electronics

MDPI

Article

Property Analysis of Gateway Refinement of Object-Oriented Petri Net with Inhibitor-Arcs-Based Representation for Embedded Systems

Chuanliang Xia *, Mengying Qin, Yan Sun and Maibo Guo

School of Computer Science and Technology, Shandong Jianzhu University, Jinan 250101, China
* Correspondence: chuanliang_xia@sdjzu.edu.cn

Abstract: This paper focuses on embedded system modeling, proposing a solution to obtain a refined net via the refinement operation of an extended Petri net. Object-oriented technology and Petri net with inhibitor-arcs-based representation for embedded systems (PIRES+) are combined to obtain an object-oriented PIRES+ (OOPIRES+). A gateway refinement method of OOPIRES+ is proposed, and the preservation of the liveness, boundedness, reachability, functionality, and timing of the refined net system is investigated. The modeling analysis of a smart home system is taken as an example to verify the effectiveness of the refinement method. The results can provide an effective way for the investigation of the refined properties of a Petri net system and a favorable means for large-scale complex embedded system modeling, which has broad application prospects.

Keywords: extended Petri nets; object-oriented technology; refinement; property preservation; embedded system modeling

Citation: Xia, C.; Qin, M.; Sun, Y.; Guo, M. Property Analysis of Gateway Refinement of Object-Oriented Petri Net with Inhibitor-Arcs-Based Representation for Embedded Systems. *Electronics* **2023**, *12*, 3977. https://doi.org/10.3390/electronics12183977

Academic Editor: Yannis Papaefstathiou

Received: 30 July 2023
Revised: 10 September 2023
Accepted: 18 September 2023
Published: 21 September 2023

1. Introduction

Embedded systems are the driving force behind the rapid development of the information technology industry and have widely penetrated and been integrated into every corner of the national economic and social development. As an industry leading the future economic boom, the research of the Internet of Things is also inseparable from embedded systems. Roberto et al. [1] introduced an event network SCIFI-II system, which can separate application components from the event brokers that connect components and realize the deployment of components in any environment. The design and manufacturing of embedded systems has become a hot research topic internationally. In order to improve the design and manufacturing efficiency of embedded systems, formal modeling and verification of the system are very important.

For embedded system modeling, there are usually two methods: the non-formal modeling method and the formal modeling method. Non-formal modeling methods can express certain functional characteristics of a system, but they cannot be validated using rigorous mathematical methods. Formal methods use symbols and mathematical language to describe the properties of a system, which can systematically describe and validate the system [2]. Common formal modeling methods mainly include the extended finite-state machine method [3], data flow diagram method [4], communication process method [5], Petri net method [6], entity relationship diagram method, object-oriented UML method [7], etc. These modeling methods describe the characteristics of embedded systems from different perspectives but do not form a unified standard. Due to their intuitive graphical expression ability and rigorous mathematical expression, Petri nets have a wide range of applications in the behavioral analysis of concurrent systems [8]. However, classical Petri nets lack a hierarchical structure, cannot fully describe data flows, and lack the concept of time [9], making it impossible to fully describe embedded systems.

In order to improve the modeling ability of Petri nets on embedded systems, domestic and foreign scholars have improved the classical Petri nets and proposed a variety of extension forms, such as colored Petri nets (CPNs) [10], time Petri nets [11], fuzzy Petri nets [12], logical delay Petri nets, PRES+ [13], etc. These extended forms of Petri nets enhance the ability to model and analyze embedded systems based on different application requirements, where PRES+ can capture real-time information and describe the hierarchical structure of the system.

In the field of embedded system modeling and analysis, research on the theory and application of PRES+ has achieved some results. Cortés et al. [13] proposed the concept of PRES+ and analyzed two embedded system PRES+ models. Karlsson et al. used PRES+ to accurately model and validate mobile phone systems and represented the embedded systems that need to be validated as PRES+ models [14]. They provided a model validation method for analyzing the system's coverage and decidability [15]. By using the PRES+ and model transformation method, Bandyopadhyay et al. [16,17] verified the Very Large Scale Integrated (VLSI) system model. In order to improve the efficiency of PRES+ model verification, Xia et al. [18] modeled and analyzed the subsystem sharing system.

Although PRES+ has significantly improved the accuracy of embedded system modeling compared to classical Petri nets, there are still issues with the inability to describe event priorities and fully express complex program control and data flow. Adding inhibitor arcs to Petri nets can be used to control event sequences, adjust priority relationships between sequences, improve model analysis capabilities, and, to some extent, affect the expression of control flow and data flow. Therefore, adding inhibitor arcs to PRES+ can generate a PRES+ with inhibitor arcs (PIRES+, Petri net with inhibitor-arcs-based representation for embedded systems). Xia et al. [19] analyzed the refinement operation of the PIRES+ and applied it to the modeling and verification of the mobile communication systems.

As a system model, the PIRES+ can consider the system and behavior uniformly, but it is not easy to reflect the division of functional modules. The outstanding advantage of object-oriented technology lies in the use of object-oriented technology to divide the system into functional modules, effectively dividing the entire complex system into multiple simple subsystems and conducting corresponding modeling research. By combining object-oriented technology with Petri nets, an object-oriented Petri net (OOPN) can be obtained, which divides the entire system into multiple subsystem modules based on functional implementation, which is beneficial for modeling large-scale complex systems. There have been some research achievements in the analysis and modeling of the object-oriented Petri nets. Zhao et al. [20] proposed an extended-object-oriented petri net (EOOPN), supporting the description and evolution of components in the context of intelligent and cloud computing. Hu et al. [21] worked on a compression algorithm to convert an object-oriented generalized stochastic Petri net (OGSPN) into a generalized stochastic Petri net (GSPN).

However, as the complexity and scale of embedded systems increase, the number of system states will increase exponentially, highlighting the problem of "state space explosion". Since the "state space explosion" problem of OOPIRES+ is NP-hard, it cannot be completely solved, but it can be alleviated through property preserving transformations. The commonly used Petri net transformation methods include refinement, synthesis, and reduction. Among them, refinement is a "top-down" modeling method. In the modeling of complex system modules, an abstract model is first established; then, it is refined layer by layer; and, finally, a refined model is obtained. When the refinement condition is satisfied, some important properties of the original Petri net system are preserved when the Petri net system is extended without the analysis of the reachable space. Therefore, refinement can alleviate the problem of "state space explosion".

In recent years, extensive research has been conducted on the refinement and application of Petri nets. Padberg et al. [22] provided an overview of refinement methods for place/transition Petri net systems. Wang et al. [23] proposed a Petri net refinement method without reachability analysis. Based on the basic net system, Bernardinello et al. [24]

proposed a refinement method for distributed systems. For the refinement operation of the place/transition net, Huang et al. [25] provided the conditions to keep the original net system's liveness, boundedness, fairness, and other properties. Lakos et al. [26] provided three types of refinement operations for the colored Petri net (CPN). Based on the workflow net, Van Hee et al. [27] presented several refinement rules for preserving system reliability. For extended-time Petri nets, Mejia et al. [28] proposed a refinement operation mode.

Cortes et al. [13] extended PRES+ by introducing the definition of hierarchical structure and proposed some equivalent concepts of PRES+ and refinement methods to preserve complete equivalence. Xia et al. [19] proposed a PIRES+ refinement method and applied it to modeling network communication systems.

