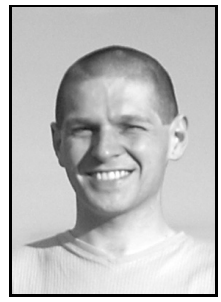
Michał GROBELNY
UNIWERSYTET ZIELONOGÓRSKI,
Licealna 9, 65-417 Zielona Góra

Odwzorowanie hierarchicznych interpretowanych sieci Petriego sterowania z makromiejscami w diagramach aktywności UML

Mgr inż. Michał GROBELNY



W roku 2007 ukończył studia na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego. Absolwent Zintegrowanych Studiów Zagranicznych Uniwersytetu Zielonogórskiego i Fachhochschule Giessen-Friedberg (Niemcy). Od października 2011 zatrudniony na stanowisku asystenta w Katedrze Mediów i Technologii Informacyjnych Uniwersytetu Zielonogórskiego. Zainteresowania naukowe obejmują metody specyfikacji osadzonych systemów sterowania. Członek Polskiego Towarzystwa Informatycznego.

e-mail: M.Grobelny@kmti.uz.zgora.pl

Streszczenie

Artykuł przedstawia metodę odwzorowania hierarchicznych interpretowanych sieci Petriego sterowania z makromiejscami w diagramach aktywności języka UML. Zgodnie z przyjętą zasadą odwzorowania akcji w tranzycjach sieci Petriego nie ma możliwości bezpośredniej graficznej reprezentacji miejsc interpretowanej sieci Petriego sterowania w diagramach UML. Jednocześnie konieczna jest zamiana takich elementów jak wyjścia przypisane do miejsc na aktywację i dezaktywację wyjść przy realizacji tranzycji oraz zamiany makromiejsc w makrotranzycje. Takie postępowanie komplikuje cały proces oraz może wprowadzać nieznaczne rozbieżności pomiędzy specyfikacją źródłową i docelową.

Słowa kluczowe: UML, diagramy aktywności, interpretowane sieci Petriego sterowania, specyfikacja behawioralna, specyfikacja hierarchiczna.

Mapping of hierarchical control interpreted Petri nets with macroplaces in UML activity diagrams

Abstract

The paper presents a method for mapping hierarchical control interpreted Petri nets into activity diagrams of UML. Usage of both specification techniques is possible considering international and multicultural design projects specifying hardware behavioural properties of a control process. Sometimes use of two different modelling techniques can be reasonable. After a short introduction (Section 1), a sample control process and its graphical interpretation using the control interpreted Petri net is described (Section 2). Fig. 1 shows the real model of the considered process of transportation of friable goods, whereas Fig. 2 presents graphical specification of the process with use of the control interpreted Petri net. Fig. 3 shows interpretation of action of UML activity diagrams in Petri nets. Due to no direct representation of the system state in UML activity diagrams, the outputs attached to places have to be exchanged with the outputs activated and deactivated with transitions firings. Sample output replacement scenario is depicted in Fig. 4 and is in details described in Section 3. Fig. 5 presents specification of the deliberated control process with usage of Mealy outputs (after replacement). On the other hand, conversion of macroplaces into macrotransitions is shown in Section 4 with graphical representation after exchange in Fig. 6. Section 5 describes transformation of the prepared Petri net into the activity diagram of UML with the process graphical representation in Fig. 7. Finally, Section 6 concludes the paper.

Keywords: UML, activity diagrams, control interpreted Petri nets, behavioural specification, hierarchical specification.

1. Wstęp

Faza specyfikacji behawioralnej sterownika logicznego [1] jest jednym z pierwszych etapów cyklu projektowego. Na tym etapie ustalany jest sposób pracy i zachowania projektowanego systemu. Specyfikacja może mieć postać nieformalną lub formalną.

Formalne metody [9] specyfikacji graficznej mogą zostać obejmują wiele technik, takich jak algorytmiczne maszyny stanów ASM, sieci SFC, sieci Petriego [3, 4] czy diagramy maszyny stanowej i diagramy aktywności UML [2, 4, 5, 6, 7, 8, 10]. Każda z powyższych technologii ma swoje zalety oraz rzeszę zwolenników. Niemniej jednak może tak się zdarzyć, że w jednym z projektów zostanie użyta więcej niż jedna metoda specyfikacji. Fakt taki może wynikać chociażby z udziału więcej niż jednego zespołu inżynierów rozproszonego po świecie i wykorzystującego na co dzień różne technologie. Pomocna w takiej sytuacji może być transformacja [4, 6, 7, 8] pomiędzy używanymi typami specyfikacji.

