Profile UML 2.x w praktyce

Język inżynierii systemów

SysML

Architektura i zastosowania

Stanisław Wrycza
Bartosz Marcinkowski

Helion

Język inżynierii systemów SysML. Architektura i zastosowania. Profile UML 2.x w praktyce

Autorzy: Stanisław Wrycza, Bartosz Marcinkowski
ISBN: 978-83-246-2541-3
Format: 158×235, stron: 176

SysML, czyli System Modeling Language, to nowy obiektowy język modelowania systemów. W prostej linii wywodzi się on z języka UML, który stanowił do tej pory swego rodzaju standard w inżynierii oprogramowania. SysML został dostosowany do specyficznych potrzeb inżynierów systemowych, zajmujących się projektami w sposób całościowy. Pozwala na specyfikację, analizę, projektowanie i weryfikację złożonych systemów różnego rodzaju, a dzięki swoim dużym możliwościom i elastyczności w ciągu kilku lat zdołał zdobyć liczną rzeszę profesjonalnych użytkowników.

Opanowanie arkanów posługiwania się tym narzędziem ułatwi książka „Język inżynierii systemów SysML. Architektura i zastosowania. Profile UML 2.x w praktyce". Pierwsza na polskim rynku pozycja poświęcona SysML stanowi jednocześnie doskonałe wprowadzenie w zagadnienia inżynierii systemowej, zawiera szczegółowy opis architektury języka oraz prezentuje najważniejsze koncepcje związane z jego zastosowaniem. Książka niemal w całości przedstawia różnego typu diagramy, a zamieszczone w niej dodatki ułatwią zrozumienie nawet najbardziej skomplikowanych zagadnień i umożliwią sprawne poruszanie się po treści oraz uzupełnienie wiedzy w oparciu o publikacje innych autorów.

- Struktura, historia i zastosowania języka SysML
- Diagram wymagań systemowych
- Diagram definiowania bloków
- Diagram bloków wewnętrznych
- Diagram parametryczny
- Rozszerzony diagram czynności
- Diagramy UML4SysML

Poznaj język SysML, opierając się na wiedzy najlepszych specjalistów w tej dziedzinie!

Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 32 230 98 63
e-mail: helion@helion.pl

Spis treści

Rozdział 2.

Diagram wymagań systemowych

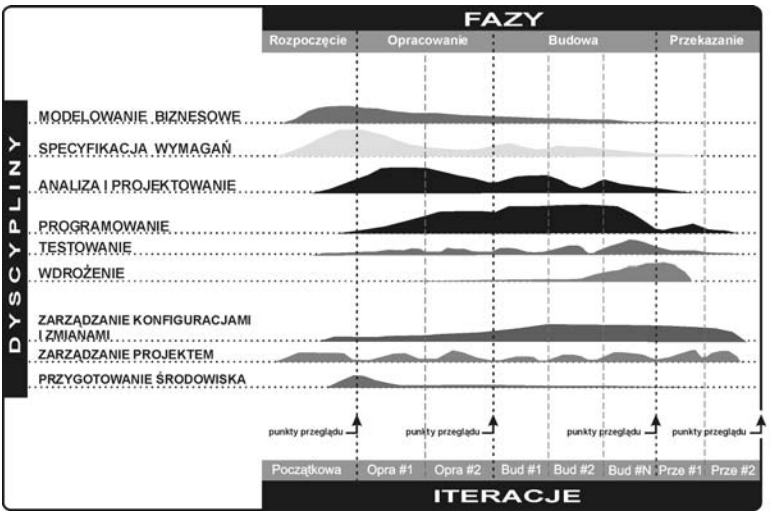
2.1. Znaczenie wymagań w procesie tworzenia systemu

Jednym z najbardziej newralgicznych etapów procesu tworzenia systemu jest gromadzenie, specyfikacja, priorytetyzacja oraz akceptacja wymagań wobec projektowanego bądź użytkowanego systemu. **Wymagania** są wyrażonymi z sposób formalny potrzebami klienta — funkcjonalnościami lub cechami, które system winien spełniać (OMG, 2008). Pozyskiwanie wymagań stanowi podstawę całego procesu budowy systemów, a od rezultatów tego etapu uzależniony jest dalszy sposób realizacji projektu; dobrze określone wymagania zapewniają lepszą jakość przyszłego oprogramowania i, w konsekwencji, wyższy poziom satysfakcji zamawiającego (Wojciechowski, 2009).

W inżynierii oprogramowania specyfikacja wymagań systemowych (ang. *requirements specification*) i ich dokumentowanie jest integralną częścią wszystkich cykli życia systemu informatycznego. Z podejściem obiektowym, językami UML i SysML ściśle związana jest metodyka **RUP** (ang. *Rational Unified Process*), którą syntetyzuje iteracyjno-przyrostowy cykl życia systemu, przedstawiony na rysunku 2.1. Jedną z dziewięciu dyscyplin cyklu życia RUP — i zarazem jedną z sześciu dyscyplin podstawowych — jest właśnie specyfikacja wymagań.

