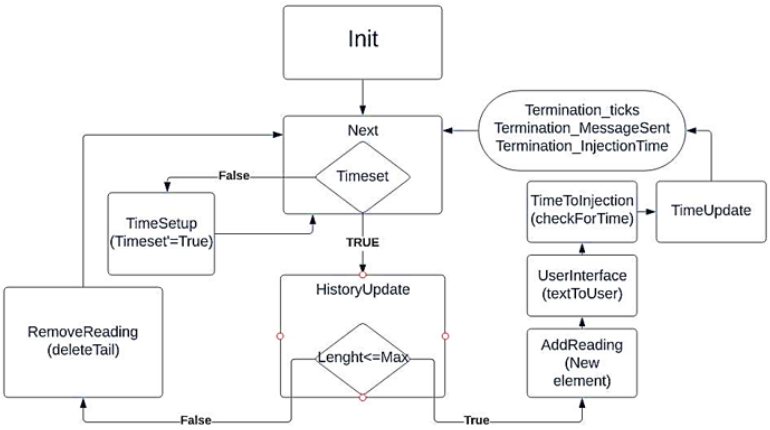

Figure 1 – Diagram of the Glucose Monitoring System Model

Source: developed by the authors

The model includes the following components:

1. Variables describing the system state: blood glucose level; indicators of glucose level exceeding permissible limits; amount of carbohydrates entering the bloodstream; alerts for insulin administration; informational messages for the user; timestamp of alerts; glucose level thresholds; critical states triggering emergency alerts; current time.
2. Actions that define CGM functionality, including: glucose level measurement; checking insulin administration time; generating user alerts; adding new glucose readings; removing outdated data; updating the current time.
3. Key constraints and invariants: limiting the number of steps in the system; controlling glucose levels within defined thresholds; guaranteeing the existence and dispatch of user notifications.

One of the main advantages of TLA+ is its capability to analyze system behavior at a high level of abstraction. This enables the validation of system performance across different scenarios and ensures compliance with specifications.

Based on the defined requirements, a model diagram was developed, serving as the foundation for its implementation in TLA+.

In the diagram, rectangles represent functions, diamonds denote conditions, and rounded rectangles correspond to system requirements. The operation of the system begins with the initialization of initial variables in the Init function, which are then passed to the Next function. In the first step, the system time is set and subsequently fixed (as CGM systems typically operate over a two-week period, resetting the time is not required). The system then updates based on new glucose level values. If the measurement history exceeds the allowable length, older entries are deleted; otherwise, a new entry is added, a user notification is sent, followed by a check of the time for insulin administration and an update of the current time. In the final stage, the system's requirements are verified, and if all conditions are satisfied, the system proceeds to the next step.

The model was developed using TLA Toolbox – the official IDE for TLA+, created by Leslie Lamport and the Microsoft Research team [10]. In the CGM model, user-defined parameters – such as glucose threshold values or insulin notification times – are treated as constants. The rationale behind this approach is threefold: (1) the parameters remain unchanged during system operation, ensuring modeling accuracy and verification reliability; (2) ease of modification – using constants enables testing the system under various conditions without altering the core model; (3) formal specification – specification languages are intended for formally describing system behavior, not for runtime user input during model checking.

To optimize the model and reduce verification load, several simplifications were introduced. These do not affect verification correctness but significantly lower computational cost: (1) the glucose level is represented as an integer value; (2) a maximum glucose level is defined to constrain the value range; (3) the time logging function is simplified, since its computation is resource-intensive, and in real systems, these parameters are user-provided, eliminating the need for step-by-step verification.

This approach allows for efficient modeling of CGM behavior and ensures its formal verification using TLA+.

The developed model includes the following key components [11]:

1. Initialization of variables and constants – the necessary parameters are defined, including: currentLength – the number of entries in the glucose level history; currentTicks – the number of executed steps (in most real-world models, the terminal state is determined by a time limit of two weeks, corresponding to the sensor's lifespan); message and reminder – user notification and reminder to administer insulin; IsBad, IsCritical – Boolean variables indicating whether the glucose level is outside the normal range or has reached critical values.

