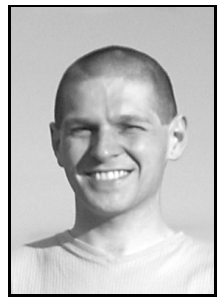
Michał Grobelny, Iwona GROBELNA
UNIWERSYTET ZIELONOGÓRSKI,
Podgórna 50, 65-246 Zielona Góra

Diagramy aktywności UML w projektowaniu rekonfigurowalnych sterowników logicznych

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



Mgr inż. Iwona GROBELNA

Absolwentka Wydziału Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego oraz Fachhochschule Giessen-Friedberg (Niemcy). Od marca 2008 roku zatrudniona na stanowisku asystenta w Instytucie Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Zainteresowania naukowe obejmują metody weryfikacji specyfikacji systemów osadzonych. Członek Polskiego Towarzystwa Informatycznego.

e-mail: I.Grobelna@iie.uz.zgora.pl

Streszczenie

Artykuł przedstawia sposób reprezentacji behawioralnej sterownika logicznego przy wykorzystaniu diagramów aktywności języka UML. Zaproponowane zostało zastosowanie diagramów aktywności do projektowania rekonfigurowalnych sterowników logicznych, a dokładnie do opisu zachowania sterownika logicznego podczas pracy. Do tego celu został dostosowany zbiór elementów diagramów aktywności w celu umożliwienia efektywnego modelowania behawioralnego. Rozważane jest także wykorzystanie hierarchicznych możliwości diagramów aktywności do częściowej rekonfiguracji układu.

Słowa kluczowe: specyfikacja behawioralna, rekonfigurowalny sterownik logiczny, UML, diagramy aktywności.

UML activity diagrams in design of reconfigurable logic controllers

Abstract

The paper focuses on behavioural representation of a logic controller with usage of UML activity diagrams. There is shown a subset of UML activity diagram elements sufficient to present logic controller behaviour simultaneously suitable for automatic synthesis with use of hardware description languages. After short introduction (Section 1) to the topic, UML activity diagrams as a specification technique are presented (Section 2). Additionally, there is described a subset of elements (Tab.1) of the discussed specification techniques fulfilling behavioural modelling requirements of a reconfigurable logic controller. Specification possibilities are given using sample control process of preparing the exact amount of liquid in two tanks (Section 3). The real model of the process is shown in Fig. 1. One of the possible behavioural specifications with use of UML activity diagrams is depicted in Fig. 2. This is a representation of the considered action state concept specification techniques in version 1.x. The other possibility is to specify a process with use of elementary system actions (Fig. 3), which is characteristic of the UML activity diagrams version 2.x. Fig. 4, on the other hand, shows signal based specification which is suitable for automatic hardware description language code generation (e.g. VHDL). Furthermore, Section 4 describes possibilities of using hierarchical aspects of activity diagrams to prepare specification for partial reconfiguration. Finally, Section 5 concludes the paper.

Keywords: behavioural specification, reconfigurable logic controller, UML, activity diagrams.

1. Wstęp

Jednym z podstawowych i jednocześnie kluczowym elementem projektowania systemów sterowania binarnego [1], w tym rekonfigurowalnych sterowników logicznych, jest etap specyfikacji behawioralnej. Projektanci podczas tej fazy muszą przewidzieć wszystkie funkcje niezbędne do prawidłowego działania docelowego urządzenia. Rozważając sterowniki bezpieczne ten etap jeszcze bardziej zyskuje na znaczeniu.

Diagramy UML [2, 3] są powszechnie stosowaną metodą modelowania w wielu dziedzinach. Ich źródło jest w modelowaniu i inżynierii oprogramowania, niemniej jednak obecnie znajdują zastosowanie w wielu dziedzinach: od modelowania biznesowego po projektowanie sprzętu [3, 4]. Prostota notacji języka UML upraszcza przepływ informacji pomiędzy członkami zespołu, jest także pomocna w trakcie ustaleń i negocjacji z klientem. Dodatkowym atutem jest tempo rozwoju tej technologii, dzięki czemu powstaje coraz więcej narzędzi usprawniających projektowanie.

