ZESZYTY NAUKOWE UNIWERSYTETU SZCZECIŃSKIEGO

NR 576 STUDIA INFORMATICA NR 24 2009

STANISŁAW WRYCZA
BARTOSZ MARCINKOWSKI
Uniwersytet Gdański

SPECYFIKACJA WYMAGAŃ SYSTEMOWYCH W JĘZYKU SysML

Wprowadzenie

Modelowanie systemów informatycznych ma swoją znaczącą historię, w której istotne miejsce zajmują metodyki tworzenia systemów informatycznych: strukturalne, społeczne, obiektowe i adaptacyjne [23]. Aktualnie szczególną pozycję w tej dziedzinie osiągnął reprezentujący paradygmat obiektowy **język UML** (*Unified Modeling Language*), który stał się swoistym standardem w inżynierii oprogramowania [30]. Jego popularność, precyzja i elastyczność zadecydowały o zainteresowaniu się nim innych niż informatycy grup zawodowych, tworzących systemy w różnych **dziedzinach technologii**, **biznesu** i **administracji**. Są one zainteresowane modelowaniem i wdrażaniem szerokiego spektrum systemów w różnych dziedzinach, czyli inżynierią systemów. Procesami tymi zajmują się inżynierowie systemowi. Inżynier systemów jest ogniwem łączącym różne, często odległe dyscypliny w projekcie, ujmując, zarządzając i realizując projekt całościowo, **holistycznie** – z uwzględnieniem uwarunkowań sprzętowych, kadrowych, czasowych i innych. Grupa zawodowa inżynierów systemowych, skupiona w organizacji INCOSE (International Council on System Engineering), postanowiła zaadaptować język UML do swoich specyficznych potrzeb i uwarunkowań. W ten sposób rozpoczęły się prace nad nowym językiem – językiem SysML, które wspólnie koordynowały 3 organizacje:

- OMG (Object Management Group),
- INCOSE,
- ISO (International Organization for Standardization).

Owocem prac zespołów roboczych tych grup stał się adekwatny do potrzeb inżynierii systemów **język modelowania systemów SysML** (*Systems Modeling Language*), czyli język modelowania ogólnego przeznaczenia, służący do specyfikowania, analizy, projektowania i weryfikacji złożonych systemów. Formalnie został on opublikowany w 2006 roku, a od tego czasu został zaktualizowany do wersji 1.1, co nastąpiło pod koniec 2008 roku.

Ponieważ język SysML stał się w zaawansowanych technologicznie krajach rodzajem nieformalnego **standardu** w inżynierii systemów, autorzy postanowili go spopularyzować również w Polsce. Podstawowym materiałem źródłowych niniejszego artykułu jest specyfikacja języka SysML w postaci dokumentu OMG Systems Modeling Language (SysML), v1.1, z 1 listopada 2008 roku [14].

Celem artykułu jest prezentacja doświadczeń autorów w sferze analizy, projektowania i weryfikacji systemów na podstawie języka SysML, w aspekcie diagramów wymagań systemowych. Po wprowadzeniu w punkcie 1 dokonano ogólnej charakterystyki dziedziny inżynierii systemów. W kolejnym punkcie przedstawiono istotę i strukturę języka SysML. Punkty 3 i 4 poświęcono kolejno charakterystyce wymagań systemowych oraz diagramom wymagań systemowych. Opracowanie kończy prezentacja praktycznego zastosowania diagramu wymagań języka SysML do identyfikacji wymagań systemowych dla banku internetowego.

1. Inżynieria systemów

Dziedzina inżynierii systemów pojawiła się w latach pięćdziesiątych XX wieku w związku z koniecznością sprostania złożoności realizowanych projektów militarnych i kosmicznych. Dziś znajduje zastosowanie jako główna praktyka w złożonych problemach, które są technologicznymi wyzwaniami w wielu innych branżach. S. Friedenthal, A. Moore i R. Steiner **definiują inżynierię systemów** jako **wielodyscyplinarne** podejście do przekształcania zestawu potrzeb i wymagań interesariuszy w zharmonizowane rozwiązania systemowe zaspokajające te wymagania [4].

Wspomniane problemy rozwiązywane są przez wielodziedzinowe zespoły, ukierunkowane na zróżnicowane perspektywy postrzegania systemu przez interesariuszy, co sprzyja osiągnięciu wspólnie uzgodnionego, spójnego, zharmonizowanego **rozwiązania**. Inżynieria systemów intensywnie wykorzystuje wielorakie przepływy pracy i narzędzia do rozwiązywania złożonych problemów, mając obszary wspólne z dziedziną zarządzania projektami.