2. Initialization of the initial state and transitions: the Init function sets the initial values of the variables; the Next function defines the possible system transitions: at the first step, the initial time is set, and subsequent transitions are implemented in the HistoryUpdate function.

3. Functional modules of the model: TimeSetup sets a fixed time as a constant and changes TimeSet to true, preventing further calls to this function; HistoryUpdate adds a new glucose reading, generates messages, and checks whether it is time to remind the user to administer insulin. If the allowed history length is exceeded, the oldest entry is deleted; RemoveReading deletes the oldest reading if the history exceeds the predefined length; AddReading adds a new glucose level reading and increases currentTicks to simulate system operation over time; UserInterface generates user messages based on glucose levels and sets IsBad and IsCritical values to alert the user or their relatives in case of critical glucose changes. If sensor data is unavailable, IsBad is set to true, signaling a read error; SendMessage generates the message text according to the provided parameters; TimeToInjection checks whether the current time matches the predefined insulin administration time. If the condition is met, SendReminder is set to true, triggering a reminder

for the user. Afterwards, TimeUpdate updates the time, since glucose level measurements by the sensor are typically performed every minute, making second-level precision unnecessary.

4. Temporal formulas for execution correctness control: Termination_Ticks – defines process termination if the number of performed actions exceeds the predefined limit; Termination_MessageSent – ensures that once a message has been sent, the user will not receive empty notifications; Termination_InjectionTime – checks whether the user has received a reminder to administer insulin; if the message content is absent, the condition is automatically considered satisfied.

To verify the correctness of the constructed model, the TLC Model Checker, part of the TLA+ Toolbox, was used. The model was verified with various sets of parameters, which included changes to MaxGlucose, MaxHistoryLength, and MaxTicks – the key variables with the greatest impact on the model's computational complexity. MaxGlucose defines the maximum glucose value that can be represented in the system. MaxHistoryLength determines the length of the stored measurement history. MaxTicks defines the number of time units during which the system must function.

The model was tested under various input parameter values, some of which are presented in Table 1. Note: Due to the specifics of specification languages, glucose level is represented as an integer value, as the current stage of the study focuses on analyzing the general system behavior across different glucose level zones.

Table 1 – model verification results

CGM system model					
system configuration				computational time	number of states
	MaxGlucose	MaxHistory Lenght	MaxTicks		
1	10	1	2000	00:00:15	$1.15 * 10^6$
2	20	1	2000	00:01:07	$1.9 * 10^7$
3	20	3	2000	Out Of Memory	
4	10	3	200	00:00:28	$5.7 * 10^7$
5	20	3	200	00:19:234	$7.5 * 10^7$
6	CONST	5	2000	00:00:05	$8.3 * 10^4$
7	10	5	20	00:15:17	$4.6 * 10^7$

Source: developed by the authors

The results of verifying the developed CGM model in TLA+ confirmed its correctness and compliance with the specified formal requirements. In particular, the specification contains no deadlock states, and all verified temporal formulas hold within the defined state space. Among the properties tested were the presence of user notifications, the system's response to critical deviations in glucose levels, and timely transitions between the system's key operational modes.

The developed model represents a foundational stage in the formal verification of CGM systems and demonstrates the potential of applying formal methods to the modeling of mission-critical IT systems. The primary practical purpose of the model is to verify the correctness of the notification logic in a simplified CGM system, which automates glucose monitoring and reduces the psychological burden on patients. Even in this simplified form, the model contributes to improving the quality of life for individuals with diabetes by reducing the need for manual measurements and ensuring timely responses to changes in glucose levels.

Based on the results obtained and the current structure of the formalized model, several directions for further extension have been identified:

1. Implementation of algorithms for analyzing glucose trend dynamics;

2. Integration of machine learning methods for glucose level prediction based on historical data;

3. Consideration of additional influencing factors (physical activity, stress, nutrition, etc.);

4. Development of mechanisms for detecting hypo-/hyperglycemia with appropriate warning logic;

5. Integration with insulin pumps as part of a closed-loop system (CLS) model.

Future extensions of the formal analysis may include the use of other logics (e.g., multivalued or modal) as well as verification of more complex properties such as liveness and fault recovery.

Overall, the CGM model in TLA+ is a promising tool for further research in diabetes management and opens up broad opportunities for improving and optimizing glucose monitoring systems.