2. Diagram aktywności UML

Diagramy aktywności języka UML [2], nazywane także *diagramami czynności*, są przedstawicielem grupy diagramów dynamiki języka UML. Pierwotnie tworzone były do wykorzystania w inżynierii oprogramowania, tak jak ma to miejsce w przypadku całej specyfikacja UML. Jednakże, z biegiem czasu ich pozycja ugruntowała się także w innych dziedzinach nauki, przemysłu i biznesu. Upowszechnienie się języka UML skutkuje także wykorzystaniem tej technologii przy projektowaniu systemów osadzonych.

Diagramy aktywności przedstawiają przepływ procesu sterowania pomiędzy akcjami odpowiadającymi za wykonanie poszczególnych funkcji systemu. Ich charakterystyka sprawia, że są metodą, która może zostać z powodzeniem wykorzystana przy specyfikacji behawioralnej procesów sterowania. Zastosowanie transformacji [5] do innych technik specyfikacji sterowników logicznych (np. sieci Petriego) może być dodatkowym mostem łączącym prezentowaną formę specyfikacji z innymi ugruntowanymi technikami.

Notacja UML w wersji 2.0 [3] (dalej oznaczana jako wersje 2.x – aktualna wersja to wersja 2.4.1 [2]) wprowadziła nową interpretację diagramów aktywności UML. Od tego czasu ta forma specyfikacji czynności przybrała postać wzorowaną na sieciach Petriego. Skupia się ona bardziej na akcjach systemu i przepływach pomiędzy nimi niż na stanach.

Podstawowe, wybrane elementy diagramów aktywności języka UML, niezbędne ale jednocześnie wystarczające do specyfikacji sterowników logicznych przedstawione zostały w Tab. 1. Można zauważyć, że proces sterowania może zostać wyspecyfikowany z uwzględnieniem takich elementów jak decyzje (węzeł decyzji i złączenia), przebiegi współbieżne (węzły rozdzielenia i złączenia) wraz z uwzględnieniem synchronizacji procesów skończywszy na strukturach hierarchicznych mających na celu ustrukturyzowanie i uporządkowanie złożonych specyfikacji dużych systemów z wieloma funkcjami. Każdy diagram aktywności rozpoczyna się węzłem początkowym. Zazwyczaj diagram aktywności posiada także zakończenie. Przy projektowaniu sterowników logicznych zakłada się, że sterowanie wykonywane jest w sposób ciągły z cyklicznym powtarzaniem całego procesu. Stąd często spotykane jest zapętlenie i możliwy brak elementu kończącego.

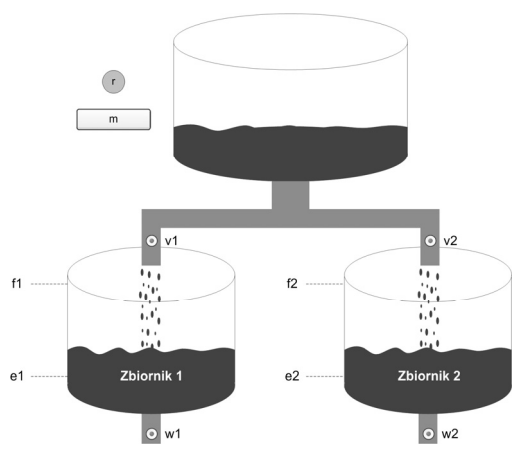
Sposób reprezentacji zależy od preferencji projektanta i umownych zasad panujących w grupie projektowej.