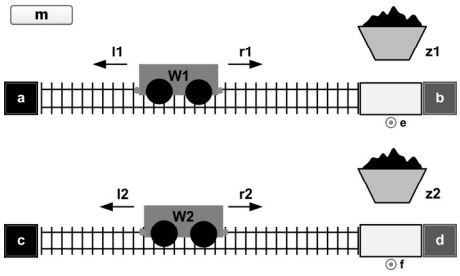
Artykuł skupia się na dwóch wcześniej wymienionych technikach, czyli na interpretowanych sieciach Petriego sterowania oraz diagramach aktywności języka UML (w wersji 2.x [10]) dostosowanych do opisu procesów sterowania. Coraz większa złożoność projektów sprawia, że są one oparte na metodach hierarchicznego przedstawiania zachowania systemu sterowania.

Artykuł podzielony jest następująco. Rozdział 2 przedstawia proces sterowania transportem materiałów sypkich, na którego przykładzie zobrazowane są kroki związane z odwzorowaniem sieci Petriego w diagramach aktywności UML. Dodatkowo, przedstawiona została graficzna specyfikacja procesu z wykorzystaniem interpretowanych sieci Petriego sterowania wraz z zaproponowanym uwzględnieniem makromiejsc. Rozdział 3 przedstawia konieczną zamianę wyjść typu Moore'a na wyjścia typu Meale'go w celu dostosowania sieci Petriego do specyfiki diagramów aktywności języka UML. Kolejnym niezbędnym krokiem do realizacji odwzorowania jest zamiana makromiejsc w makrotranzycje sieci Petriego. Proces zamiany omówiony został w rozdziale 4. Rozdział 5 przedstawia z kolei zasady transformacji pomiędzy dostosowanymi interpretowanymi sieciami Petriego sterowania a diagramami aktywności języka UML oraz rozpatruje zasadność przyjętych założeń. Rozdział 6 podsumowuje artykuł.

2. Proces sterowania i graficzna specyfikacja

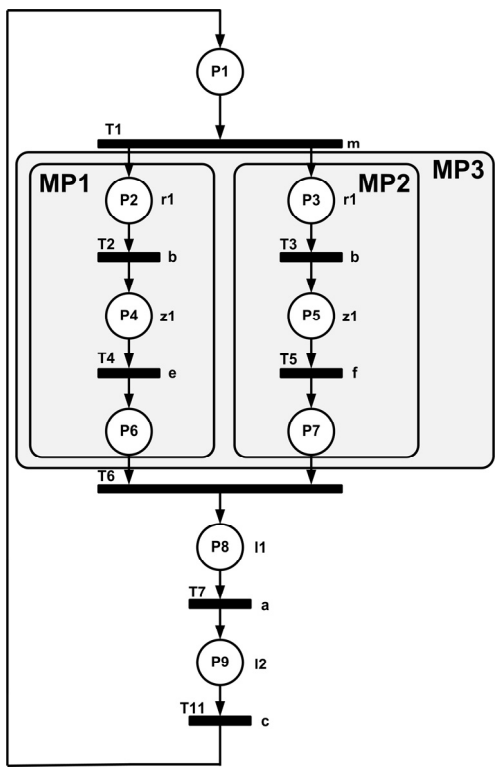
Odwzorowanie hierarchicznej interpretowanej sieci Petriego sterowania [3] z wykorzystaniem makromiejsc w diagramach aktywności języka UML [10] przedstawiono posługując się przykładem sterowania transportem materiałów sypkich z wykorzystaniem dwóch wózków. Rzeczywisty model opisywanego procesu sterowania przedstawiono na rysunku 1.