This paper proposes an object-oriented PIRES+ (OOPIRES+) and a gateway refinement operation for the formal modeling and analysis of the large-scale complex embedded systems. So far, there has been little research on the gateway refinement operations. Due to the crucial role of gateway refinement in the transmission and control of system data and information flows, it is necessary to conduct in-depth research on it. For the refinement operation of the OOPIRES+'s gateway, we studied the preservation of liveness, boundedness, reachability, functionality, and timing. This can greatly alleviate the "state space explosion" problem encountered in the modeling process of complex embedded systems. We applied this refinement operation to the modeling of smart home control systems, and we found that it has great guiding significance for the research of embedded system modeling.

The structure of this paper is as follows: Section 2 proposes the related concepts of OOPIRES+. Section 3 presents the gateway refinement operation of OOPIRES+ and investigates the preservation of liveness, boundedness, reachability, functionality, and timing. In Section 4, a smart home remote control system is modeled and analyzed by using the OOPIRES+ gateway refinement operation. Section 5 summarizes the contributions of the refinement method.

2. Relevant Concepts of OOPIRES+

This section provides the relevant concepts of OOPIRES+.

Definition 1 ([19]). *$N = (P, T; F_I, F_O, I, M)$ is said to be a PIRES+ model, where $P = \{p_1, p_2, \ldots, p_m\}$ is the place set, $T = \{t_1, t_2, \ldots, t_n\}$ is the transition set, $F_I \subseteq P \times T$ is the input arc set, $F_O \subseteq T \times P$ is the output arc set, $I \subseteq P \times T$ is the inhibitor arc set, and M is the marking.*

Definition 2 ([19]). *For $\forall t \in T$, there exists a transition function, $f : \tau(p_1) \times \tau(p_2) \times \ldots \times \tau(p_n) \to \tau(q)$, where $p_1, p_2, \ldots, p_n \in \cdot t$, $q \in t\cdot$. For $\forall t \in T$, there exists a minimum transition delay, d^-, and a maximum transition delay, d^+, both of which are non-negative real numbers.*

Figure 1 shows an example of the PIRES+ model.

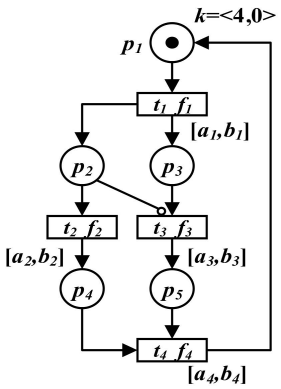

Figure 1. An example of the PIRES+ model.

Here, $P = \{p_1, p_2, p_3, p_4, p_5\}$, and $T = \{t_1, t_2, t_3, t_4\}$; $F_I = \{(p_1, t_1), (p_2, t_2), (p_3, t_3), (p_4, t_4), (p_5, t_4)\}$; $F_O = \{(t_1, p_2), (t_1, p_3), (t_2, p_4), (t_3, p_5), (t_4, p_1)\}$; $I = \{(p_2, t_3)\}$; and M_0 is the initial marking, where $M_0(p_1) = \{\langle 4, 0\rangle\}$ and $M_0(p_2) = M_0(p_3) = M_0(p_4) = M_0(p_5) = \varnothing$. The transition functions of transitions t_1, t_2, t_3, and t_4 are f_1, f_2, f_3, and f_4, respectively. The transition delays of transitions t_1, t_2, t_3, and t_4 are $[a_1, b_1]$, $[a_2, b_2]$, $[a_3, b_3]$, and $[a_4, b_4]$, respectively.

Definition 3. *An OOPIRES+ object subnet is a six-tuple $OPSN = (P, T; F, I, Q, W)$, where $P = \{p_1, p_2, \ldots, p_m\}$ is the place set of OPSN, represented by $OPSN(P)$; $T = \{t_1, t_2, \ldots, t_n\}$ is the transition set of OPSN, represented by $OPSN(T)$; F represents the flow relationship of OPSN, including F_I and F_O, where $F_I \subseteq (P \times T) \cup (Q \times T)$ is the set of input arcs and $F_O \subseteq (T \times P) \cup (T \times Q)$ is the set of output arcs; $I \subseteq P \times T$ is the inhibitor arc set of OPSN; and $Q = \{q_1, q_2, \ldots, q_r\}$ is the message place set of OPSN, represented by $OPSN(Q)$, including input place Q_I and output place Q_O.*

For each transition (t), there are a transition function (f) and a transition delay (d^- and d^+) associated with it. Object subnets are connected by the gateway, g. Gateways are classified into input gateways and output gateways according to their positions relative to subnets. As the gateway can be used to preprocess data before flowing into the next object subnet, here, the gateway (g) is also assigned the attributes of the gateway function, f_g, and the gateway delay, $\left[d_g^-, d_g^+\right]$.

Figure 2 shows an example of the OOPIRES+ object subnet.

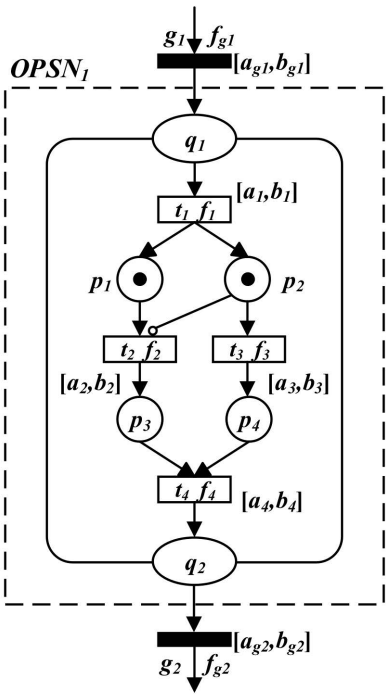

Figure 2. An example of the OOPIRES+ object subnet.

Here, $P = \{p_1, p_2, p_3, p_4\}$, and $T = \{t_1, t_2, t_3, t_4\}$; $F_I = \{(q_1, t_1), (p_1, t_2), (p_2, t_3), (p_3, t_4), (p_4, t_4)\}$; $F_O = \{(t_1, p_1), (t_1, p_2), (t_2, p_3), (t_3, p_4), (t_4, q_2),\}$; $I = \{(p_2, t_2)\}$; $Q = \{q_1, q_2\}$, where q_1 is the input message place, and q_2 is the output message place; and g_1 and g_2 are the input gateway and the output gateway of the object subnet $OPSN_1$. Moreover, f_1, f_2, f_3, and f_4 are transition functions of transitions t_1, t_2, t_3, and t_4, respectively. Their corresponding transition delays are $[a_1, b_1]$, $[a_2, b_2]$, $[a_3, b_3]$, and $[a_4, b_4]$, respectively. In addition, f_{g1} and f_{g2} are the gateway functions of gateways g_1 and g_2, and their corresponding gateway delays are $\left[a_{g1}, b_{g1}\right]$ and $\left[a_{g2}, b_{g2}\right]$.