Inżynieria systemów wyznacza **holistyczny**, **kompleksowy** sposób myślenia o rozwiązaniu, począwszy od koncepcji systemu, przez jego tworzenie, aż do zastosowania. Proces inżynierii systemów może przebiegać zgodnie z **modelem SIMILAR**:

- postawienie problemu (***S****tate the problem*),
- wyszczególnienie alternatyw (***I****nvestigate alternatives*),
- modelowanie systemu (***M****odel system*),
- integracja (***I****ntegrate*),
- uruchomienie systemu (***L****aunch the system*),
- oszacowanie osiągów (***A****ssess performance*),
- ponowna ocena (***R****e-evaluate*).

Zgodnie z zaleceniami profesjonalnej organizacji inżynierów systemów INCOSE [20], **20–30% całego budżetu** realizowanego projektu powinno być przeznaczonych na prace i czynności związane z inżynierią systemową. Organizacja ta rekomenduje SysML jako podstawowe narzędzie modelowania systemów w różnych obszarach projektów inżynierskich.

2. Struktura języka SysML

Opracowany w latach 90. XX wieku język UML (*Unified Modeling Language*) stał się rodzajem *lingua franca* dla modelowania obiektowych systemów informatycznych. Popularność tę zauważono w środowiskach profesjonalnych, tworzących innego rodzaju systemy. W ten sposób język UML stał się inspiracją do zaproponowania założeń języka inżynierii systemów SysML, użytecznego głównie w takich obszarach jak **inżynieria oprogramowania**, ale także **inżynieria procesów**, **chemiczna**, **mechaniczna**, **elektryczna**. **Język modelowania systemów SysML** jest językiem modelowania ogólnego przeznaczenia, służącym do specyfikowania, analizy, projektowania i weryfikacji złożonych systemów [4]; [14]. Systemy te mogą obejmować sprzęt, oprogramowanie, zasoby informacyjne i ludzkie, procedury oraz urządzenia. Jest to więc graficzny język modelowania, oparty na semantyce, umożliwiającej reprezentowanie wymagań, dynamiki, struktury oraz cech systemu.

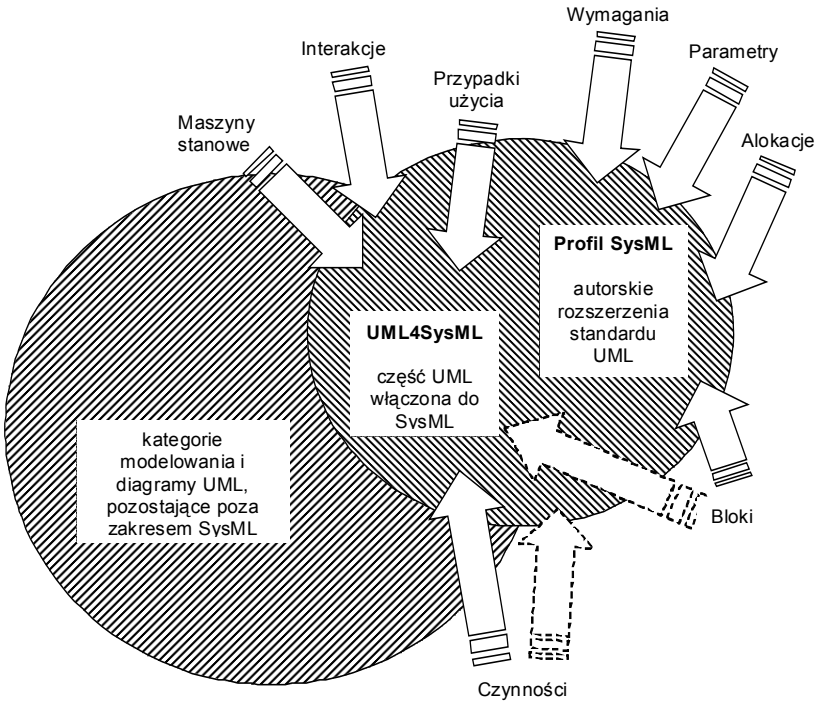
Oprócz zastosowań ściśle informatycznych, język ten jest użyteczny w wielu dziedzinach, wykorzystujących szeroko pojętą inżynierię systemów. Należą do nich **branże technologiczne**, między innymi kosmonautyka, automatyka, przemysł zbrojeniowy, telekomunikacja, biotechnologia, nanotechnologia, ener-

getyka, chemia. Naturalnie, tak jak UML, język inżynierii systemów SysML służy również rozwiązaniom informatycznym w **biznesie**, **administracji** i **służbie zdrowia**.

