Sławomir SAMOLEJ, Bartosz TRYBUS
POLITECHNIKA RZESZOWSKA, KATEDRA INFORMATYKI I AUTOMATYKI

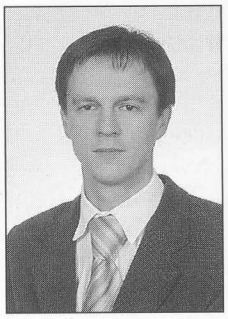
Zastosowanie kolorowanych sieci Petriego w projektowaniu systemów czasu rzeczywistego

Dr inż. Sławomir SAMOLEJ

Adiunkt w Katedrze Informatyki i Automatyki Politechniki Rzeszowskiej. Absolwent Wydziału Elektrycznego Politechniki Rzeszowskiej (1996). Stopień naukowy doktora uzyskał w 2004 r. na Wydziale Elektrotechniki, Automatyki, Informatyki i Elektroniki AGH. Zajmuje się zastosowaniami metod formalnych i inżynierskich w wytwarzaniu oprogramowania systemów czasu rzeczywistego.

e-mail: ssamolej@prz-rzeszow.pl

Dr inż. Bartosz TRYBUS



Absolwent specjalności Automatyka i Robotyka na Wydziale Elektrotechniki, Automatyki, Informatyki i Elektroniki AGH. W 2004 r. uzyskał tam również stopień doktora nauk technicznych w dyscyplinie informatyka. Od 1996 pracuje w Katedrze Informatyki i Automatyki Politechniki Rzeszowskiej. Zajmuje się projektowaniem systemów czasu rzeczywistego, w tym komputerowych systemów automatyki.

e-mail: btrybus@prz-rzeszow.pl

Streszczenie

W artykule przedstawiono metodę zastosowania kolorowanych sieci Petriego (sieci CPN) do wytwarzania systemów czasu rzeczywistego. Opis sytemu prowadzony jest przy użyciu metodyki SDRTS (Structured Design for Real-Time Systems) i sieci CPN. Ostatecznym rezultatem procesu projektowego jest zbiór hierarchicznych czasowych sieci CPN odpowiadających diagramom metodyki SDRTS, umożliwiający symulację i formalną analizę wytwarzanego systemu.

Abstract

A way of application of coloured Petri nets (CPN) in real-time systems development is presented in the paper. The system is described simultaneously using SDRTS (Structured Design for Real-Time Systems) method and CPN language. The final result of the development process is a set of hierarchical timed coloured Petri nets corresponding to SDRTS diagrams that makes it possible to simulate and formally analyse the system designed.

Słowa kluczowe: kolorowane sieci Petriego, systemy czasu rzeczywistego, inżynieria oprogramowania

Keywords: coloured Petri nets, real-time systems, software engineering

1. Wstęp

Modelowanie bardziej złożonych systemów przy pomocy klasycznych sieci Petriego wiąże się często z nadmiernym rozrostem sieci. Jest to główny powód wprowadzenia rozszerzonych sieci wyższego rzędu. Jednym z rodzajów takich sieci są kolorowane sieci Petriego (CPN - Coloured Petri Nets) [1]. Pozwalają one na bardziej zwięzłą i łatwiejszą do interpretacji reprezentację nawet dużych systemów. Kolorowane sieci Petriego rozszerzają zakres modelu wprowadzając do niego wartości przypisane do krążących znaczników oraz funkcje sterujące przepływem tych znaczników i obliczające nowe wartości. Możliwa jest również reprezentacja większych modeli w postaci zbioru powiązanych stron (arkuszy) - do tego celu stosuje się hierarchiczne kolorowane sieci Petriego (sieci HCPN). Z kolei czasowe kolorowane sieci Petriego (sieci TCPN) dostarczają notację uwzględniającą aspekty związane z upływem czasu.

Kolorowane sieci Petriego powstały jako narzędzie do modelowania i formalnej analizy szeroko rozumianych systemów. Stosunkowo duże możliwości specyfikacyjne tych sieci stały się podstawą opracowania prezentowanej dalej metody, która stosuje sieci CPN do projektowania systemów czasu rzeczywistego.