Definition 4. *An OOPIRES+ net system is a four-tuple* $OPN = (OPSN, G, F, M)$*, where* $OPSN = (OPSN_1, OPSN_2, \ldots, OPSN_s)$ *is the object subnet set of OOPIRES+;* $G = \{g_1, g_2, \ldots, g_t\}$ *is the set of all gateways; F includes* F_I *and* F_O*, where* $F_I \subseteq (Q \times G)$*,* $F_O \subseteq (G \times Q)$*; M is the marking; and the initial marking is* M_0*.*

Definition 5. *Suppose that* $OPN = (OPSN, G, F, M)$ *is an OOPIRES+ net system,* $M \in R(M_0)$*, and* t^* *is the inhibitor place set of transition, t.*

(i) *If* $t^* \neq \varnothing$ *and* $\cdot t \cap Q \neq \varnothing$*, then t is called enabled under M if* $\forall p \in \cdot t - t^* : M(p) \geq W(p,t), \forall p \in t^* : M(p) = \varnothing$*, and* $\forall q \in Q : M(q) \geq W(q,t)$*;*

(ii) *If* $t^* = \varnothing$ *and* $\cdot t \cap Q \neq \varnothing$*, then t is called enabled under M if* $\forall p \in \cdot t : M(p) \geq W(p,t)$ *and* $\forall q \in Q : M(q) \geq W(q,t)$*;*

(iii) *If* $t^* \neq \varnothing$ *and* $\cdot t \cap Q = \varnothing$ *and, then t is called enabled under M if* $\forall p \in \cdot t - t^* : M(p) \geq W(p,t), \forall p \in t^* : M(p) = \varnothing$*;*

(iv) *If* $t^* = \varnothing$ *and* $\cdot t \cap Q = \varnothing$*, then t is called enabled under M if* $\forall p \in \cdot t : M(p) \geq W(p,t)$*;*

(v) *The system marking changes after the firing of transition* $t: M \rightarrow M'$*, where*

$$M'(p) = \begin{cases} M(p) - W(p,t) & p \in \cdot t - t\cdot - t^* \\ M(p) + W(t,p) & p \in t\cdot - \cdot t \text{ or } p \in t\cdot \cap t^* \\ M(p) - W(p,t) + W(t,p) & p \in \cdot t \cap t\cdot - t^* \\ M(p) & others \end{cases}$$

$$M'(q) = \begin{cases} M(q) - W(q,t) & p \in \cdot t \\ M(q) + W(t,q) & p \in t\cdot \\ M(q) & others \end{cases}$$

Definition 6. *Suppose an OOPIRES+ net system is* $OPN = (OPSN, G, F, M_0)$*, where* M_0 *is the initial marking,* $t \in OPSN(T)$*,* and $g \in G$*. Then, we have the following:*

(i) *If* $\forall M \in R(M_0)$*,* $\exists M' \in R(M)$*,* $M'[t >$*, then t is said to be live. If* $\forall M \in R(M_0)$*,* $\exists M' \in R(M)$*,* $M'[g >$*, then g is said to be live.*

(ii) *If* $\forall t \in OPSN(T)$*,*$\forall g \in G$*, t, and g are live, then the OOPIRES+ net system OPN is said to be live.*

Definition 7. *Suppose an OOPIRES+ net system is* $OPN = (OPSN, G, F, M_0)$*, where* M_0 *is the initial marking,* $p \in OPSN(P)$*, and* $q \in OPSN(Q)$*. Then, we have the following:*

(i) *Place p is said to be bounded if* $\exists k > 0, \forall M \in R(M_0) : M(p) \leq k$*. Message place q is said to be bounded if* $\exists l > 0, \forall M \in R(M_0) : M(q) \leq l$*.*

(ii) *If* $\forall p \in OPSN(P)$*,*$\forall q \in OPSN(Q)$*,p, and q are bounded, then the OOPIRES+ net system OPN is said to be bounded.*

Definition 8. *Assuming that* $OPSN_1$ *and* $OPSN_2$ *are two OOPIRES+ object subnets, then* $OPSN_1$ and $OPSN_2$ *are said to have the same reachability if and only if the following takes place:*

(i) *The pre-set place's number of input gateway and the post-set place's number of output gateway of* $OPSN_1$ *are the same as those of* $OPSN_2$*.*

(ii) *If the number of tokens in pre-set places of the input gateway of* $OPSN_1$ *is equal to that of* $OPSN_2$*, then the number of tokens in post-set place of the output gateway of* $OPSN_1$ *is equal to that of* $OPSN_2$*.*

Definition 9. *Assuming that* $OPSN_1$ *and* $OPSN_2$ *are two OOPIRES+ object subnets, then* $OPSN_1$ *and* $OPSN_2$ *are said to have the same functionality if and only if the following takes place:*

(i) $OPSN_1$ *and* $OPSN_2$ *have the same reachability.*

(ii) *If the token type of tokens in pre-set places of the input gateway of* $OPSN_1$ *is equal to that of* $OPSN_2$*, then the token type of tokens in the post-set place of output gateway of* $OPSN_1$ *is equal to that of* $OPSN_2$*.*

Definition 10. *Assuming that $OPSN_1$ and $OPSN_2$ are two OOPIRES+ object subnets, $OPSN_1$ and $OPSN_2$ are said to have the same timing if and only if the following takes place:*

(i) *$OPSN_1$ and $OPSN_2$ have the same reachability.*
(ii) *If the token time of tokens in pre-set places of the input gateway of $OPSN_1$ is equal to that of $OPSN_2$, then the token time of tokens in the post-set place of output gateway of $OPSN_1$ is equal to that of $OPSN_2$.*

3. OOPIRES+ Gateway Refinement Operation and Property Analysis

In this section, the gateway refinement operation of OOPIRES+ is proposed, and the preservation of liveness, boundedness, reachability, functionality, and timing of the original net system by this refinement operation is investigated.

Figure 3 is a schematic diagram of the gateway refinement operation of the OOPIRES+ net system.

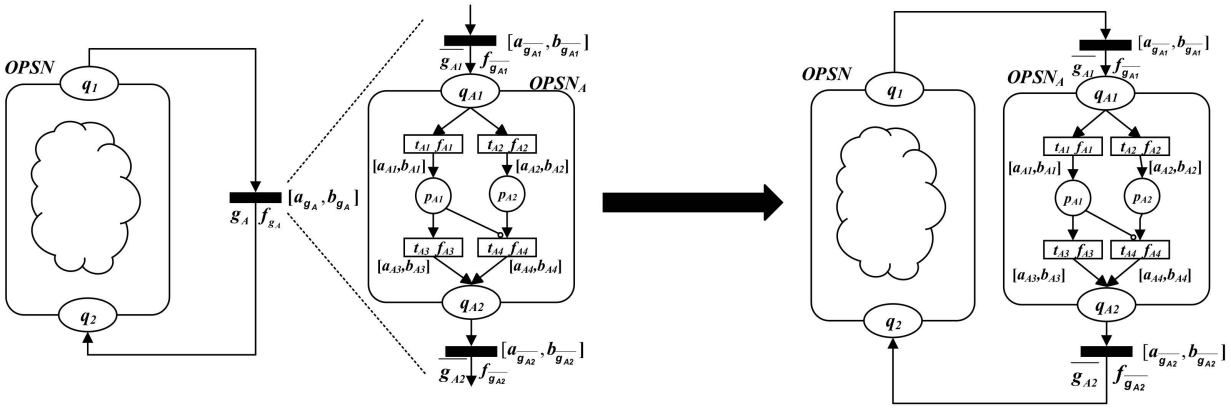

Figure 3. Gateway refinement operation schematic diagram of the OOPIRES+ net system.