Ponieważ język inżynierii systemów SysML wywodzi się z języka modelowania systemów informatycznych UML, tym samym przejmuje ze swojego poprzednika te kategorie modelowania i diagramy, które najlepiej sprawdzają się w zastosowaniach technicznych. Tę grupę diagramów SysML często się określa mianem **UML4SysML**. Naturalnie, język ten wprowadza wiele oryginalnych koncepcji i rozszerzeń. W strukturze języka inżynierii systemów SysML można wyróżnić następujące diagramy:

- specyficzne dla języka SysML,
- zmodyfikowane diagramy UML 2.x,
- diagramy przeniesione bezpośrednio z języka UML 2.x.

Podział ten przedstawiono na rysunku 1.

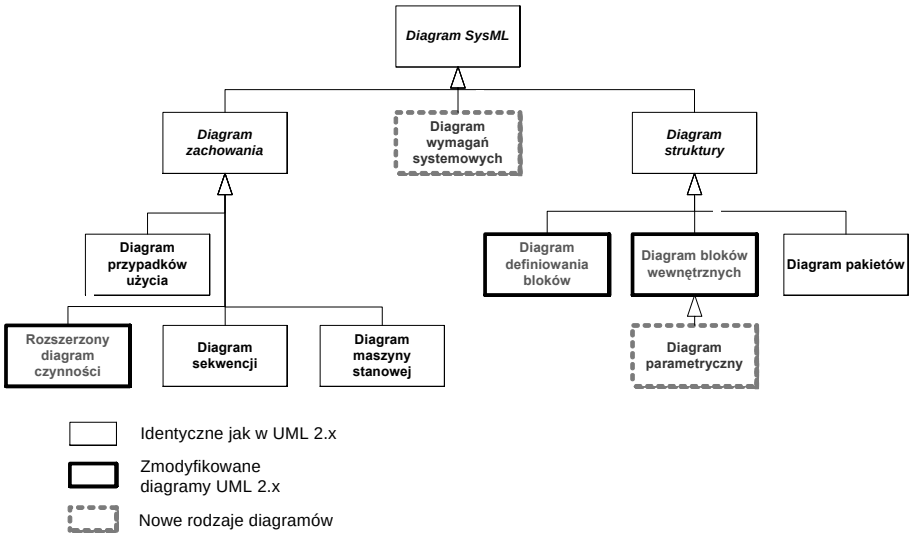


Rys. 1. Źródła i główne elementy języka SysML

Źródło: opracowanie własne na podstawie (Balmelli 2006).

Język SysML w wersji 1.1 zawiera **9 rodzajów diagramów**:
- diagram wymagań systemowych (*requirements diagram*),
- diagram przypadków użycia (*use case diagram*),
- rozszerzony diagram czynności (*activity diagram*),
- diagram sekwencji (*sequence diagram*),
- diagram maszyny stanowej (*state machine diagram*),
- diagram definiowania bloków (*block definition diagram*),
- diagram bloków wewnętrznych (*internal block diagram*),
- diagram parametryczny (*parametric diagram*),
- diagram pakietów (*package diagram*).

Oprócz powszechnie przyjętego podziału na **diagramy** struktury i dynamiki systemu, język SysML traktuje diagramy wymagań systemowych jako osobną kategorię. Hierarchę diagramów języka SysML przedstawiono na rysunku 2.



Rys. 2. Hierarcha diagramów języka SysML

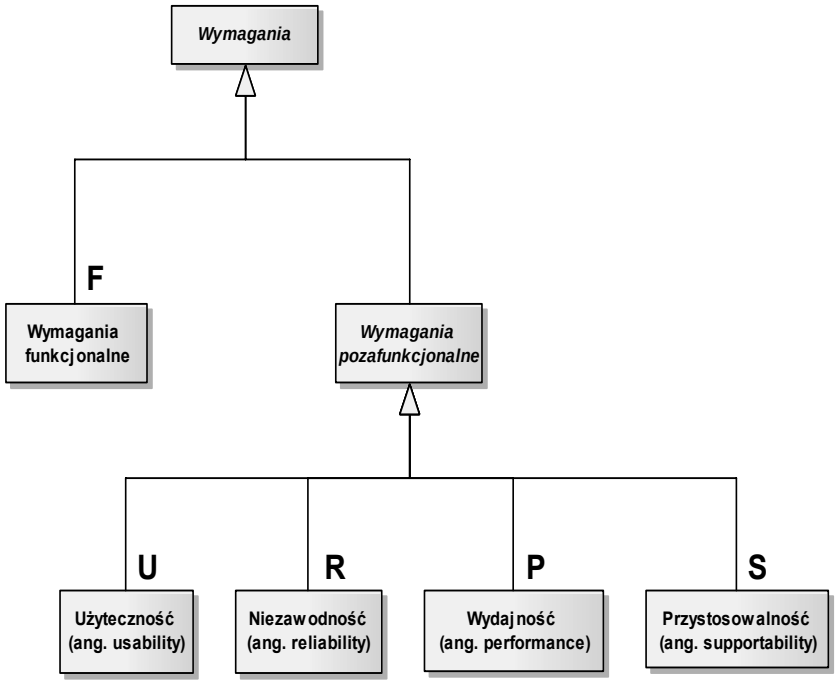
Źródło: [14].