Tab. 1. Elementy graficzne diagramu aktywności UML i ich interpretacja
Tab. 1. Graphic elements in the UML activity diagram and their interpretation

Element	Interpretacja
Początek	Rozpoczęcie procesu sterowania
Koniec	Zakończenie procesu sterowania
akcja Akcja	Elementarna akcja o nieskończenie krótkim czasie realizacji
Aktywność	Złożona aktywność zawierająca w sobie opis podprocesu służąca do budowania struktur hierarchicznych
Przepływ	Element pokazujący przebieg przepływu sterowania
Rozdzielenie (fork)	Rozpoczęcie procesu współbieżnego
Złączenie (join)	Zakończenie procesu współbieżnego z synchronizacją wątków
Decyzja	Węzeł decyzyjny, do wyboru alternatywnych ścieżek sterowania
Złączenie (merge)	Złączenie przepływów rozdzielonych węzłem decyzyjnym lub złączenie procesów współbieżnych bez synchronizacji

3. Przykładowa specyfikacja behawioralna procesu sterowania

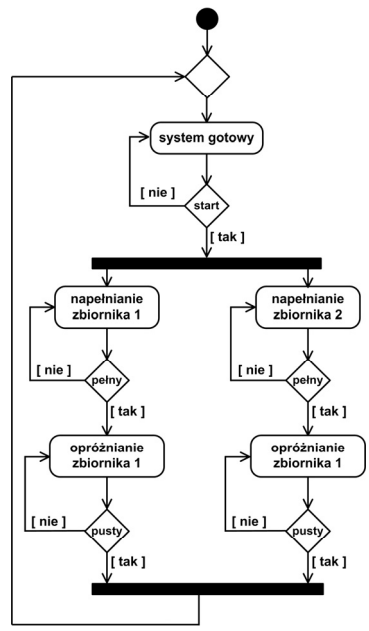
Zastosowanie diagramów aktywności języka UML w opisie zachowania procesów sterowania przedstawione jest na przykładzie nieskomplikowanego procesu odpowiedzialnego za przygotowanie odmierzonej ilości cieczy w dwóch zbiornikach. Model rzeczywisty omawianego procesu przedstawiony został na rys. 1.



Rys. 1. Model rzeczywisty opisywanego procesu sterowania
Fig. 1. The real model of the described control process

Proces rozpoczyna się od sygnalizacji gotowości systemu poprzez zapalenie diody gotowości (sygnał wyjściowy *r*). Rozpoczęcie pracy systemu inicjowane jest przez operatora poprzez naciśnięcie przycisku (sygnał wejściowy *m*). Po naciśnięciu przycisku następuje rozpoczęcie współbieżnego napełniania zbiorników – otwarcie zaworów napełniających (sygnały wyjściowe *v1* i *v2*). Napełnianie zbiorników odbywa się do momentu osiągnięcia przez ciecz pożądanego poziomu sygnalizowanego przez czujniki (sygnały wejściowe *f1* i *f2*). W tym momencie zawory napełniające są zamykane i otwierają się zawory opróżniające zbiorniki (*w1* i *w2*). Wylewanie cieczy odbywa się do momentu sygnalizacji przez czujniki całkowitego opróżnienia (sygnały wejściowe *e1* i *e2*). Po tym system przechodzi ponownie do stanu gotowości do działania sygnalizowanego przez diodę.

Specyfikacja graficzna omawianego przykładu została zaprezentowana na rys. 2 do rys. 4. Każda z tych trzech specyfikacji jest poprawna, jednakże przedstawione zostały różne cechy i interpretacje wykorzystania diagramów aktywności UML w systemach sterowania binarnego. Zaprezentowano także pewne różnice pomiędzy tą techniką specyfikacji w wersjach 1.x i 2.x.



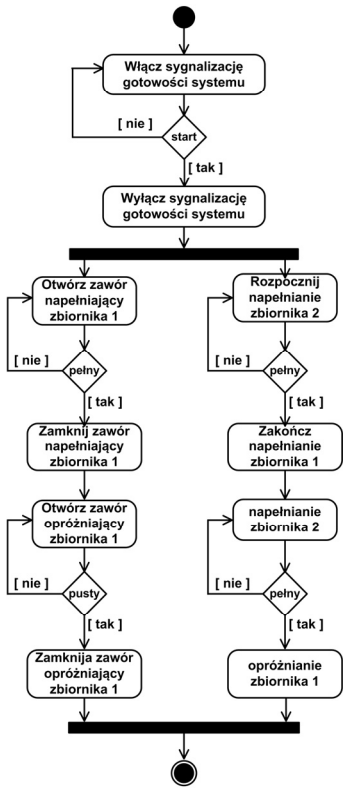
Rys. 2. Specyfikacja systemu z wykorzystanie diagramu aktywności UML
Fig. 2. System specification with usage of the UML activity diagram