Definition 11. *Gateway refinement operation of the OOPIRES+ net system $Ref(g_A, OPSN_A + \overline{G_A})$: The gateway, g_A, of the OOPIRES+ net system $OPN = (OPSN, G, F, M)$ is refined into a subnet and a gateway set, $\overline{G_A} = \{\overline{g_{A1}}, \overline{g_{A2}}\}$; that is, g_A is replaced by the subnet, $OPSN_A$, and the gateway set $\overline{G_A} = \{\overline{g_{A1}}, \overline{g_{A2}}\}$. Then, a new OOPIRES + net system, $OPN_B = (OPSN_B, G_B, F_B, M_B)$, is obtained, where we have the following:*

(i) $OPSN_B = OPSN \cup OPSN_A$;
(ii) $G_B = G \cup \overline{G_A} - g_A$;
(iii) $f_{gB} = f_{\overline{g_{A2}}} \circ f_s \circ f_{\overline{g_{A1}}} = f_{gA}$ *(here, $\circ$ is the function composition operator), where $f_{\overline{g_{A1}}}$, $f_{\overline{g_{A2}}}$, and f_{g_A} are the gateway functions of $\overline{g_{A1}}$, $\overline{g_{A2}}$, and g_A, respectively; $f_s : \tau(p_1) \times \tau(p_2) \times \ldots \times \tau(p_n) \to \tau(q)$, $p_1, p_2, \ldots, p_n \in OPSN_A(Q_{AI})$, and $q \in OPSN_A(Q_{AO})$.*
(iv) $d^-_{g_B} = d^-_{\overline{g_{A1}}} + d^-_s + d^-_{\overline{g_{A2}}} = d^-_{g_A}$, *and* $d^+_{g_B} = d^+_{\overline{g_{A1}}} + d^+_s + d^+_{\overline{g_{A2}}} = d^+_{g_A}$, *where $d^-_{\overline{g_{A1}}}$, $d^-_{\overline{g_{A2}}}$, and $d^-_{g_A}$ are the minimum gateway delays of $\overline{g_{A1}}$, $\overline{g_{A2}}$, and g_A, respectively; $d^+_{\overline{g_{A1}}}$, $d^+_{\overline{g_{A2}}}$, and $d^+_{g_A}$ are the maximum gateway delays of $\overline{g_{A1}}$, $\overline{g_{A2}}$, g_A, respectively; and d^-_s and d^+_s are the minimum and the maximum transition delays of $OPSN_A(T)$.*
(v) F_B *includes F_{BI} and F_{BO}, where* $F_{BI} = F_I \cup F_{AI} \cup \{(q, \overline{g_{A1}}) | q \in \cdot g_A\} \cup \{(q, \overline{g_{A2}}) | q \in \cdot \overline{g_{A2}}\} - \{(q, g_A) | q \in \cdot g_A\}$, *and* $F_{BO} = F_O \cup F_{AO} \cup \{(\overline{g_{A2}}, q) | q \in g_A \cdot\} \cup \{(\overline{g_{A1}}, q) | q \in \overline{g_{A1}} \cdot\} - \{(g_A, q) | q \in g_A \cdot\}$;
(vi) $M_B(p) = \begin{cases} M_0(p) & p \in OPSN(P) \\ M_{A0}(p) & p \in OPSN_A(P) \end{cases}$.

Definition 12. *The closed net system of the OOPIRES+ subnet $OPSN = (P, T; F, I, Q, W)$ is defined as a three-tuple $OPN' = (OPSN, G', p')$; that is, the gateway set, $G' = \{g'_1, g'_2\}$, and message place, p', are added to OPSN, where g'_1 is the input gateway of OPSN, g'_2 is the output gateway of OPSN, and $p' = \{g'_2\} \wedge p' \cdot = \{g'_1\}$.*

Definition 13. *The extended subnet of the gateway, g, is a three-tuple $g'' = (g, Q, F)$; that is, a message place set, Q, and a flow relationship, F, are added to the gateway, g, where Q includes Q_I and Q_O, and $F = \{(q_1, g), (g, q_2) | q_1 \in Q_I, q_2 \in Q_O\}$.*

In the following, we investigate the preservation of liveness, boundedness, reachability, functionality, and timing of the gateway refinement operation of the OOPIRES+ net system.

It is assumed that the OOPIRES+ net system $OPN_B = (OPSN_B, G_B, F_B, M_B)$ is obtained from the OOPIRES+ net system $OPN = (OPSN, G, F, M)$ by using the gateway refinement operation $Ref(g_A, OPSN_A + \overline{G_A})$.

Theorem 1. *OPN_B is live if and only if OPN and the closed net system $OPN_A{}' = (OPSN_A, G'_A, p'_A)$ of $OPSN_A$ are live.*

Proof. ($\Leftarrow$) For the net system OPN, since OPN is live, according to Definition 6, $\forall t \in OPSN(T)$ and $\forall g \in G$, for $\forall M \in R(M_0)$, there exists $M' \in R(M)$, such that $M'[t >$ and $M'[g >$, and then t and g are live. For the closed net system $OPN_A{}'$, since $OPN_A{}'$ is live, according to Definition 6, $\forall t \in OPSN_A(T)$ and $\forall g \in G'_A$, and for $\forall M \in R(M_0)$, there exists $M' \in R(M)$, such that $M'[t >$ and $M'[g >$; then, t and g are live. Since OPN_B is obtained from OPN by using $OPSN_A$ and $\overline{G_A}$ to replace g_A, $OPSN_B(T) = \{OPSN(T), OPSN_A(T)\}$, $OPN_B(G) = \{OPN(G), OPN_A(G'_A)\}$. According to Definition 12, for $\forall t \in OPSN_B(T)$, $\forall g \in G_B$, $\forall M \in R(M_{B0})$, $\exists M' \in R(M)$, such that $M'[t >$, $M'[g >$, t, and g are live. Therefore, the net system OPN_B is live.

($\Rightarrow$) Suppose that OPN_B is live. Without loss of generality, assume that OPN is not live, and then $\exists t \in OPSN(T)$ or $\exists g \in G$, $\exists M \in R(M_0)$, $\forall M' \in R(M)$, such that $\neg M'[t >$ or $\neg M'[g >$. Since $OPSN(T) \subseteq OPSN_B(T)$ and $G \subseteq G_B$, then $\exists t \in OPSN_B(T)$ or $\exists g \in G_B$, $\exists M \in R(M_{B0})$, $\forall M' \in R(M)$, $\neg M'[t >$ or $\neg M'[g >$, so OPN_B is not live. This contradicts the hypothesis. Therefore, OPN and $OPN_A{}'$ are live. □

Theorem 2. *OPN_B is bounded if and only if OPN and the closed net system $OPN_A{}' = (OPSN_A, G'_A, p'_A)$ of $OPSN_A$ are bounded.*

Proof. ($\Leftarrow$) Since the net system OPN is bounded, according to Definition 7, $\forall p \in OPSN(P) \wedge \forall q \in OPSN(Q)$, $\exists k > 0$, $\forall M \in R(M_0)$, such that $M(p) \leq k \wedge M(q) \leq k$. Since the closed net system $OPN_A{}'$ is bounded, according to Definition 7, $\forall p \in OPSN_A(P) \cup \{p'_A\} \wedge \forall q \in OPSN_A(Q)$, $\forall M \in R(M_{A0})$, $\exists k_A > 0$, such that $M(p) \leq k_A \wedge M(q) \leq k_A$. Let $k_B = \max\{k, k_A\}$ and then $\forall p \in OPSN_B(P) \wedge \forall q \in OPSN_B(Q)$,$\forall M \in R(M_{B0})$, such that $M(p) \leq k_B \wedge M(q) \leq k_B$. Thus, OPN_B is bounded.