Conclusions. As part of this study, an adaptive model of a Continuous Glucose Monitoring (CGM) system was developed, which can be flexibly configured to meet the specifications of particular systems under verification [14]. Formal methods, particularly modal logics (temporal and alethic), were employed as a universal toolkit to provide a mathematically rigorous description of properties such as necessity, possibility, and temporal dependence. The model was implemented in the TLA Toolbox—an official integrated development environment for the TLA+ specification language.

The developed specification models the core functional components of a second-generation CGM system, including temporal constraints, safety invariants, user notification logic, and measurement history retention. The use of TLA+ enabled a formal definition of system behavior in a temporal context, ensured the verification of invariants, and facilitated automated functional verification using the TLC Model Checker.

Verification results confirmed the correctness of the model: it contains no deadlock states and satisfies all defined temporal properties, thus ensuring its stability and reliability under various operating scenarios. Notably, the model implements automated logic for notifying users of critical glucose level deviations, which reduces the cognitive burden on users and improves the quality of life for individuals with diabetes.

The proposed modular structure of the model is extendable and scalable to the level of closed-loop systems (CLS), which include automated insulin delivery control. Promising directions for further development include:

List of References

1. Rafeh, R., Rabiee, A. Towards the design of safety-critical software. *Journal of Applied Research and Technology*. 2013. Vol. 11. P. 683–694.
2. Capozucca, A., Guelfi, N. The fault-tolerant insulin pump. *Rigorous Development of Complex Fault-Tolerant Systems*. Berlin : Springer-Verlag, 2006. P. 59–75.
3. Lamport, L. *Specifying systems: The TLA+ language and tools for hardware and software engineers*. Boston : Addison-Wesley Professional, 2002. 384 p.
4. Wayne, H. *Practical TLA+: Planning driven development*. New York : Apress, 2018. 221 p.
5. Fainekos, G., Kress-Gazit, H., Pappas, G. Temporal logic motion planning for mobile robots. *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. Barcelona, 2005. P. 2020–2025.
6. Humberstone, L. *Philosophical applications of modal logic*. London : College Publications, 2016. 588 p.
7. Kripke, S. Semantical considerations for modal logics. *Acta Philosophica Fennica*. 1962. Helsinki. 389 p.
8. Guéron, J. *Time and modality*. (Studies in Natural Language and Linguistic Theory ; Vol. 75). Dordrecht : Springer, 2008. 316 p.
9. Newcombe, C., Rath, T., Zhang, F. et al. How Amazon Web Services uses formal methods. *Communications of the ACM*. 2015. Vol. 58, No. 4. P. 66–73.
10. Henzinger, T. A., Sifakis, J. The discipline of embedded systems design. *IEEE Computer*. 2006. Vol. 39, No. 10. P. 36–44.
11. Woodcock, J., Larsen, P. G., Bicarregui, J., Fitzgerald, J. Formal methods: Practice and experience. *ACM Computing Surveys (CSUR)*. 2009. Vol. 41, No. 4. Article No. 19. P. 1–3.
12. Blood glucose monitoring model: specification and verification. *Github: веб-сайт*. URL: https://github.com/AndrewShysh/TLAPLUS_CGM_System/tree/main (дата звернення: 15.02.2025).