Rozważany przykład jest reprezentacją systemu do transportu materiałów sypkich. Proces rozpoczyna swoją pracę w przypadku, jeżeli oba wózki znajdują się w swoich miejscach startowych *a* i *c* oraz zostaje naciśnięty przycisk *m*. Pierwszą czynnością jest jednoczesny ruch obu wózków w prawo w kierunku miejsc załadunku oznaczonych odpowiednio *b* (wózek *W1*) i *d* (wózek *W2*). W momencie jak wózek osiągnie swój punkt docelowy następuje rozpoczęcie napełniania wózka. Procesy te wykonują się jednocześnie, ale są od siebie niezależne. Napełnianie realizowane jest przez ustawienie aktywnych sygnałow *z1* (wózek *W1*) i *z2* (wózek *W2*). Napełnienie wózków jest wskazywane prze sygnał *e* dla wózka *W1* i *f* dla wózka *W2*. Jeżeli oba wózki są pełne następuje powrót wózków – jeden za drugim. Pierwszy wraca wózek *W1* (wysterowany sygnał *l1*). Gdy wózek *W1* osiągnie swój punkt startowy *a* odbywa się powrót wózka *W2* (sygnał *l2* aż do momentu osiągnięcia punktu *c*).



Rys. 1. Rzeczywisty model omawianego procesu sterowania
Fig. 1. Real model of the described control process

Specyfikacja procesu sterowania z wykorzystaniem interpretowanej sieci Petriego przedstawiona została na rysunku 2. Do realizacji funkcji (ruch odpowienich wózków i napełnianie) wykorzystane zostały wyjścia typu Moore'a (sygnał r1 aktywny, gdy miejsce P2 jest aktywne).



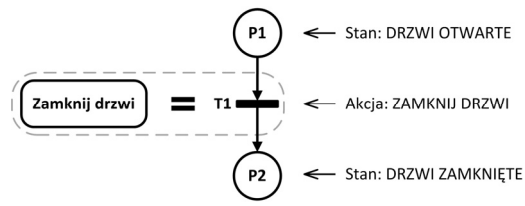
Rys. 2. Specyfikacja procesu sterowania z makromiejscami
Fig. 2. Specification of a control process with macroplaces

Na diagramie na rysunku 2 przedstawione zostały dodatkowo możliwe makromiejsca MP1, MP2 i MP3. Makromiejsca te są odpowiednikami poszczególnych etapów realizowanych przez proces sterowania. Makromiejsce MP1 odpowiada przemieszczaniu się wózka W1 oraz jego napełnianiu, miejsce MP2 odpowiada analogicznej sytuacji realizowanej równolegle przez wózek W2. Makromiejsce MP3 przedstawia oba etapy jako jeden. Zaprezentowane na rysunku 2 makromiejsca i ich struktura wynika z budowy sieci Petriego, ale jest także uzasadniona, gdy brana jest pod uwagę częściowa rekonfiguracja sterownika logicznego. Przy takim ułożeniu makromiejsc i ich odpowiednim dostosowaniu do rekonfiguracji w procesie syntezy i implementacji w sterowniku logicznym możliwa jest wymiana odpowieniej części programu dla procesu sterowania bez naruszania reszty procesu.

3. Zamiana wyjść typu Moore'a na wyjścia typu Mealy'ego

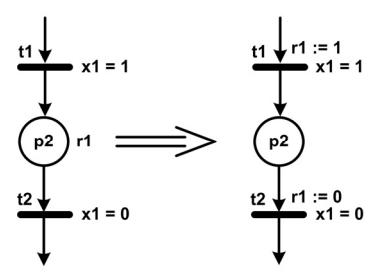
Diagramy aktywności języka UML są drugą możliwą formą specyfikacji behawioralnej omawianego procesu sterowania. Interpretowana sieć Petriego sterowania opisująca prezentowany przykład sterowania systemem do transportu materiałów sypkich zbudowana jest z wykorzystaniem wyjść typu Moore'a, których aktywność zależy od wewnętrznego stanu systemu.

Na potrzeby transformacji przyjęto zasadę odwzorowania akcji diagramów aktywności UML w tranzycjach sieci Petriego [6, 7, 8]. Interpretacja ta wynika z faktu, że akcja diagramu aktywności UML odpowiada za zmianę zachodzącą w specyfikowanym systemie. W sieciach Petriego za zmiany w systemie odpowiadają tranzycje. Miejsca przedstawiają lokalne stany systemu. Stąd przyjęto, że tranzycja i akcja są tożsame i tak powinno się je traktować. W tym przypadku każdą z akcji rozpatruje się bardziej jako wywołanie komendy *ZAMKNIJ DRZWI* niż określenie stanu *DRZWI SĄ ZAMYKANE*. Schematycznie przykład odwzorowania przedstawiony został na rysunku 3, gdzie miejsce *P1* odpowiada za stan *DRZWI OTWARTE*, miejsce *P2* przedstawia stan *DRZWI ZAMKNIĘTE* a tranzycja przedstawia akcję *ZAMKNIJ DRZWI* i w takiej sytuacji ta tranzycja odpowiada akcji diagramu aktywności *ZAMKNIJ DRZWI*. Jak zostanie to przedstawione w dalszej części artykułu interpretacja „akcja = tranzycja" przysparza wielu kłopotów przy transformacji i nie w każdym przypadku może zostać uznana za zasadną.