2. Specyfikacja wymagań funkcjonalnych

Tworzenie systemów informatycznych, w tym czasu rzeczywistego, rozpoczyna się zwykle od określenia wymagań funkcjonalnych, jakie musi spełniać projektowany system. Metody systematyzujące proces projektowy określają przy tym notację, najczęściej graficzną, za pomocą której wyrażane są wspomniane wymagania. Jedną z takich metod jest projektowanie strukturalne (*structured design*), opracowane początkowo dla klasycznych systemów przetwarzających dane. W związku z uzyskaniem przezeń znacznej popularności, powstało kilka rozszerzeń umożliwiających zastosowanie go do projektowania systemów czasu rzeczywistego, wśród których wyróżnia się metoda SDRTS (*Structured Design for Real-Time Systems*) opracowana przez P. Warda i W. Mellora [2]. Metody tego typu dominują w praktyce inżynierskiej. Ich cechą jest otwartość notacji graficznej, która umożliwia specyfikację cech nietypowych lub charakterystycznych dla danego zastosowania. Wadą jest jednak brak możliwości formalnej analizy właściwości modeli tworzonych w oparciu o metody projektowe. W przypadku systemów czasu rzeczywistego, szczególnie tych o podwyższonych wymogach bezpieczeństwa, może to być dużym ograniczeniem. Pojawiły się w związku z tym rozwiązania, których celem jest zapewnienie możliwości formalnej analizy.

W pracy [3] wykazano, że możliwe jest prowadzenie procesu projektowego łączącego metodykę strukturalną SDRTS Warda-Mellora z kolorowanymi sieciami Petriego CPN do tworzenia poprawnie funkcjonujących systemów czasu rzeczywistego. Podstawowym instrumentem projektowym pozostaje modelowanie strukturalne, co ułatwia zastosowanie metody w praktyce, głównie przemysłowej, i może przynieść poprawę jakości budowanych systemów. Głównym elementem łączonej metody jest konwersja modelu strukturalnego systemu w notacji SDRTS do sieci CPN. Sposób konwersji powstał w wyniku rozwinięcia metodyki opisanej w [4]. Umożliwiono objęcie konwersją szerszego zbioru elementów modelu SDRTS, m.in.:

- diagramów przepływu danych i sterowania, które są podstawowym elementem modelowania w SDRTS;
- hierarchii modelu strukturalnego z możliwością jej zachowania w hierarchicznej sieci HCPN;
- ujęcie w sieci TCPN cech czasowych, wyrażonych w sposób nieformalny w modelu SDRTS.

W wyniku konwersji modelu strukturalnego do kolorowanej sieci Petriego otwiera się dostęp do narzędzi formalnej weryfikacji poprawności, pojawia się możliwość symulowania pracy systemu oraz łatwiejszego przejścia do dalszych etapów jego rozwoju.

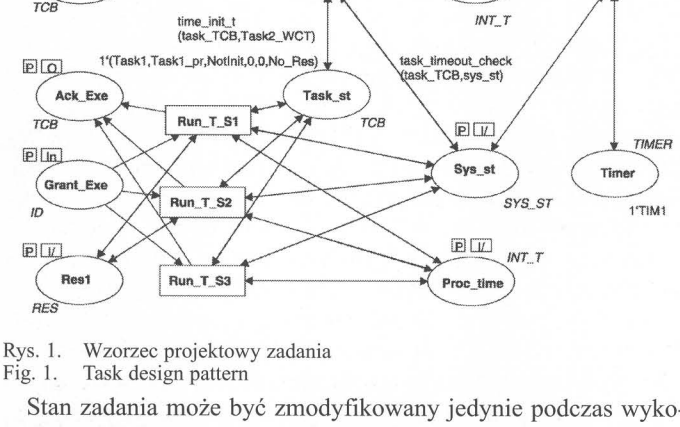
3. Specyfikacja zadań

Kolejnym etapem projektowania systemów czasu rzeczywistego jest zwykle dekompozycja wydzielonych składników oprogramowania na zbiór współbieżnie wykonywanych zadań czasu rzeczywistego [2, 5]. W pracy [6] wykazano, że język czasowych kolorowanych sieci Petriego (sieci TCPN) zastosować można również do sformułowania opisu oprogramowania z wymienionej perspektywy. Zaproponowano predefiniowane struktury sieci TCPN (wzorce projektowe) umożliwiające odwzorowanie indywidualnych czasowych cech zadania czasu rzeczywistego oraz powiązanie zbioru zadań ze zbiorem procesorów. Końcowym efektem formułowania