($\Rightarrow$) Suppose that OPN_B is bounded. Without loss of generality, assume that OPN is not bounded, and then, for $\forall M \in R(M_0)$, $\exists p \in OPSN(P)$ or $\exists q \in OPSN(Q)$, for $\forall k > 0$, $M(p) > k$ or $M(q) > k$. Since $OPSN(P) \subseteq OPSN_B(P)$ and $OPSN(Q) \subseteq OPSN_B(Q)$, for $\forall M \in R(M_{B0})$, $\exists p \in OPSN_B(P)$ or $\exists q \in OPSN_B(Q)$, $\forall k_B > 0$, $M(p) > k$ or $M(q) > k$, so OPN_B is not bounded. This contradicts the hypothesis. Therefore, OPN and $OPN_A{}'$ are both bounded. □

Theorem 3. *OPN_B and OPN have the same reachability.*

Proof. According to gateway refinement operation, $Ref(g_A, OPSN_A + \overline{G_A})$, of the OOPIRES + system, in $OPSN_A + \overline{G_A}$ and g''_A, the number of pre-set places and post-set places of $\overline{G_A}$ and g_A is the same, and the number of tokens in the pre-set places of $\overline{G_A}$ is equal to that of g_A. According to Definition 11, the tokens' number in the post -set places of $\overline{G_A}$ is equal to that of g_A. According to Definition 8, the reachability of g''_A and $OPSN_A + \overline{G_A}$ is the same. Since $OPN - \{g_A\} = OPN_B - \{OPSN_A + \overline{G_A}\}$, the reachability of OPN_B and OPN is the same. □

Theorem 4. *OPN_B and OPN have the same functionality.*

Proof. According to Theorem 3, net system OPN_B and OPN have the same reachability. In $OPSN_A + \overline{G_A}$ and g''_A, the token type of tokens in the pre-set places of $\overline{G_A}$ is the same as that of g_A. According to Definition 11, $f_{g_B} = f_{\overline{g_{A2}}} \circ f_s \circ f_{\overline{g_{A1}}} = f_{g_A}$; therefore, the token type of tokens in post-set places of $\overline{G_A}$ is the same as that of g_A. By Definition 9, $OPSN_A + \overline{G_A}$ and g''_A have the same functionality. Since $OPN_B - \{OPSN_A + \overline{G_A}\} = OPN - \{g_A\}$, OPN_B and OPN have the same functionality. □

Theorem 5. *OPN_B and OPN have the same timing.*

Proof. According to Theorem 3, net systems OPN_B and OPN have the same reachability. In $OPSN_A + \overline{G_A}$ and g''_A, the token time of tokens in the pre-set places of $\overline{G_A}$ is the same as that of g_A. According to Definition 11, $d^-_{\overline{g_{A1}}} + d^-_s + d^-_{\overline{g_{A2}}} = d^-_{g_A}$, $d^+_{\overline{g_{A1}}} + d^+_s + d^+_{\overline{g_{A2}}} = d^+_{g_A}$; therefore, the token time of tokens in post-set places of $\overline{G_A}$ is the same as that of g_A. By Definition 10, $OPSN_A + \overline{G_A}$ and g''_A have the same timing. Since $OPN_B - \{OPSN_A + \overline{G_A}\} = OPN - \{g_A\}$, then OPN_B and OPN have the same timing. □

4. Applications

This section applies the gateway refinement method of the object-oriented PIRES+ (OOPIRES+) to the modeling and analysis of smart home remote control systems.

The smart home remote control system is mainly composed of five functional modules: mobile control module, PC control module, instruction data transmission module, function echo feedback module, and system function control module. When users want to manage their smart homes, they can give instructions to the smart home devices connected to the network, one by one, through the mobile phones or PC terminals. Instructions are summarized by the mobile control module or PC control module and uploaded to the cloud for storage. The instructions are transmitted to the system function control module through the instruction data transmission module, achieving management and control of the smart home, and ultimately completing the feedback of the instructions.

4.1. Construct OOPIRES+ Model for Smart Home Remote Control System

In this subsection, the smart home remote control system's OOPIRES+ model is constructed. We first provide the OOPIRES+ abstract model and then apply the above refinement operation method to obtain its refined OOPIRES+ model.

As shown in Figure 4, the abstract OOPIRES+ net system model $OPN = (OPSN, G, F, M)$ of the smart home remote control system is proposed, which includes instructions transmission module $OPSN_1$, function feedback module $OPSN_2$, and gateways g_A and g_{AA}. The meaning of the internal transition of $OPSN_1$ and $OPSN_2$ is as follows: t_1, receive current instructions; t_2, remote transmission of instructions; t_3, instruction transmission completed; t_4, receive functional feedback; t_5, functional feedback transmission; and t_6, screen status display.

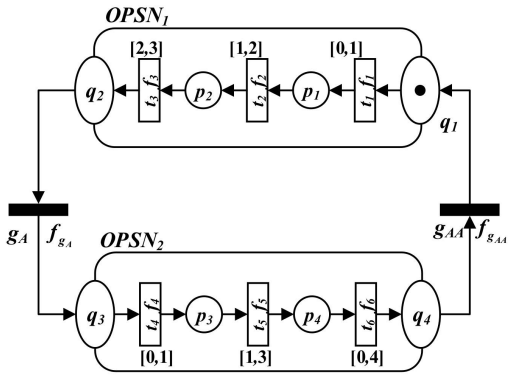

Figure 4. Abstract model OPN of the system.

Figure 5 shows the object subnet model $OPSN_A$, which includes system function control module $OPSN_3$ and gateways $\overline{g_{A1}}$ and $\overline{g_{A2}}$. According to the received instructions, $OPSN_A$ completes the control of subsystems. The meaning of the internal transition $OPSN_3$ is as follows: t_{A1}, apply for control permission; t_{A2}, confirm system status; t_{A3}, confirm control permission; and t_{A4}, complete instruction execution. Moreover, $t_{A1'}$~$t_{Am'}$ corresponds to control systems, which include the lighting control system, monitoring control system, and electrical control system, and realizes corresponding instruction functions.

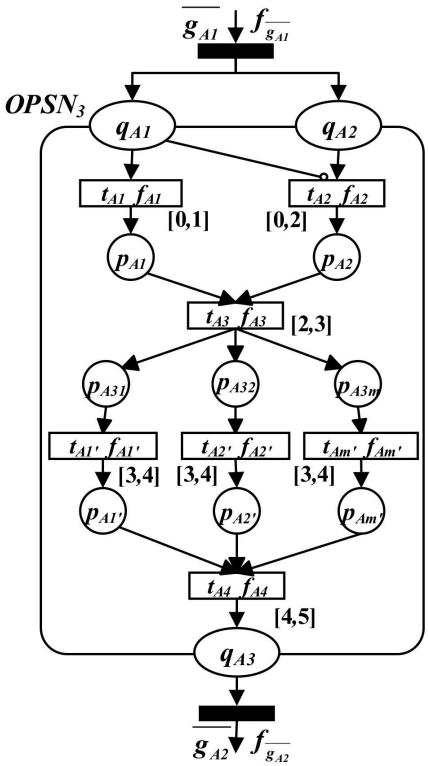

Figure 5. The object subnet $OPSN_A$.