Rys. 3. Interpretacja akcji diagramu aktywności w sieciach Petriego
Fig. 3. Interpretation of action of UML activity diagrams in Petri nets

Typowo akcyjna interpretacja diagramów aktywności dodatkowo nie daje możliwości bezpośredniej graficznej reprezentacji stanów wewnętrznych sterownika logicznego. Fakt ten wymusza wykorzystanie wyjść typu Mealy'ego co wiąże się w omawianym przykładowym procesie z jego przekonstruowaniem. W przypadku omawianego odwzorowania wspomniana zamiana wyjść z typu Moore'a na wyjścia typu Mealy'ego realizowana jest na etapie wykorzystania specyfikacji zapisanej przy pomocy sieci Petriego. Przykładowa zmiana realizacji wyjść przedstawiona została na rysunku 4.

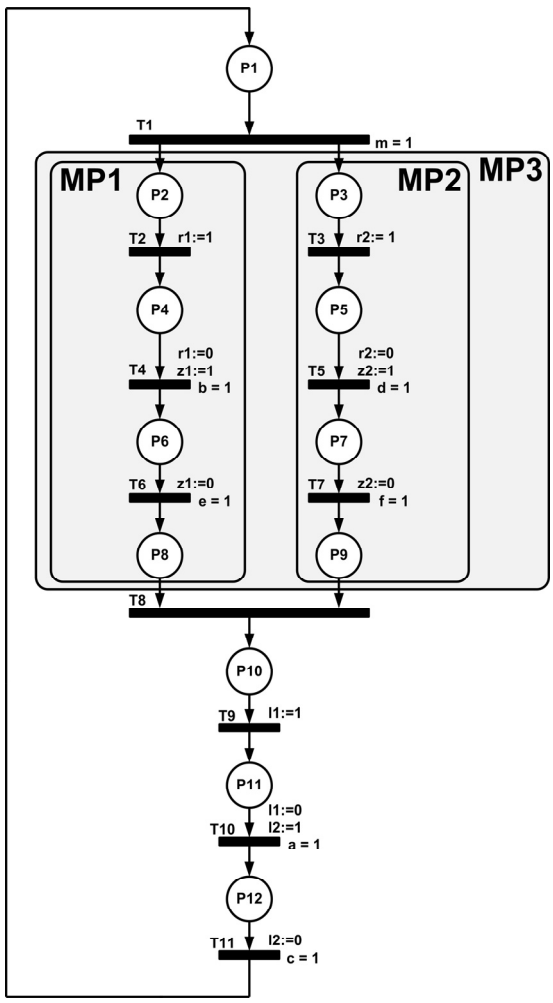


Rys. 4. Zamiana wyjść typu Moore'a na wyjścia typu Mealy'ego
Fig. 4. Exchange of Moore type outputs with Mealy type outputs

W tym przypadku w pierwotnym diagramie sygnał jest aktywny w sytuacji, jeżeli miejsce *p2* sieci Petriego jest aktywne. Jest to typowe wyjście zależne tylko i wyłącznie od stanu systemu, ponieważ zawsze jeżeli miejsce *p2* będzie aktywne (system będzie w stanie lokalnym *p2*) jednocześnie sygnał *r1* będzie aktywny. Zamiana wyjścia typu Moore'a na wyjście typu Mealy'ego polega na odłączeniu sygnału od konkretnego miejsca i przypisaniu jego aktywacji i dezaktywacji do otaczających dane miejsce tranzycji.