opisu systemu jest hierarchiczna sieć TCPN (sieć HTCPN) stanowiąca wykonywalny (w sensie symulacji) model projektowanego oprogramowania.

Przyjęto następujące założenia dotyczące zaproponowanego wzorca projektowego zadania. Zadania mogą się komunikować, a na poziomie pojedynczego systemu mikroprocesorowego poddawane są szeregowaniu. Wzorzec umożliwia zdefiniowanie: interfejsów pomiędzy zadaniem, a innymi komponentami systemu (inne zadania, moduł szeregujący, zasoby); mechanizmu inicjalizującego (cykliczne) obliczenia zadania; wydzielonych faz obliczeń zadania, w których wymagane są poszczególne zasoby.

Wzorzec projektowy zadania pokazano na rysunku 1. Bieżący stan zadania reprezentowany jest przez wartość znacznika w miejscu *Task_st*. W opisie zadania uwzględnia się jego identyfikator, priorytet, przydzielane zasoby oraz czas realizacji obliczeń.

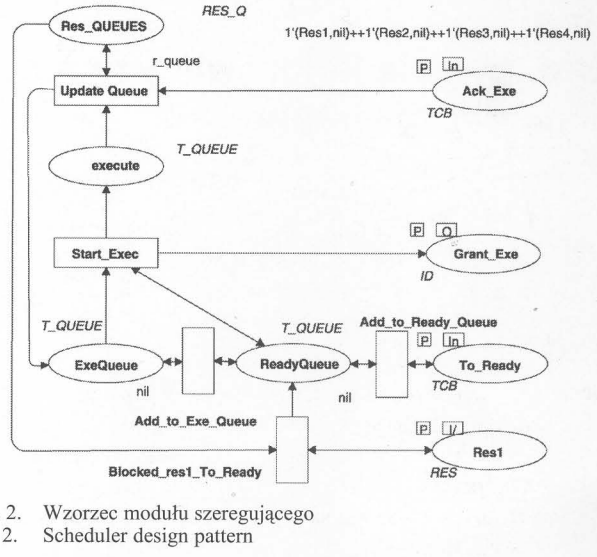


Rys. 1. Wzorzec projektowy zadania
Fig. 1. Task design pattern

Stan zadania może być zmodyfikowany jedynie podczas wykonania przejścia *Act_task* (aktywuj zadanie) lub jednego z przejść *Run_T_Sn* (gdzie n=1,2,...) modelujących wykonywanie określonych faz obliczeniowych przez dany proces (Rys. 1). Podczas wykonywania wybranych faz obliczeniowych zadanie może żądać dostępu do zasobów (por. miejsce *Res*1). Przewidziano również możliwość zablokowania zadania na zasobie oraz sterowania wykonywaniem obliczeń przez zewnętrzny moduł szeregujący (interfejs złożony z miejsc *Grant_Exe, Ack_Exe* i *To_Ready*), który opisano w następnym punkcie. Zbiór zadań przywiązanych do jednego systemu mikroprocesorowego współdzieli miejsce *Proc_time* umożliwiające modelowanie konsumpcji czasu procesora. Kolejne instancje zadania mogą być inicjalizowane (przez pojawienie się odpowiedniego znacznika w miejscu *Init_Port*) z zastosowaniem budzika złożonego z miejsca *Timer* oraz przejścia *Init_task*. Wzorzec projektowy przekształca się w model konkretnego zadania przez sprecyzowanie odpowiednich inskrypcji powiązanych z wybranymi miejscami, lukami i przejściami sieci wzorcowej [6].