Figure 6 shows the object subnet $OPSN_{AA}$, including mobile terminal control module $OPSN_4$, PC control module $OPSN_5$, and the gateways $\overline{g_{AA1}}$ and $\overline{g_{AA2}}$. According to the operator's intention, this module sends instructions to the functional control terminal through different devices. The meaning of the internal transitions of $OPSN_4$ and $OPSN_5$ is as follows: t_{AA1}, apply to activate mobile devices; t_{AA2}, summarize current instructions; t_{AA3}, check network connectivity; t_{AA4}, upload instructions to the cloud; t_{AA5}, complete cloud upload; t_{AA6}, apply to enable PC devices; t_{AA7}, check synchronization on PC devices; t_{AA8}, confirm synchronization on PC devices; t_{AA9}, summarize current instructions; and t_{AA10}, instruction cloud synchronization. Moreover, $t_{AA1'}$~$t_{AAm'}$ and $t_{AA1''}$~$t_{AAm''}$ are used to realize the collection of system instructions on a mobile phone and PC terminal.

Next, the object subnets $OPSN_A$ and $OPSN_{AA}$ are used to carry out refinement operations on g_A and g_{AA} in the net system abstract model OPN, and the smart home remote control system OPN_B is obtained (as shown in Figure 7). The system begins to issue control instructions to smart home devices from $OPSN_4$ or $OPSN_5$. $OPSN_1$ receives the instructions and transmits them remotely to $OPSN_3$. $OPSN_3$ receives the instructions and executes them. $OPSN_2$ is used to control the screen display and remind the users of their smart home status.

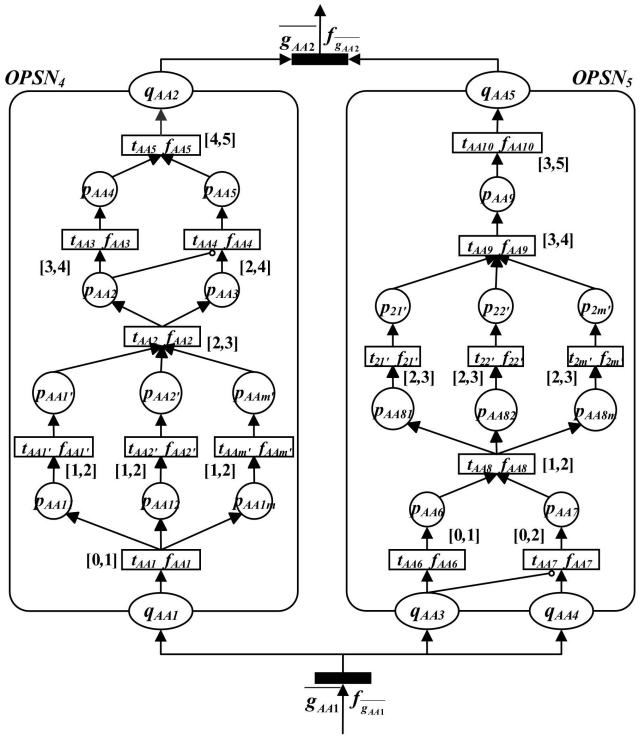

Figure 6. The object subnet $OPSN_{AA}$.

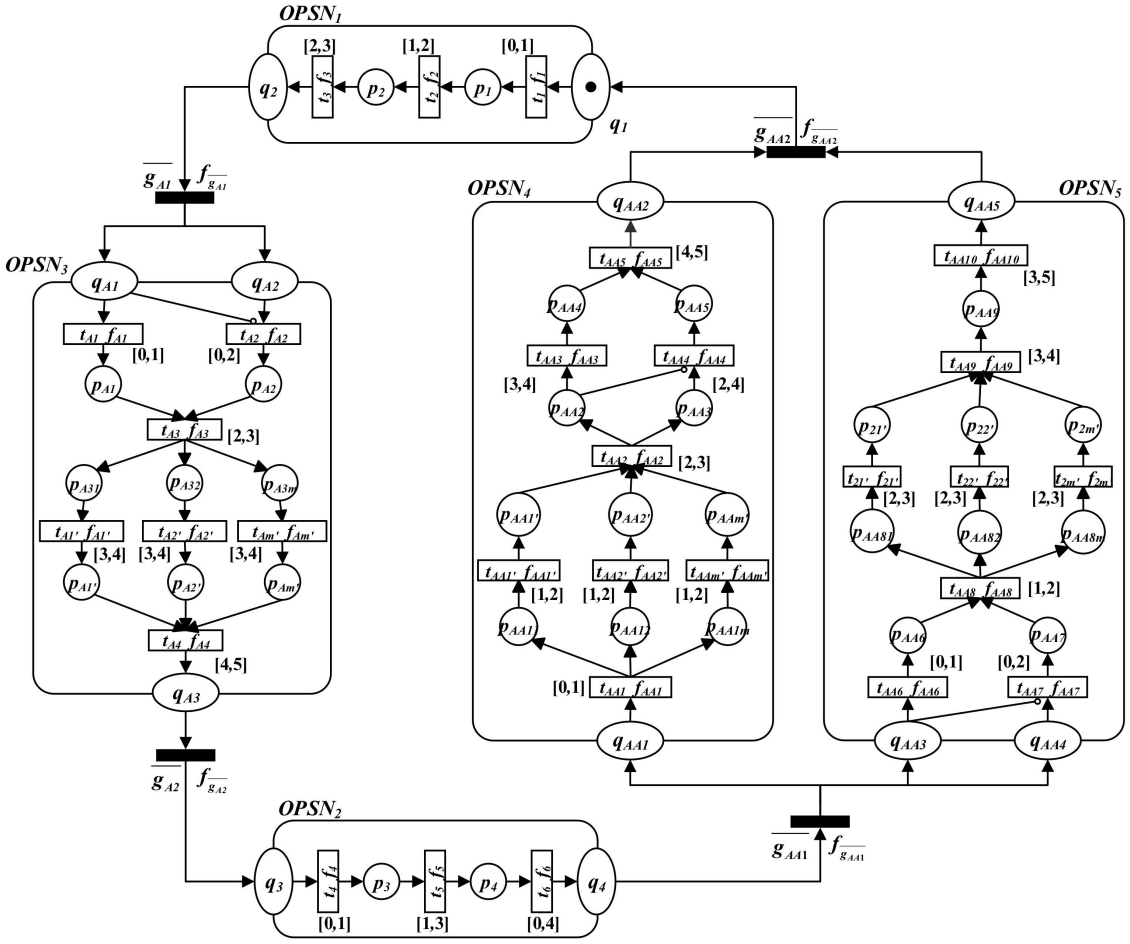

Figure 7. The refined smart home remote control system OPN_B.