W omawianym przykładzie (przedstawionym na rysunku 4) w docelowej sieci Petriego aktywacja sygnału wyjściowego *r1* (przypisanie sygnałowi *r1* wartości *1*) odbywa się w chwili uruchomienia (odpalenia) tranzycji *t1*. Odpalenie tranzycji zależne jest od aktywności sygnału wejściowego *x1*. Dezaktywacja sygna-

łu wyjściowego *r1* na omawianym rysunku 4 realizowana jest w momencie uruchomienia (odpalenia) tranzycji *t2*, a to zależne jest od stanu sygnału wejściowego *x1* (sygnał *x1* musi być nieaktywny, aby tranzycja została zrealizowana).

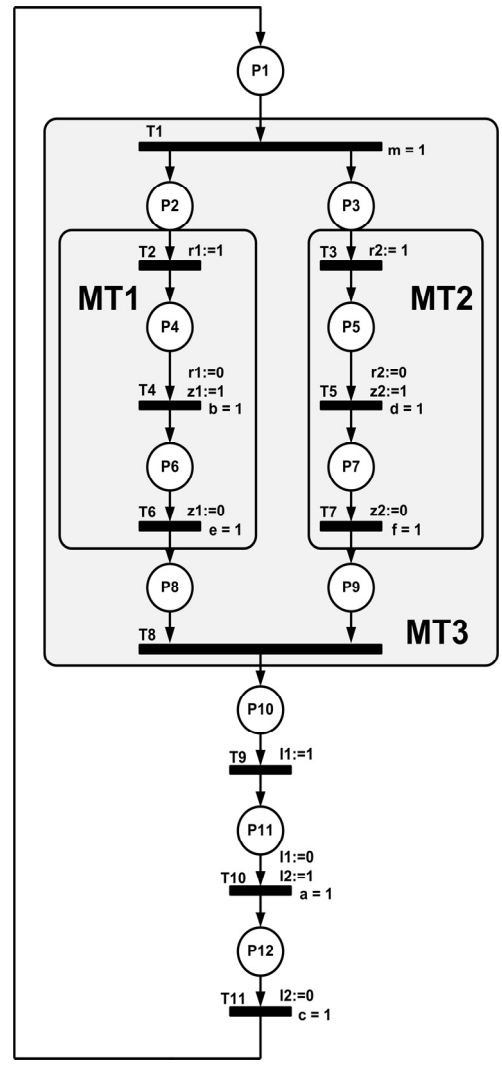


Rys. 5. Specyfikacja procesu sterowania z użyciem wyjść typu Mealy'ego
Fig. 5. Specification of a control process with use of Mealy outputs

Omawiany proces sterowania i odwzorowanie jego specyfikacji w diagramach aktywności także wymaga zamiany typu sygnałów wyjściowych. Sieć Petriego po realizacji takiej zamiany przedstawiona została na rysunku 5. Jak można zaobserwować, zamiana ta spowodowała zmianę liczby miejsc i tranzycji. Zwiększenie liczby miejsc i tranzycji w omawianym przykładzie nie byłoby konieczne, gdyby specyfikacja nie posiadała makromiejsc. Zwiększenie ich liczby odbyło się dodatkowo w celu zachowania czytelności sieci Petriego.

4. Zmiana makromiejsc w makrotranzycje

Kolejnym krokiem koniecznym do odwzorowania procesów jest przypisanie makromiejscom odpowiadającym im makrotranzycji, które docelowo mogą zostać przetransformowane na aktywności diagramu aktywności języka UML. Wynika to z przyjętej interpretacji akcji w postaci tranzycji i konsekwentnie aktywności w postaci makrotranzycji. Analogicznie w owej interpretacji miejsca nie posiadają bezpośredniego graficznego odwzorowania i w związku z tym makromiejsca też nie będą posiadały graficznego odpowiednika w diagramach aktywności. Stąd konieczna jest zmiana makromiejsc w makrotranzycje dla uzyskania spójności i poprawności omawianego procesu odwzorowania jednej techniki specyfikacji w drugiej.



Rys. 6. Specyfikacja procesu sterowania z wykorzystaniem makrotranzycji
Fig. 6. Specification of a control process with use of macrotransitions

Sieć Petriego z makromiejscami zamienionymi w makrotranzycje, opisująca omawiany w artykule proces sterowania transportem materiałów sypkich z wykorzystaniem dwóch wózków przedstawiona została na rysunku 6. Proces zamiany makromiejsc na makrotranzycje może przyjąć trzy możliwe scenariusze, omówione kolejno w dalszej części rozdziału.