Diagram przedstawiony na rys. 2 przedstawia proces z wykorzystaniem opisu słownego. Dodatkowo akcje mają stanowo akcyjny status charakterystyczny dla diagramów aktywności z serii 1.x, aczkolwiek czasami używany przez projektantów systemów z wykorzystaniem UML 2.x. Cechuje się to faktem, że akcja nie jest rozpatrywana jak krótkie wykonanie elementarnej operacji systemu, ale jest stanem systemu który ma za zadanie wykonanie określonej funkcji. Dodatkowo diagram z rys. 2 posiada pętle z węzłem decyzyjnym, które w ten sposób jawnie przedstawiają, że np. napełnianie zbiornika odbywa się do momentu gdy ów zbiornik jest pełny. W rzeczywistym systemie nie jest to operacja cykliczna, gdyż polega na otwarciu zaworu i oczekiwaniu na napełnienie, jednakże sposób z rys. 2 często jest preferowany przez projektantów. Dodatkowo omawiany diagram posiada zapętlenie całego proces przez co nie posiada elementu końcowego.

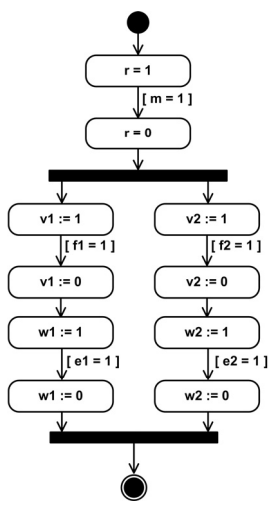
Kolejnym sposobem przedstawienia cech behawioralnych opisywanego sterownika logicznego jest diagram aktywności zobrazowany na rys. 3. W odróżnieniu od rys. 2 prezentuje on podejście typowo akcyjne. Cechą charakterystyczną jest tutaj fakt, że nie jest opisywana akcja *napełnianie zbiornika 1* tylko są dwie odrębne akcje z których pierwsza to *otwórz zawór napełniający zbiornik 1* a druga to *zamknij zawór napełniający zbiornik 1*. Jak widać jest to sprowadzenie specyfikacji do bardzie elementarnych funkcji systemu.

W diagramie na rys. 3, podobnie jak to jest przedstawione na rys. 2, akcje systemu opisane są w sposób słowny bez prezentacji sygnałów sterownika, które są w danym momencie wykorzystane. Taki sposób prezentacji jest jak najbardziej dopuszczalny i pewne jego aspekty, jak możliwość efektywnych konsultacji z osobą bez wiedzy na temat aspektów technicznych, może przemawiać na jej korzyść. Jednakże taki sposób prezentacji specyfikacji zachowania sterownika logicznego jest niewystarczający do poprawnego sporządzenia kodu w języku opisu sprzętu (np. VHDL) a tym

bardziej, gdy rozpatrywane jest automatyczne generowanie takiego kodu. W tym celu konieczne jest posłużenie się tabelą tłumacząca poszczególne komendy i warunki słowne na sygnały.



Rys. 3. Specyfikacja systemu z wykorzystaniem elementarnych akcji
Fig. 3. Specification of the system with elementary actions



Rys. 4. Specyfikacja systemu z wykorzystaniem rzeczywistych sygnałów
Fig. 4. System specification with usage of real signals

Odmienne podejście z wykorzystaniem rzeczywistych sygnałów systemu sterowania zaprezentowane zostało na rys. 4. Specyfikacja ta, także z wykorzystaniem diagramów aktywności UML, prezentuje system z prezentacją zmian poszczególnych docelowych sygnałów sterownika logicznego. Warunki także zostały zaprezentowane z użyciem sygnałów sterownika logicznego. Takie podejście umożliwia bezpośrednie przetłumaczenie specyfikacji na docelowy kod gotowy do syntezy i implementacji w rekonfigurowalnym sterowniku logicznym. Możliwe jest wygenerowanie kodu VHDL bezpośrednio z prezentowanego diagramu, gdyż diagram przewiduje wszystkie operacje na docelowych sygnałach.