3. Rodzaje wymagań systemowych

Jednym z najbardziej newralgicznych etapów procesu tworzenia systemu jest gromadzenie, specyfikacja, priorytetyzacja oraz akceptacja wymagań wo-

bec analizowanego, projektowanego bądź użytkowanego systemu. **Wymagania** są formalnie wyrażonymi potrzebami klienta – funkcjonalnościami lub cechami, którą system powinien spełniać [14]. Pozyskiwanie wymagań jest podstawą całego procesu budowy systemów, a od rezultatów tego etapu uzależniony jest dalszy sposób realizacji projektu. Dobrze określone wymagania zapewniają lepszą jakość przyszłego systemu i w konsekwencji wyższy poziom satysfakcji zamawiającego [22]. Wymagania mogą być uzyskane bezpośrednio od zleceniodawcy wykonania systemu w drodze wywiadu bądź analizy strategicznej dokumentacji firmy.

W literaturze jest wiele **klasyfikacji wymagań**. W branży informatycznej najbardziej rozpowszechniony jest model FURPS, opracowany w firmie Hewlett-Packard [5]. Zgodnie z modelem FURPS, na rysunku 3 przedstawiono wymagania systemowe podzielone na funkcjonalne i pozafunkcjonalne.



Rys. 3. Wymagania funkcjonalne i pozafunkcjonalne

Źródło: [5].

Wymagania funkcjonalne określają zachowanie systemu [10] – reprezentują usługi, które system musi oferować bez uwzględniania uwarunkowań technologicznych. Wymagania te są bezpośrednio przenoszone na kod źródłowy systemu w postaci konkretnych usług i funkcji. Z kolei **wymagania pozafunkcjonalne** odnoszą się do cech, parametrów systemu oraz jego otoczenia, wyrażonych w takich kategoriach, jak:

a) **użyteczność** (*usability*) – spełnienie tych wymagań skutkuje zwiększeniem stopnia przyswajalności obsługi systemu przez użytkowników dzięki estetycznemu i ergonomicznemu interfejsowi użytkownika, zapewniającemu intuicyjną nawigację w systemie;
b) **niezawodność** (*reliability*) – to własność systemu, określająca, czy pracuje on poprawnie; jej miernikami są między innymi średni czas międzyawaryjny, średni czas wdrożenia obejścia, średni czas naprawy, liczba błędów na tysiąc linii kodu;
c) **wydajność** (*performance*) – czyli wolumen pracy wykonanej przez system w określonym czasie i z zaangażowaniem określonych zasobów; miernikami wydajności są między innymi: czas odpowiedzi systemu, liczba transakcji w jednostce czasu, liczba jednocześnie obsługiwanych klientów zdalnych;
d) **przystosowalność** (*supportablity*) – czyli łatwość konfigurowania, aktualizowania, serwisowania systemu, rejestrowania zdarzeń systemowych w logach i przystosowywania oprogramowania do specyficznych potrzeb użytkownika przez help desk i personel wsparcia technicznego.

4. Diagramy wymagań systemowych

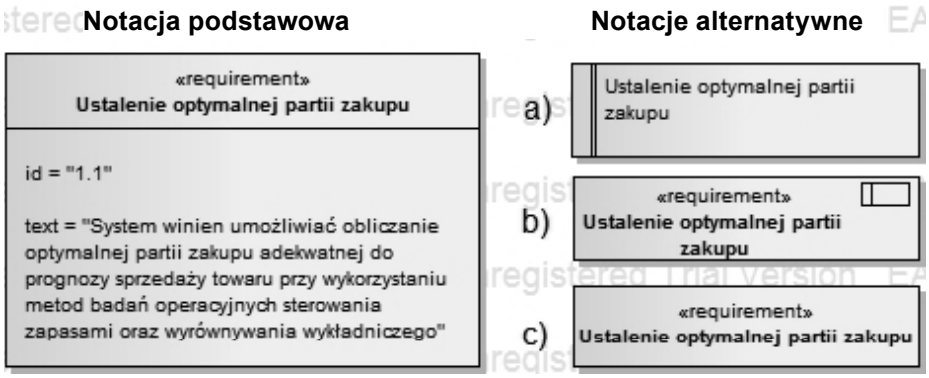
Diagram wymagań systemowych umożliwia ich **graficzne przedstawienie**, a także ich relacji z innymi kategoriami modelowania systemu. Specyfika wymagań jest oparta na następujących **podstawowych kategoriach modelowania** diagramów wymagań systemowych:

– wymaganie (*requirement*),
– związek (*relationship*),
– blok (*block*),
– przypadek użycia (*use case*),
– testowy przypadek użycia (*test case*),
– pakiet (*package*).