4. Moduł szeregujący

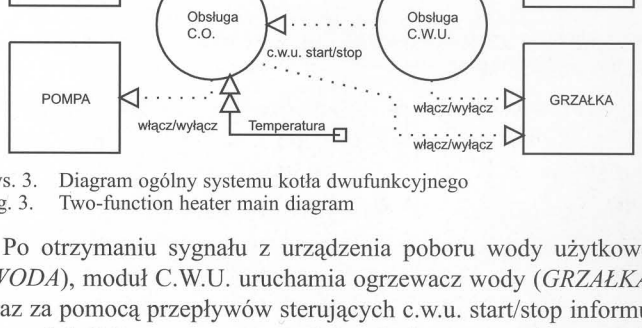
Wyspecyfikowany zbiór zadań czasu rzeczywistego, który ma być wykonywany na określonym procesorze powinien być powiązany z pojedynczym modułem szeregującym (Rys. 2). Moduł dokonuje cyklicznie wyboru tego zadania, które posiada najwyższy priorytet i nie jest zablokowane na zasobie, a następnie przydziela mu czas procesora na wykonanie kwantu obliczeń. Komunikacja pomiędzy zbiorem zadań a modułem szeregującym odbywa się przez miejsca *Grant_Exe* (udziel prawa wykonywania), *Ack_Exe* (potwierdź zakończenie wykonywania) i *To_Ready*. W miejscu *To_Ready* umieszczane są zadania zgłaszające gotowość do wykonywania. W dalszej kolejności przekształcane są one w kolejkę procesów gotowych do wykonywania (miejsce *ReadyQueue*) i kolejkę procesów wykonywanych (miejsce *ExeQueue*). Zadanie zablokowane na pewnym zasobie jest przenoszone do osobnej kolejki (miejsce *Res_QUEUES*) i tam oczekuje do momentu zwolnienia zasobu (wykonanie przejścia *Blked_res*n*_To_Ready*).



Rys. 2. Wzorzec modułu szeregującego
Fig. 2. Scheduler design pattern

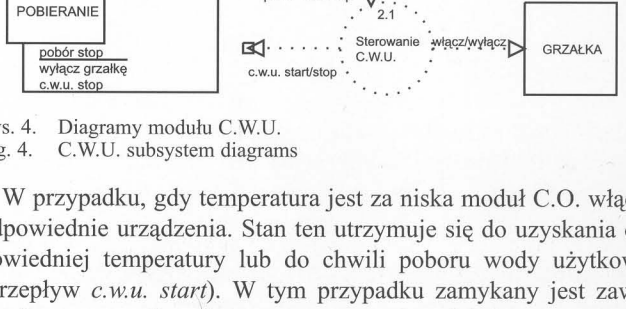
5. Projekt systemu - przykład

W przykładzie rozważony zostanie model opisujący działanie systemu sterującego kotła dostarczającego ciepłą wodę do centralnego ogrzewania (C.O.) oraz celów użytkowych (C.W.U) [7]. Pierwsza część przykładu ilustruje konwersję modelu strukturalnego do hierarchicznej kolorowanej sieci Petriego, druga zaś zastosowanie sieci HCPN do modelowania wytwarzanego systemu na poziomie zbioru zadań czasu rzeczywistego. Diagram ogólny modelu strukturalnego przedstawiono na rys. 3.



Rys. 3. Diagram ogólny systemu kotła dwufunkcyjnego
Fig. 3. Two-function heater main diagram

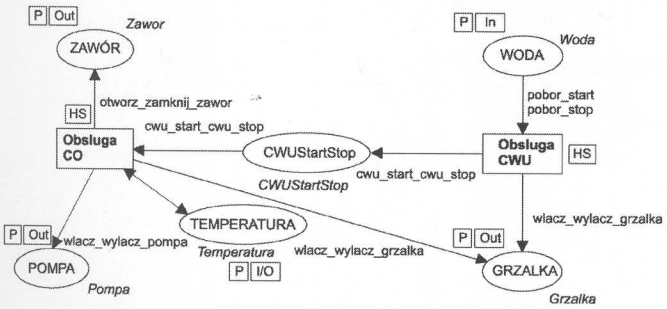
Po otrzymaniu sygnału z urządzenia poboru wody użytkowej (*WODA*), moduł C.W.U. uruchamia ogrzewacz wody (*GRZAŁKA*) oraz za pomocą przepływów sterujących c.w.u. start/stop informuje moduł C.O. o rozpoczętym lub zakończonym poborze wody (Rys. 4).