Pierwszy scenariusz przewiduje utworzenie dodatkowych tranzycji bez funkcjonalności otaczających makromiejsc. Po połączeniu tych tranzycji z makromiejscem otrzymana zostaje makrotranzycja. W opisywanym przykładzie ten wariant nie został wykorzystany.

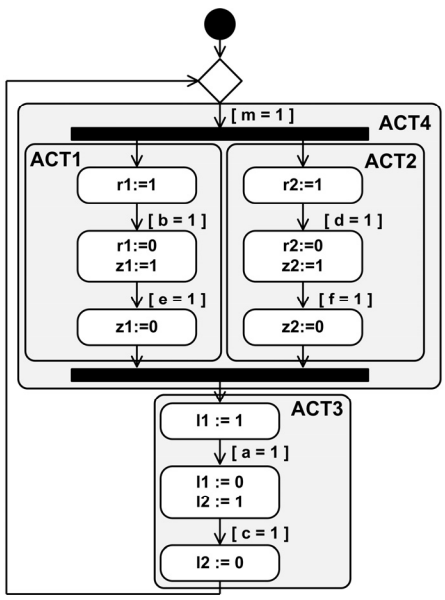
Drugą możliwością jest wcielenie tranzycji już istniejących w procesie sterowania otaczających makromiejsce. Jeżeli taka zamiana nie będzie miała wpływu na sam proces, jest ona dopuszczalna. Przykład takiej zamiany został zastosowany w opisywanym procesie, makromiejsce *MP3* (rysunek 5) po włączeniu tranzycji *T1* i *T8* zostało przekształcone na makrotranzycję *MT3* (rysunek 6). Włączenie obu tranzycji nie zmienia sensu wykonywanych czynności, dlatego mogło zostać zastosowane.

Kolejną, trzecią możliwością, jest wyłączenie skrajnych miejsc makromiejsca i w ten sposób otrzymanie makrotranzycji. Tak samo jak w dwóch poprzednich przypadkach należy pamiętać, że takie wyłączenie miejsc nie może zmieniać charakteru rozpatrywanej hierarchicznej makrokonstrukcji w szczególności jeżeli chodzi o sposób zachowania i wykonywane przez nią czynności. W przypadku omawianego procesu sterowania ostatni scenariusz polegający na wyłączeniu „zbędnych" miejsc i w ten sposób otrzymaniu makrotranzycji zrealizowany został przy zamianie

makromiejsc *MP1* i *MP2* w makrotranzycje - odpowiednio *MT1* i *MT2*.

5. Transformacja do diagramu aktywności języka UML

Kolejnym etapem prezentowanej w artykule metody odwzorowania hierarchicznych interpretowanych sieci Petriego sterowania w diagramach aktywności języka UML jest transformacja zmodyfikowanej i dostosowanej sieci Petriego na diagram aktywności. W przypadku opisywanego procesu sterowania ostateczną zmodyfikowaną siecią Petriego będącą podstawą transformacji jest sieć zaprezentowana na rysunku 6. Diagram aktywności języka UML dla omawianego procesu sterowania otrzymany w wyniku transformacji przedstawiony został na rysunku 7. Odwzorowanie bazuje na zasadach transformacji prostych (bez hierarchii) sieci Petriego do diagramów aktywności UML opisanych w [6, 7, 8]. Transformacja hierarchiczna jest rozbudowaniem tych zasad o elementy związane z hierarchicznym opisem procesów sterowania.



Rys. 7. Diagram aktywności UML dla omawianego procesu sterowania
Fig. 7. UML activity diagram for the described control process

Podstawową zasadą transformacji diagramów aktywności języka UML do interpretowanych sieci Petriego sterowania jest odwzorowanie tranzycji sieci Petriego w akcjach diagramu aktywności UML. Stąd, bazując na interpretacji dotyczącej akcji i tranzycjach, przyjęto zasadę, że makrotranzycje zamieniane są na aktywności złożone diagramu aktywności. Miejsca sieci Petriego nie posiadają odpowiedniego elementu na który mogłyby zostać odwzorowane, więc są poniekąd pomijane. W związku z tym makromiejsca sieci Petriego także nie mogą zostać uwzględnione i zostają zastąpione makrotranzycjami w celu dalszego ich odwzorowania w aktywnościach. Postępowanie znacząco komplikuje proces transformacji i jednocześnie modyfikuje częściowo specyfikowany proces, co może być powodem rozbieżności interpretacji specyfikacji behawioralnej danego sterownika logicznego. Stąd należy rozważyć zasadność każdorazowej interpretacji akcji w postaci tranzycji. W przypadku interpretowanych sieci Petriego sterowania i ich charakterystyki wydaje się zasadnym odwzorowanie akcji w miejscach sieci Petriego. Jednocześnie w takiej sytuacji należałoby przyjąć, że miejsca interpretowanej sieci Petriego sterowania mają charakter bardziej akcyjny niż stanowy. Akcyjna charakterystyka miejsc jest szczególnie widoczna w sytuacji rozpatrywania specyfikacji z punktu widzenia klienta lub systemu sterowanego.