W języku SysML wymagania (*requirements*) symbolizują **kontrakt** między organizacją, klientem, zamawiającym system a jego wykonawcami. Podstawowymi charakterystykami każdego wymagania są:

- **numer porządkowy** (*id*),
- **treść wymagania** (*text*).

Na rysunku 4 przedstawiono podstawową i alternatywne **notacje graficzne** wymagań systemowych w diagramach wymagań języka SysML.



Rys. 4. Oznaczenia graficzne wymagań systemowych w języku SysML
Źródło: [5].

W praktycznych zastosowaniach wymagania systemowe mają charakter **hierarchiczny**. W związku z tym wskazane jest nadawanie im numeracji porządkowej w konwencji Dewey'a, określającej umiejscowienie danego wymagania w wielopoziomowej strukturze wymagań systemowych. Z kolei treść wymagania powinna zawierać spójny, syntetyczny opis właściwości, jakimi powinien się cechować system. Jednym z czynników utrzymania wysokiej jakości tworzonego diagramu wymagań systemowych jest stosowanie konsekwentnego, jednolitego, zrozumiałego stylu opisu i definiowania treści poszczególnych wymagań. W procesie tworzenia diagramu wymagań punktem wyjścia jest **identyfikacja** poszczególnych wymagań funkcjonalnych i pozafunkcjonalnych. Wymagania na diagramach wymagań systemowych języka SysML łączy się poprzez **związki**:

a) zagnieżdżenia (*containment*), umożliwiające tworzenie wielopoziomowej hierarchii wymagań oraz
b) szerokiego zakresu zależności (*dependencies*).

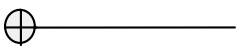
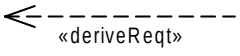
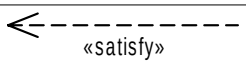
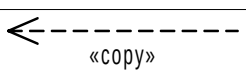
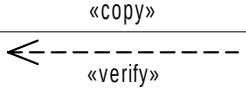
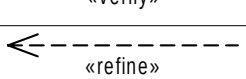
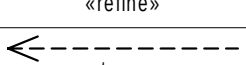
Wskazują one na charakter **logicznej zależności** między poszczególnymi wymaganiami.

Związkiem najpowszechniej stosowanym w diagramach wymagań systemowych jest **zagnieżdżenie**. Umożliwia ono połączenie wymagań nadrzędnych z podrzędnymi, przez co tworzona jest wielopoziomowa, **hierarchiczna struktura** wymagań systemowych. Wymaganie na danym poziomie hierarchii może mieć tylko jeden element nadrzędny (poza wymaganiem zamieszczonym na szczycie hierarchii). Wymaganie nadrzędne uznaje się za spełnione tylko i wyłącznie wtedy, gdy zostaną spełnione wszystkie jego wymagania podrzędne, czyli podwymagania.

Liczba **stereotypów**, które można przypisać **zależnościom** wiążącym dane wymaganie systemowe z inną kategorią modelowania, wynika z uniwersalności pojęcia wymagania. Specyfikacja języka SysML ujmuje siedem podstawowych odmian związków między wymaganiami. Związki te wyszczególniono w tabeli 1.

Tabela 1

Rodzaje związków na diagramach wymagań systemowych

Nazwa	Notacja
zagnieżdżenie	
zależność wyprowadzania	 «deriveReqt»
zależność realizacji	 «satisfy»
zależność powielania	 «copy»
zależność weryfikowania	 «verify»
zależność precyzowania	 «refine»
zależność śledzenia	 «trace»

Źródło: opracowanie własne.

Wszystkie odmiany zależności ujęte w specyfikacji języka SysML przedstawiają powiązanie pomiędzy parą wymagań: źródłowym i docelowym. Graficznie

te dwa wymagania łączy się linią przerywaną, wzbogaconą o **stereotyp tekstowy**, wskazujący na rodzaj zależności: «*deriveReqt*», «*satisfy*», «*copy*», «*verify*», «*refine*» lub «*trace*». Grot strzałki skierowany jest do wymagania źródłowego. Spotyka się także następujące określenia poszczególnych **stron zależności**:

– element źródłowy: dostawca, mistrz, oryginał,
– element docelowy: klient, niewolnik, kopia.

Bloki, przypadki użycia i testowe przypadki użycia są kategoriami modelowania istotnymi z punktu widzenia precyzji opisu kontekstu poszczególnych wymagań oraz nadzoru, monitorowania sposobu ich implementacji w systemie. Na diagramach wymagań systemowych stosuje się je w połączeniu z odpowiednimi rodzajami zależności.