References

1. Rafeh, R., & Rabiee, A. (2013). Towards the design of safety-critical software. *Journal of Applied Research and Technology*, 11, 683–694.
2. Capozucca, A., & Guelfi, N. (2006). The fault-tolerant insulin pump. In *Rigorous Development of Complex Fault-Tolerant Systems* (pp. 59–75). Berlin: Springer-Verlag.
3. Lamport, L. (2002). *Specifying systems: The TLA+ language and tools for hardware and software engineers*. Boston: Addison-Wesley Professional.
4. Wayne, H. (2018). *Practical TLA+: Planning driven development*. New York: Apress.
5. Fainekos, G., Kress-Gazit, H., & Pappas, G. (2005). Temporal logic motion planning for mobile robots. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation* (pp. 2020–2025). Barcelona: IEEE.
6. Humberstone, L. (2016). *Philosophical applications of modal logic*. London: College Publications.
7. Kripke, S. (1962). Semantical considerations for modal logics. *Acta Philosophica Fennica*, Helsinki, 389.
8. Guéron, J. (2008). *Time and modality* (Studies in Natural Language and Linguistic Theory; Vol. 75). Dordrecht: Springer.
9. Newcombe, C., Rath, T., Zhang, F., et al. (2015). How Amazon Web Services uses formal methods. *Communications of the ACM*, 58(4), 66–73.
10. Henzinger, T.A., & Sifakis, J. (2006). The discipline of embedded systems design. *IEEE Computer*, 39(10), 36–44.
11. Woodcock, J., Larsen, P.G., Bicarregui, J., & Fitzgerald, J. (2009). Formal methods: Practice and experience. *ACM Computing Surveys (CSUR)*, 41(4), Article 19, 1–3.
12. AndrewShysh. (2025, February 15). *Blood glucose monitoring model: Specification and verification* [GitHub repository]. GitHub. https://github.com/AndrewShysh/TLAPLUS_CGM_System/tree/main.

О. В. Шишацька[1], канд. фіз.-мат. наук, **Л. П. Матійчук**[2], доц. д-р екон. наук, **А. В. Шишацький**[1]
[1]*Київський національний університет імені Тараса Шевченка, м. Київ, Україна*
[2]*Тернопільський національний технічний університет імені Івана Пулюя, м. Тернопіль, Україна*

Специфікація та верифікація формальної моделі системи реального часу із використанням TLA+

У роботі представлено розробку формальної моделі програмно-апаратної системи реального часу на прикладі системи безперервного моніторингу рівня глюкози (Continuous Glucose Monitoring, CGM). CGM-системи є критично важливими IT-рішеннями, що використовуються в медичному середовищі для постійного вимірювання рівня глюкози в крові та своєчасного сповіщення користувачів. З огляду на дедалі ширше впровадження CGM у замкнені контури керування (Closed-Loop Systems, CLS), які автоматизують введення інсуліну, зростає потреба у забезпеченні коректності таких систем, особливо враховуючи популярність неперевірених рішень, створених користувачами самостійно (DIY). Попри досягнення в галузі фізичних та сенсорних технологій, програмне забезпечення CGM-систем часто не має формальних специфікацій та перевірок. Це створює ризики у передбачуваності, безпеці та надійності, що є особливо критичними у медичних застосуваннях. Метою дослідження є застосування формальних методів, зокрема TLA+, для специфікації, моделювання та верифікації поведінки CGM-системи, що функціонує в режимі реального часу.

Запропонована модель описує основні компоненти CGM-системи: цикли вимірювання в часі, збереження історичних даних, виявлення перевищень порогових значень і логіку сповіщення. Також модель фіксує інваріанти системи, зокрема правильну послідовність переходів між станами та своєчасне генерування сповіщень. Для реалізації обрано TLA+, що ґрунтується на темпоральній та модальній логіці, підтримує моделювання конкурентних та реактивних систем і має потужні інструменти, зокрема перевірку моделей TLC. Модель реалізовано та протестовано за допомогою TLA+ Toolbox і TLC. Розглянуто кілька конфігурацій (порогові значення глюкози, довжина історії, тривалість роботи системи) для оцінки її стійкості. Результати верифікації засвідчили відсутність тупикових станів і виконання всіх заданих властивостей безпеки. Це підтверджує доцільність використання формального моделювання в TLA+ для коректності CGM-систем, особливо в умовах, де критичними є безпечність, правильність та відслідковуваність. Модульна структура моделі забезпечує можливість її подальшого розширення — включення алгоритмів машинного навчання для прогнозування рівня глюкози, моделювання впливу фізичної активності або харчування, а також поступовий перехід до повністю автономних CLS-середовищ. Крім того, ця робота сприяє поширенню практик формальної специфікації в комп'ютерних науках, зокрема в галузі систем реального часу та критично важливих застосунків.

системи реального часу, CGM, модель, формальні методи проєктування, специфікація, TLA+, верифікація

Одержано (Received) 17.03.2025 *Прорецензовано (Reviewed) 27.04.2025*
Прийнято до друку (Approved) 06.05.2025