4.2. Property Preservation Analysis of the OOPIRES+ Model

In this subsection, the property preservations of the smart home remote control system's OOPIRES+ model are analyzed.

4.2.1. Liveness and Boundedness Preservation

As can be seen from Figure 5, the net system OPN is live and bounded, and the corresponding closed net systems can be obtained by adding message places on the basis of the models of $OPSN_A$ and $OPSN_{AA}$ in Figures 5 and 6. It is easy to see that the closed net systems are also live and bounded. The net system OPN_B is obtained from the net system OPN by using $OPSN_A$ and $OPSN_{AA}$ to replace g_A and g_{AA}, respectively. Therefore, according to Theorems 1 and 2, the net system OPN_B is also live and bounded. As can be seen from Figures 4 and 7, OPN and OPN_B are both live and bounded. In addition, the liveness and boundedness of OPN (Figure 4) and OPN_B (Figure 7) can also be verified by the modeling tool Tina 3.7.0, and the verification results are shown in Figures 8 and 9.

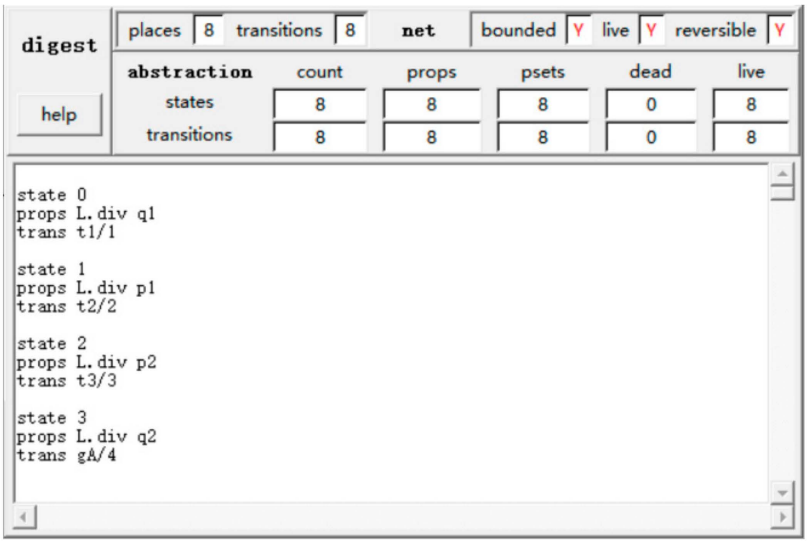

Figure 8. Verification result of liveness and boundedness of OPN.

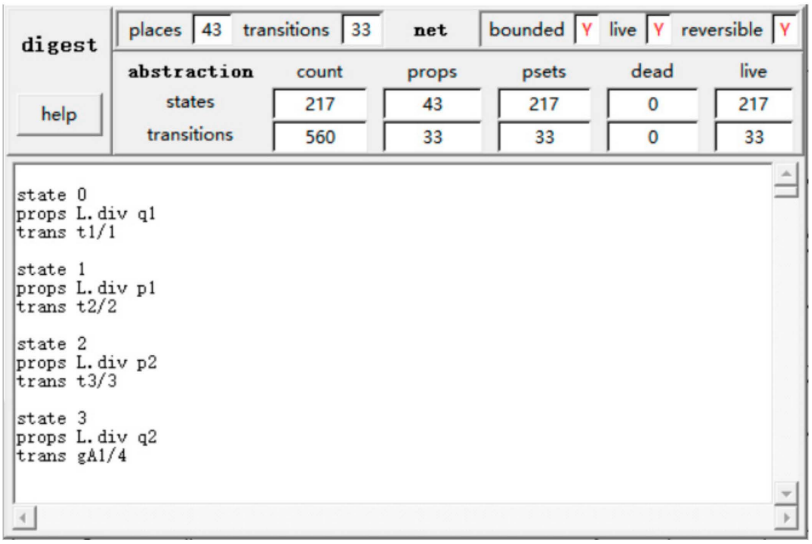

Figure 9. Verification result of liveness and boundedness of OPN_B.

4.2.2. Reachability, Functionality, and Timing Preservation

In Figures 4 and 7, the pre-set places of g_A and $\overline{g_{A1}}$ are q_2, and the post-set places of g_A and $\overline{g_{A2}}$ are q_3. Since g_A and $\overline{g_{A1}}$ have the same number of pre-set places and the same number of tokens, and after the refinement operation, g_A and $\overline{g_{A2}}$ have the same number of post-set places and the same number of tokens, g_A's extended subnet has the same reachability as subnet $OPSN_A$. Since the token type of tokens in the pre-set places of g_A and $\overline{g_{A1}}$ is the same; the gateway function of g_A is equal to the function of $OPSN_A$; and, after the refinement operation, the token type of tokens in the post-set places of g_A and $\overline{g_{A2}}$ is the same, g_A's extended subnet has the same functionality as subnet $OPSN_A$. Since the token time of tokens in the pre-set places of g_A and $\overline{g_{A1}}$ is the same; the gateway delay of g_A is equal to the delay of $OPSN_A$; and, after the refinement operation, the token time of tokens in the post-set places of g_A and $\overline{g_{A2}}$ is the same, g_A's extended subnet has the same timing as subnet $OPSN_A$. Similarly, the extended subnet of g_{AA} and the subnet $OPSN_{AA}$ have the same reachability, functionality, and timing. Thus, by Theorems 3, 4, and 5, OPN_B and OPN have the same reachability, functionality, and timing. According to Figures 4 and 7, it is easy to see that OPN_B and OPN have the same reachability, functionality, and timing.

Note that the gateway refinement method can be used for the modeling and validation of various embedded-system instances, such as intelligent transportation, smart cities, smart homes, etc. The modeling and verification of the smart home remote control system is only one of many system examples. The application of the refinement method to model and validate different system instances may vary, and a specific analysis should be conducted on specific issues. However, the basic modeling and validation process is similar.

5. Conclusions

For embedded system modeling, common formal modeling methods, such as the extended finite-state machine method, data flow diagram method, communication process method, etc., can describe the characteristics of embedded systems from different perspectives but do not form a unified standard. Classical Petri nets lack a hierarchical structure and the concept of time, making it impossible to fully describe embedded systems. PRES+ improved the accuracy of embedded system modeling, but there are still issues with the inability to describe event priorities and fully express complex program control and data flow. PIRES+ can consider the system and behavior uniformly, but it is not easy to reflect the division of functional modules.

In response to the problem of formal modeling and analysis of large-scale complex embedded systems, object-oriented technology is combined with PIRES+, and an object-oriented PIRES+ (OOPIRES+) and a gateway refinement operation method are proposed. This paper investigates the preservation of the liveness, boundedness, reachability, functionality, and timing of the original net system by the gateway refinement operation and presents the preservation conditions for the above important properties of the refinement operation, which can effectively alleviate the "state space explosion" problem of OOPIRES+. The application in the refined modeling and analysis of smart home remote control systems also demonstrates the effectiveness and practicality of the proposed refinement method. The research results of this paper will also provide a new method for modeling and analysis of large complex embedded systems.

The limitation of the gateway refinement approach is that the operation must meet the constraints of gateway replacement, gateway function, gateway delay, and so on. The next research work may consider proposing a wider range of conditions for the study of OOPIRES+ gateway refinement operations or other refinement operations (such as a subnet refinement operation).