Pakiety to mechanizm ogólnego zastosowania, służący do **organizowania** dokumentacji, w tym wymagań i innych kategorii modelowania diagramu wymagań systemowych. Tym samym pakiety i diagramy pakietów są użyteczne w zarządzaniu złożonością modelu wymagań systemowych.

5. Przykład diagramu wymagań systemowych SysML

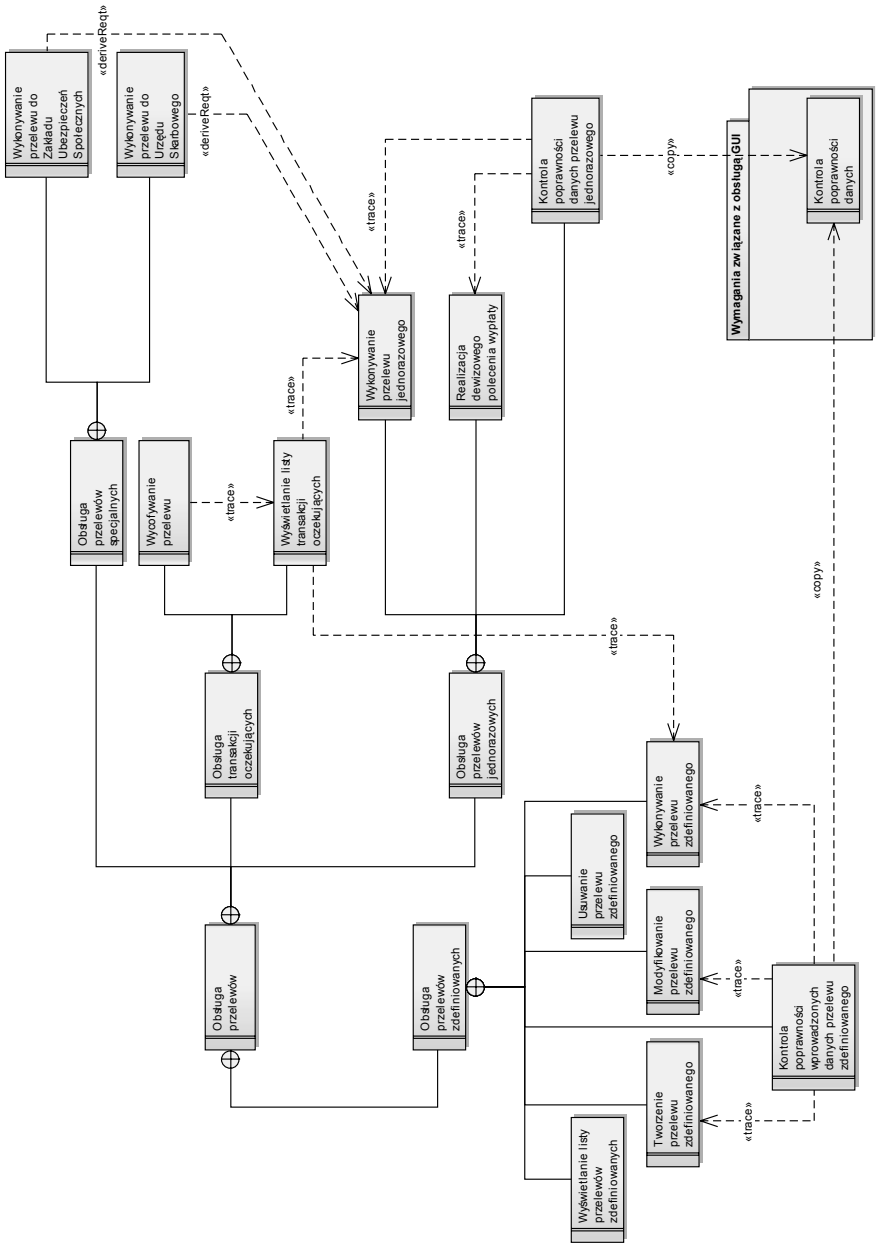
Praktyczne zastosowanie diagramu wymagań systemowych przedstawiono na rysunku 5. Przedmiotem wymagań jest obsługa przelewów w **banku internetowym**. Nadrzędnym wymaganiem w hierarchii jest właśnie wymaganie obsługa przelewów. Ma ono cztery bezpośrednie podwymagania, powiązane z wymaganiem nadrzędnym związkami **zagnieżdżenia**:

– obsługę przelewów zdefiniowanych,
– obsługę przelewów jednorazowych,
– obsługę transakcji oczekujących,
– obsługę przelewów specjalnych.

Z kolei podwymaganie obsługa przelewów zdefiniowanych dzieli się na następujące **wymagania szczegółowe**:

– wyświetlanie listy przelewów zdefiniowanych,
– usuwanie przelewu zdefiniowanego,
– wykonywanie przelewu zdefiniowanego,
– tworzenie przelewu zdefiniowanego,
– modyfikowanie przelewu zdefiniowanego.