Rys. 4. Diagramy modułu C.W.U.
Fig. 4. C.W.U. subsystem diagrams

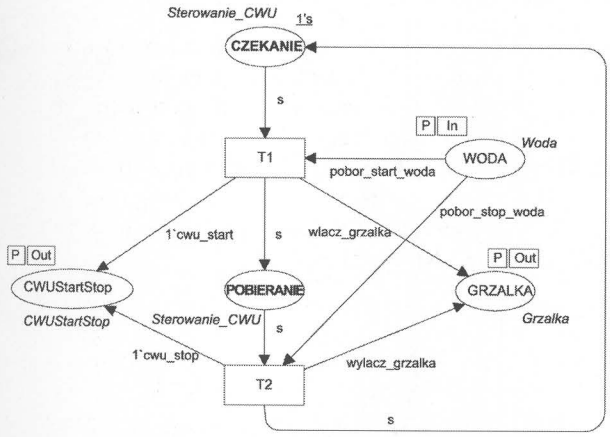
W przypadku, gdy temperatura jest za niska moduł C.O. włącza odpowiednie urządzenia. Stan ten utrzymuje się do uzyskania odpowiedniej temperatury lub do chwili poboru wody użytkowej (przepływ *c.w.u. start*). W tym przypadku zamykany jest zawór i wyłączana grzałka, ale praca pompy trwa nadal. System przyznaje pierwszeństwo modułowi poboru wody użytkowej C.W.U. w uzyskaniu dostępu do urządzeń (grzejnika i zaworu).

Model strukturalny kotła można poddać konwersji do sieci CPN. Na podstawie reguł podanych w pracy [3] otrzymuje się sieć HCPN, której dwie podstrony pokazano na rys. 5 i 6.



Rys. 5. Strona sieci HCPN powstała z diagramu na Rys. 3
Fig. 5. HCPN page created from the diagram of Fig. 3

Pierwsza z nich powstała na podstawie diagramu ogólnego z rys. 3, zaś druga odpowiada specyfikacji procesu *Sterowanie C.W.U.* Przepływy z modelu strukturalnego są reprezentowane w sieci za pomocą trójek: przejście-miejsce-przejście, zaś powiązania pomiędzy poszczególnymi stronami są realizowane za pomocą odpowiednich portów wejściowych i wyjściowych.



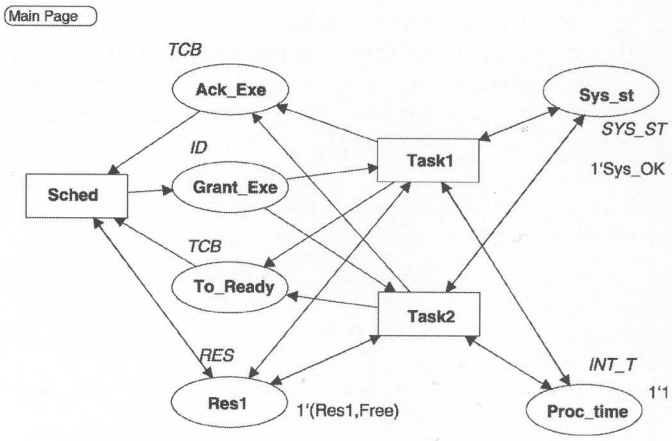
Rys. 6. Strona sieci HCPN modelująca moduł C.W.U.
Fig. 6. HCPN page with the C.W.U. subsystem model