Author Contributions: Conceptualization, C.X.; methodology, C.X.; validation, C.X., M.Q. and Y.S.; formal analysis, C.X. and M.Q.; investigation, C.X.; resources, C.X.; writing—original draft preparation, C.X.; writing—review and editing, M.Q., Y.S. and M.G.; supervision, C.X.; project administration, C.X.; funding acquisition, C.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Natural Science Foundation of Shandong Province (Grant No. ZR2022MF348).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Roberto, B.; Montserrat, M.; María, E.B.G.; Ana, F.G. An Event Mesh for Event Driven IoT Applications. *Int. J. Interact. Multimed. Artif. Intell.* **2022**, *7*, 54–59.
2. Marwedel, P. *Embedded System Design—Embedded Systems Foundations of Cyber-Physical Systems, and the Internet of Things*, 4th ed.; Springer: Cham, Switzerland, 2021.
3. Tausan, N.; Markkula, J.; Kuvaja, P.; Oivo, M. Choreography in the embedded systems domain: A systematic literature review. *Inf. Softw. Technol.* **2017**, *91*, 82–101. [CrossRef]
4. Stoutchini, A.; Benini, L. StreamDrive: A dynamic dataflow framework for clustered embedded architectures. *J. Signal Process. Syst.* **2019**, *91*, 630–640.
5. Firdaus, A.F.; Meribout, M. A new parallel VLSI architecture for real-time electrical capacitance tomography. *IEEE Trans. Comput.* **2016**, *65*, 30–41. [CrossRef]
6. Ding, Z.; Yang, R.; Cui, P.; Zhou, M.C.; Jiang, C. Variable Petri nets for mobility. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 4784–4797. [CrossRef]
7. Ciccozzi, F.; Malavolta, I.; Selic, B. Execution of UML models: A systematic review of research and practice. *Softw. Syst. Model.* **2019**, *18*, 2313–2360. [CrossRef]
8. Moutinho, F.; Gomes, L. Asynchronous-channels within Petri net-based GALS distributed embedded systems modeling. *IEEE Transections Ind. Inform.* **2014**, *10*, 2024–2033. [CrossRef]
9. Andrzej, B.; Tomasz, R.; Dariusz, R. Timed colored Petri net-based event generators for web systems simulation. *Appl. Sci.* **2022**, *12*, 12385.
10. Borstel, F.D.V.; Villa-Medina, J.F.; Gutiérrez, J. Development of mobile robots based on wireless robotic components using UML and hierarchical colored Petri nets. *J. Intell. Robot. Syst.* **2022**, *104*, 70. [CrossRef]
11. Gutierrez, A.; Bressan, M.; Jimenez, J.F.; Alonso, C. Real-time emulation of boost inverter using the systems modeling language and Petri nets. *Math. Comput. Simul.* **2019**, *158*, 216–234. [CrossRef]
12. Jiang, W.; Zhou, K.-Q.; Sarkheyli-Hägele, A.M.; Zain, A. Modeling, reasoning, and application of fuzzy Petri net model: A survey. *Artif. Intell. Rev.* **2022**, *55*, 6567–6605. [CrossRef]
13. Cortés, L.A.; Eles, P.; Peng, Z. Modeling and formal verification of embedded systems based on a Petri net representation. *J. Syst. Archit.* **2003**, *49*, 571–598. [CrossRef]
14. Karlsson, D.; Eles, P.; Peng, Z. Formal verification of component-based designs. *J. Des. Autom. Embed. Syst.* **2007**, *11*, 49–90. [CrossRef]
15. Karlsson, D.; Eles, P.; Peng, Z. Model validation for embedded systems using formal method-aided simulation. *LET Comput. Digit. Tech.* **2008**, *2*, 413–433. [CrossRef]
16. Bandyopadhyay, S.; Sarkar, S.; Sarkar, D.; Mandal, C. SamaTulyata: An efficient path based equivalence checking Tool. In *Automated Technology for Verification and Analysis, Proceedings of the 15th International Symposium, ATVA 2017, Pune, India, 3–6 October 2017*; D'Souza, D., Narayan Kumar, K., Eds.; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2017; Volume 10482, pp. 109–116.
17. Bandyopadhyay, S.; Sarkar, D.; Mandal, C. Equivalence checking of petri net models of programs using static and dynamic cut-points. *Acta Inform.* **2019**, *56*, 321–383. [CrossRef]
18. Xia, C.; Li, C. Property preservation of Petri synthesis net based representation for embedded systems. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 905–915. [CrossRef]
19. Xia, C.; Wang, Z.; Wang, Z. The refinement of Petri net with inhibitor arcs based representation for embedded systems. *Electronics* **2022**, *11*, 1389. [CrossRef]
20. Zhao, N.; Yu, Y.; Wang, J.; Xie, Z.W.; Yang, Z.D.; Cheng, L. An extended object-oriented Petri net supporting the description and evolution of components: EOOPN. *Clust. Comput.* **2019**, *22*, 2701–2708. [CrossRef]
21. Hu, H.; Yu, J.; Li, Z.; Chen, J.; Hu, H. Modeling and analysis of cyber–physical system based on object-oriented generalized stochastic Petri net. *IEEE Trans. Reliab.* **2021**, *70*, 1271–1285. [CrossRef]
22. Padberg, J.; Urášek, M. Rule-based refinement of Petri nets: A Survey. In *Petri Net Technology for Communication-Based SystemsEhrig*; Reisig, H., Rozenberg, W., Weber, G.H., Eds.; Springer: Berlin/Heidelberg, Germany, 2023; pp. 161–196.

23. Wang, S.; You, D.; Zhou, M.; Seatzu, C. Characterization of admissible marking sets in Petri nets with uncontrollable transitions. *IEEE Trans. Autom. Control.* **2016**, *61*, 1953–1958. [CrossRef]
24. Bernardinello, L.; Kılınç, G.; Mangioni, E.; Pomello, L. Modeling distributed private key generation by composing Petri nets. In *Transactions on Petri Nets and Other Models of Concurrency IX*; Koutny, M., Haddad, S., Yakovlev, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; pp. 19–40.
25. Huang, H.; Cheung, T.-y.; Mak, W.M. Structure and behavior preservation by Petri-net-based refinements in system design. *Theor. Comput. Sci.* **2004**, *328*, 245–269. [CrossRef]
26. Lakos, C.; Lewis, G. Incremental state space construction of colored Petri nets. In Proceedings of the International Conference on Application and Theory of Petri Nets, Virtual Event, 23–25 June 2021; Colom, J.-M., Koutny, M., Eds.; Springer: Berlin/Heidelberg, Germany, 2021; pp. 263–282.
27. Van Hee, K.M.; Sidorova, N.; van der Werf, J.M. Business process modeling using Petri nets. In *Transactions on Petri Nets and Other Models of Concurrency VII*; Jensen, K., Koutny, M., Balbo, G., Wolf, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 116–161.
28. Mejía, G.; Niño, K.; Montoya, C.; Sánchez, M.A.; Palacios, J.; Amodeo, L. A Petri net based framework for realistic project management and scheduling: An application in animation and videogames. *Comput. Oper. Res.* **2016**, *66*, 190–198. [CrossRef]