Z tymi ostatnimi wymaganiami powiązana jest **zależność śledzenia** przez wymaganie Kontrola poprawności wprowadzonych danych przelewu zdefiniowane-



Rys. 5. Diagram wymagań systemowych banku internetowego

Źródło: [5].

go. Oznacza to, że wykonanie, tworzenie lub modyfikowanie przelewu wiąże się z wywołaniem funkcjonalności, kontrolującej poprawność danych wprowadzonych do poszczególnych formatek.

Na zasadzie związku zagnieżdżenia w wymaganiu podstawowym na rysunku 5 inne jego podwymagania są dalej **dekomponowane**, podobnie jak wyżej przedstawiona obsługa przelewów zdefiniowanych. Poza związkami zagnieżdżenia na rysunku 5 pokazano zastosowanie **zależności** <<trace>>, <<copy>> i <<deriveReqt>>.

Literatura

1. Bock C. *SysML and UML 2 Support for Activity Modeling*, „Systems Engineering" 2006, vol. 9.
2. *Ćwiczenia. UML 2.1*, red. S. Wrycza, Helion, Gliwice 2007.
3. Booch G., Rumbaugh J., Jacobson I., *The UML Reference Manual 2nd Edition*, Addison-Wesley, Boston 2004.
4. Friedenthal S., Moore A., Steiner R., *A Practical Guide to SysML*, OMG Press, Indianapolis 2008.
5. Grady R., Caswell D., *Software Metrics: Establishing a Company-wide Program*, Prentice Hall, New York 1987.
6. Hause M., *SysML hits the home straight*, http://www.esemagazine.com/index.php?option=com_content&task=view&id=140&Itemid=2; stan na dzień 5.08.2009.
7. http://www.magicdraw.com; stan na dzień 5.08.2009.
8. http://www.sysmlforum.com; stan na dzień 5.08.2009.
9. *Informatyka ekonomiczna*, red. S. Wrycza, PWE, Warszawa 2009.
10. Leffingwel D., Widrig D., *Zarządzanie wymaganiami*, WNT, Warszawa 2003.
11. Marcinkowski B., *Business modeling with UML and BPMN: Features and Comparison*, w: *Proceedings of BIR'2008, The Seventh International Conference on Perspectives in Business Informatics Research*, Wydawnictwo Uniwersytetu Gdańskiego, Gdańsk 2008.
12. Marcinkowski B., *Isomorphism of Interaction Diagrams in UML 2*, w: *Proceedings of 8th International Conference on Business Information Systems*, red. W. Abramowicz, Wydawnictwo Uniwersytetu Ekonomicznego, Poznań 2005.
13. Marcinkowski B., Wrycza S., Wyrzykowski K., *Interaction Occurrences and Combined Fragments in System Dynamics Modeling with UML 2 Sequence Diagram*, w: *Proceeding of the 14th International Conference on Information Systems Development. Advances in ISD: Bridging the Gap between Academia and Practice*, red.

A.G. Nilsson, R. Gustas, W. Wojtkowski, W.G. Wojtkowski, S. Wrycza, J. Zupančič, Karstad University Studies, Karlstad 2005.

14. *Object Management Group, OMG Systems Modeling Language (OMG SysML)*, Version 1.1, http://www.sysmlforum.com/docs/specs/OMGSysML-v1.1-08-11-01.pdf; stan na dzień 5.08.2009.
15. *Object Management Group, OMG Unified Modeling Language (OMG UML), Superstructure*, Version 2.2, http://www.omg.org/spec/UML/2.2/Superstructure/PDF/; stan na dzień 5.08.2009.
16. *Proceedings of BIR '2008. The Seventh International Conference on Perspectives in Business Informatics Research*, red. S. Wrycza, Wydawnictwo Uniwersytetu Gdańskiego, Gdańsk 2008.
17. *Proceedings of The Second AIS SIGSAND European Symposium on Systems Analysis and Design*, red. S. Wrycza, Wydawnictwo Uniwersytetu Gdańskiego, Gdańsk 2007.
18. Rumbaugh J., Blaha M.R., Lorensen W., Eddy F., Premerlani W., *Object-Oriented Modeling and Design*, Prentice Hall, New Jersey 1991.
19. *Systems Analysis and Design for Advanced Modeling Methods. Best Practices*, red. A. Bajaj, S. Wrycza, Information Science Reference, IGI Global, New York 2009.
20. *Technical Board International Council on Systems Engineering; Systems Engineering Handbook*, Version 2a, http://www.incose.org/ProductsPubs/products/sehandbook.aspx; stan na dzień 5.08.2009.
21. Weilkiens T., *Systems Engineering with SysML/UML. Modeling, Analysis, Design*, OMG Press, Indianapolis 2007.
22. Wojciechowski A., *Wprowadzenie do inżynierii wymagań*, www.inmost.org.pl/articles/Wprowadzenie_do_inżynierii_wymagań; stan na dzień 5.08.2009.
23. Wrycza S., *Analiza i projektowanie systemów informatycznych zarządzania*, Wydawnictwo Naukowe PWN, Warszawa 1999.
24. Wrycza S., Marcinkowski B., *A Light Version of UML 2: Survey And Outcomes redistribution permission*, w: *Proceedings of the 2007 Computer Science and IT Education Conference*, University of Technology Mauritius Press, Mauritius 2007.
25. Wrycza S., Marcinkowski B., *Język inżynierii systemów SysML. Architektura i zastosowanie*, Helion, Gliwice 2009.
26. Wrycza S., Marcinkowski B., *Język UML 2.x w dydaktyce akademickiej*, „Software Developer's Journal" 2008, nr 10.
27. Wrycza S., Marcinkowski B., *Kompleksowe podejście do nauczania języka UML 2.x na uczelniach wyższych*, w: *Bazy danych: rozwój metod i technologii. Architektura,*