Opisana wyżej struktura funkcyjna oprogramowania w naturalny sposób może zostać zdekomponowana na system składający się z dwu zadań czasu rzeczywistego, w którym pierwsze zadanie odpowiada za obsługę C.O. (sterowanie ciągłe) a drugie za obsługę C.W.U. (sterowanie dyskretne). Komunikacja pomiędzy zadaniami odbywała się będzie przez współdzielony rejestr (zasób Res1). Parametry zadań czasu rzeczywistego sprecyzowano w tabeli Tab.1

Tab. 1. Parametry zadań
Tab. 1. Tasks parameters

Task	Task Per.	Task WCET	Task Prior.	Resources
Task1	200	50	5	*Res1*: 10 20 30 40 50
Task2	10	5	3	*Res1*: 1 2 3 4 5

System uruchamiany będzie na jednym komputerze. Dla każdego z zadań określono cykl wzbudzenia, najdłuższy czas wykonywania, priorytet oraz przedziały procesu obliczeniowego, w którym wymagany jest dostęp do współdzielonego zasobu (pola zaciemnione). Metoda ostatecznego formułowania opisu systemu na poziomie zadań [6] zakłada konstruowanie pewnej ściśle określonej sieci nadrzędnej stanowiącej ogólny opis systemu. We wspomnianej sieci poszczególne przejścia reprezentują składniki systemu (moduły szeregujące i zadania), zaś miejsca oraz łuki określają kanały komunikacyjne pomiędzy składnikami (Rys. 7).



Rys. 7. Główna strona projektu na poziomie zadań
Fig. 7. Main project page (task set level)

Ostateczny, wykonywalny model systemu uzyskuje się przez powiązanie odpowiednich przejść z podsieciami (podstronami) definiującymi właściwości poszczególnych zadań i modułu szeregującego (por. punkty 3, 4). Opisana struktura sieci umożliwia również wykazanie szeregowalności zbioru zadań przy założonej wydajności systemu mikroprocesorowego [6].

6. Podsumowanie

W artykule pokazano, że sieci TCPN mogą być z powodzeniem stosowane we wspomaganiu wytwarzania oprogramowania systemów czasu rzeczywistego. Rozwijanie opisu oprogramowania przy pomocy metod inżynierskich [2] uzupełnione może zostać o opis w języku formalnym [3, 6] umożliwiając przewidzenie określonych właściwości algorytmicznych i czasowych systemu na drodze matematycznej. Prezentowany w artykule przykład systemu czasu rzeczywistego opisany początkowo w języku diagramów metodyki Warda-Mellora przekształcono do odpowiadających struktur sieci HTCPN uzyskując formalny opis oprogramowania zarówno z perspektywy funkcyjnej jak i implementacyjnej.

7. Literatura

[1] Jensen K.: Coloured Petri Nets, Basic Concepts, Analysis Methods and Practical Use, Vol. 1-3, Springer 1996.

[2] Ward P. T., Mellor S. J.: Structured Development for Real-Time Systems, Yourdon Press, Englewood Cliffs, Prentice Hall 1985.

[3] Trybus B.: Zastosowanie kolorowanych sieci Petriego do analizy strukturalnej systemów czasu rzeczywistego, rozprawa doktorska pod kierunkiem prof. T. Szmuca, AGH 2004.

[4] Elmstrom R., Lintulampi R, Pezze M.: Giving Semantics to SA/RT by Means of High-Level Timed Petri Nets, Journal on Real-Time Systems, May 1993, 249-272.

[5] Szmuc T.: Modele i metody inżynierii oprogramowania systemów czasu rzeczywistego, Wydawnictwa AGH, Kraków 2001.

[6] Samolej S.: Projektowanie systemów wbudowanych z zastosowaniem czasowych kolorowanych sieci Petriego, rozprawa doktorska pod kierunkiem prof. T. Szmuca, AGH 2003.

[7] Świder Z., Trybus L., Śnieżek M.: Two-channel Temperature Controller for Buildings, 3rd IEEE Conf. Control Appl., Glasgow, 1994, Vol. 1, 21-26.

Title: Using coloured Petri nets for real-time systems design

Artykuł recenzowany