4. Hierarchia i jej wykorzystanie w rekonfiguracji

Wykorzystanie specyfikacji hierarchicznej z użyciem złożonych aktywności jest wysoce wskazane przy projektowaniu rekonfigurowalnych sterowników logicznych, szczególnie w sytuacji jeżeli przewidziana jest częściowa rekonfiguracja układu. W opisywanym przykładowym sterowniku logicznym można wyróżnić alternatywnie trzy lub pięć złożonych czynności. W pierwszym przypadku byłyby to sygnalizacja gotowości i oczekiwanie na uruchomienie procesu jako pierwsza czynność, kolejne dwie odpowiedzialne byłyby za napełnianie i opróżnianie zbiorników (po jednej czynności przypadającej na zbiornik). W drugim przypadku, istnieje możliwość podzielenia każdego z dwóch współbieżnych procesów napełniania i opróżniana zbiorników na niezależne aktywności. W tym przypadku napełnianie zbiornika i jego opróżnianie byłyby osobnymi elementami na hierarchicznym diagramie. Takie przygotowanie specyfikacji i kodu do syntezy i implementacji pozwoli na późniejszą niezależną częściową rekonfigurację układu pozwalająca na wymianę tylko jednej (lub kilku) czynności (części układu).

5. Wnioski

W artykule przedstawiono jedną z wielu możliwych technik specyfikacji behawioralnej rekonfigurowalnych sterowników logicznych, a mianowicie wykorzystano do tego celu diagramy aktywności języka UML. Diagramy aktywności języka UML, jako element grupy diagramów dynamiki, odpowiedzialne są za opis zachowania projektowanego systemu. Autorzy przedstawiają kilka alternatywnych możliwości pokazując różne aspekty zastosowanej technologii. Odpowiedni dobór niezbędnych i jednocześnie wystarczających elementów spośród całego zbioru jaki udostępniają diagramy aktywności pozwala na skuteczne projektowanie zachowania systemów sterowania.

Dodatkowo zrealizowanie specyfikacji sterownika logicznego z wykorzystaniem docelowych sygnałów wejściowych i wyjściowych pozwala na wygenerowanie syntezowalnego kodu, który może zostać zaimplementowany w układzie FPGA. Hierarchiczna natura diagramów aktywności może się natomiast okazać pomocna, gdy rozpatrywana będzie częściowa rekonfiguracja sterownika logicznego.

Autorzy są stypendystami w ramach Poddziałania 8.2.2 „Regionalne Strategie Innowacji", Działania 8.2 „Transfer wiedzy", Priorytetu VIII „Regionalne Kadry Gospodarki" Programu Operacyjnego Kapitał Ludzki współfinansowanego ze środków Europejskiego Funduszu Społecznego Unii Europejskiej i z budżetu państwa.

6. Literatura

[1] Adamski M., Barkalov A., Węgrzyn M. (edytorzy): Design of Digital Systems and Devices, Lecture Notes in Electricel Engineering, Vol. 79, Springer-Verlag, Berlin Heidelberg 2011.
[2] Unified Modeling Language Infrastructure and Superstructure complementary specifications, aktualna wersja normatywna 2.4.1, http://www.omg.org/spec/UML/2.4.1/
[3] Schattkowsky T.: UML 2.0 – Overview and Perspectives in SoC Design, Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE'05).
[4] 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 - MIXDES 2010: proceedings of the 17th international conference, 2010, str. 607 612.
[5] Grobelny M.: Transformacja diagramów aktywności UML 2.0 do sieci Petriego w systemach sterowania binarnego, Pomiary, Automatyka, Kontrola, nr 7, 2009, str. 498-500.

otrzymano / received: 02.04.2012
przyjęto do druku / accepted: 01.06.2012 *artykuł recenzowany / revised paper*