metody formalne i zaawansowana analiza danych, red. S. Kozielski, B. Małysiak, P. Kasprowski, D. Mrozek, WKiŁ, Warszawa 2008.

28. Wrycza S., Marcinkowski B., *The UML2 Academic Teaching Challenge – An Integrated Approach*, in: *European and Mediterranean Conference on Information Systems*, Dubai 2008.
29. Wrycza S., Marcinkowski B., *Towards a Light Version of UML 2.x: Appraisal and Model*, „Organizacija" (Słowenia) 2007, nr 4.
30. Wrycza S., Marcinkowski B., Wyrzykowski K., *Język UML 2.0 w modelowaniu systemów informatycznych*, Helion, Gliwice 2005.
31. Wrycza S., Marcinkowski B., Wyrzykowski K., *Rola fragmentów wyodrębnionych w języku UML 2*, w: *Bazy danych: Modele. Technologie. Narzędzia. Architektura, metody formalne, bezpieczeństwo*, red. S. Kozielski, B. Małysiak, P. Kasprowski, D. Mrozek, Wydawnictwo Komunikacji i Łączności, Warszawa 2005.
32. Wrycza S., Marcinkowski B., Wyrzykowski K., *Rola i funkcje diagramów harmonogramowania w modelowaniu systemów informatycznych z wykorzystaniem języka UML 2*, w: *Infobazy '2005. Bazy danych dla nauki*, red. A. Nowakowski, Centrum Informatyczne TASK, Gdańsk 2005.
33. Wrycza S., Marcinkowski B., Wyrzykowski K., *UML. Tablice informatyczne*, Helion, Gliwice 2007.
34. Wrycza S., Rybiński W., *Information Systems Development Education: Assumptions and Practice*, w*: Proceedings of The Third International Conference on Systems Science: Addressing Global Issues*, red. F. Stowell, D. West, J.G. Howell, Plenum Press, New York 1993.

SYSTEMS REQUIREMENTS SPECIFICATION WITH SYSML

Summary

Information systems modeling has gone through meaningful evolution in which the significant place is occupied by information systems development methodologies: structured, soft, object oriented, and agile ones (Wrycza, 1999). At present the particular position in this field has been gained by UML (*Unified Modeling Language*), representing object-oriented paradigm. It became specific standard, a kind of lingua franca in software engineering (Wrycza, Marcinkowski i Wyrzykowski, 2005).

The professionals interested and involved in systems engineering are gathered in INCOSE organization. They have made up decision of Unified Modeling Language adoption to their specific requirements and work standards. In this way the efforts and

works for establishing new language – SysML, have been started. The joint project has been coordinated by three organizations:

- OMG (*Object Management Group*),
- INCOSE (*International Council on System Engineering*),
- ISO (*International Organization for Standardization*).

In result of the working groups efforts, the System Modeling Language SysML, adequate for system engineering needs was established. SysML is the modeling language of general use, serving specification, analysis, design and verification of the complex systems. Formally, its architecture was released in 2006 and then upgraded to version 1.1 in 2008.

The aim of this paper is the presentation of the authors experiences in systems analysis, design and verification by use of SysML requirements diagrams. After Introduction, chapter 2 follows, which includes the general characteristics of the system engineering field. In the next chapter, the structure and content of SysML was outlined. The chapters 4 and 5 are succeedingly dedicated to system requirements and requirements diagrams of SysML. The last chapter regards presentation of the practical application of SysML requirements diagrams in respect of systems requirements for Internet bank.

Translated by Stanisław Wrycza i Bartosz Marcinkowski