6. Wnioski

Specyfikacja behawioralna sterownika logicznego może zostać przygotowana z użyciem wielu technik. W artykule zaprezentowano możliwość wykorzystania sieci Petriego i diagramów aktywności.

Prezentowany sposób odwzorowania i założone zasady powodują znaczny wzrost złożoności całego procesu transformacji. Należałoby rozpatrzyć zasadność przyjętych założeń, że akcja musi zawsze być odwzorowana w tranzycji. Należy rozważyć, czy w przypadku procesów sterowania specyfikowanych z wykorzystaniem interpretowanych sieci Petriego sterowania nie bardziej zasadnym jest przypisanie akcji diagramu aktywności miejsca jako odpowiednika. Wynikać to może z fakty, że miejsce z aktywnym sygnałem wyjściowym jest odpowiednikiem stanu systemu sterowania ale jednocześnie odpowiada za czynność wykonywaną przez system sterowany. W takiej sytuacji odpowiednie traktowanie akcji, a co za tym idzie aktywności diagramów aktywności języka UML zależne jest od typu rozpatrywanego systemu i znaczenia jego specyfikacji w postaci sieci Petriego.

Priorytetu VIII „Regionalne Kadry Gospodarki" Programu Operacyjnego Kapitał Ludzki współfinansowanego ze środków Europejskiego Funduszu Społecznego Unii Europejskiej i z budżetu państwa.

7. Literatura

[1] Adamski M., Barkalov A., Węgrzyn M. (Eds.): Design of Digital Systems and Devices, Lecture Notes in Electricel Engineering, Vol. 79, Springer-Verlag, Berlin Heidelberg 2011.
[2] Basile F., Chiachio P., Del Grosso D.: Modelling automation systems by UML and Petri Nets, Proceedings of the 9th International Workshop on Discreet Event Systems, Goteborg, 2008, str. 308 - 313.
[3] David R., Alla H.: Discrete, Continuous, and Hybrid Petri Nets, Springer Verlag, Berlin Heidelberg 2010.
[4] Eshuis R.: Semantics and Verification of UML Activity Diagrams for Workflow Modelling, rozprawa doktorska, University of Twente, Wierden 2002.
[5] Girault C., Valk R.: Petri Nets for Systems Engineering. A Guide to Modeling, Verification, and Applications, Springer Verlag, Berlin Heidelberg 2003.
[6] Grobelna I., Grobelny M., Adamski M.: Petri Nets and activity diagrams in logic controller specification - transformation and verification, Mixed Design of Integrated Circuits and Systems, Proceedings of the 17th International Conference, 2010, str. 607 – 612.
[7] Grobelny M., Grobelna I.: Diagramy aktywnosci jezyka UML i sieci Petriego w systemach sterowania binarnego od transformacji do weryfikacji, Pomiary Automatyka Kontrola, nr 10, 2010, str. 1154 – 1158.
[8] Grobelny M.: Transformation of UML 2.x activity diagrams into control interpreted Petri nets in hardware behavioural modelling, Prace Instytutu Elektrotechniki - Proceedings of Electrotechnical Institute, zeszyt 251, 2011, str. 87 – 95.
[9] Szmuc T., Szpyrka M.: Metody formalne w inżynierii oprogramowania systemów czasu rzeczywistego, Wydawnictwa Naukowo-Techniczne, Warszawa 2010.
[10] Unified Modeling Language Infrastructure and Superstructure complementary specifications, aktualna wersja normatywna 2.4.1, http://www.omg.org/spec/UML/2.4.1/

otrzymano / received: 15.03.2012
przyjęto do druku / accepted: 01.05.2012 *artykuł recenzowany / revised paper*