Poprawna specyfikacja wymagań jest **niezbędna** w dalszych fazach pracy nad systemem, wszelkie błędy zaś, popełnione na tym etapie, są trudne — a w konsekwencji kosztowne — do usunięcia w przyszłości. Wymagania mogą być uzyskane bezpośrednio od zleceniodawcy wykonania systemu w drodze wywiadu bądź analizy strategicznej dokumentacji firmy.

Rysunek 2.1.
Rational Unified Process

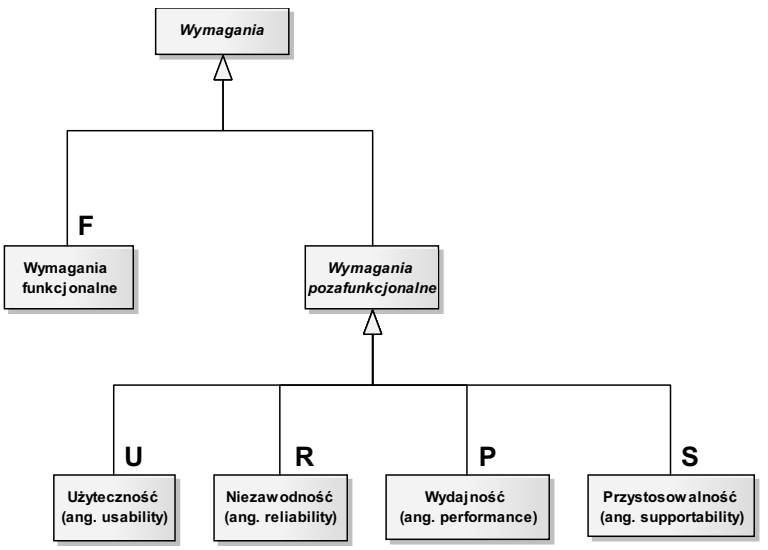


Źródło: opracowanie własne na podstawie (RUP, 2003)

2.1.1. Klasyfikacja wymagań

W literaturze funkcjonuje szereg **klasyfikacji wymagań**. W branży informatycznej najbardziej rozpowszechniony jest model FURPS, opracowany w firmie Hewlett-Packard (Grady i Caswell, 1987). Klasyfikację wymagań systemowych zgodnie z modelem FURPS przedstawiono na rysunku 2.2.

Rysunek 2.2.
Wymagania funkcjonalne i pozafunkcjonalne



Źródło: opracowanie własne na podstawie (Grady i Caswell, 1987)

Wymagania funkcjonalne określają zachowanie systemu (Leffingwel i Widrig, 2003) — 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:

- **użyteczność** (ang. *usability*) — spełnienie tych wymagań skutkuje zwiększeniem stopnia przyswajalności obsługi systemu przez użytkowników dzięki estetycznemu i ergonomicznemu interfejsowi użytkownika, zapewniającemu intuicyjną nawigację w systemie;
- **niezawodność** (ang. *reliability*) — czyli własność systemu, określająca, czy pracuje on poprawnie; jej miernikami są między innymi: średni czas międzyawaryjny, średni czas wdrożenia obejścia (ang. *bypass*), średni czas naprawy, liczba błędów na tysiąc linii kodu;
- **wydajność** (ang. *performance*) — wolumen pracy wykonanej przez system w określonym czasie i przy zaangażowaniu określonych zasobów; miernikami wydajności mogą być między innymi: czas odpowiedzi systemu, liczba transakcji w jednostce czasu, liczba jednocześnie obsługiwanych klientów zdalnych;
- **przystosowalność** (ang. *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.

2.1.2. Metody dokumentowania wymagań systemowych

W zależności od zastosowanego podejścia metodologicznego wykorzystuje się wiele sposobów specyfikowania wymagań. Mogą one być dokumentowane w formie tekstowej, graficznej, tabelarycznej lub hierarchicznej. Jednym z popularnych sposobów specyfikowania wymagań systemowych jest dokument **SRS** (ang. *System Requirements Specification*), opracowany przez organizację standaryzacyjną IEEE (IEEE, 1998). Struktura szablonu specyfikacji wymagań zgodnie z tym dokumentem składa się z trzech podstawowych sekcji:

- **wprowadzenia** — ujmującego takie kwestie, jak cel, zakres, definicje, akronimy i skróty, odwołania oraz ramy odpowiedzialności dostawcy;
- **opisu ogólnego** — poruszającego takie kwestie, jak perspektywy produktu, funkcje produktu, charakterystyki użytkownika, ograniczenia ogólne oraz założenia i zależności;
- **wymagań szczegółowych** — w tym wymagań wobec interfejsów zewnętrznych, wymagań funkcjonalnych, wymagań wydajnościowych, ograniczeń produktu, atrybutów oraz pozostałych wymagań.

Język SysML wprowadza nowy rodzaj diagramu, tj. **diagram wymagań systemowych**, dedykowany wyłącznie problematyce specyfikowania wymagań. W ten sposób uzyskuje się wizualny, czytelny w odbiorze sposób prezentacji wymagań systemowych

oraz jednoznaczne powiązanie zgromadzonych wymagań z realizującymi te wymagania elementami dokumentacji systemowej, przygotowanej z wykorzystaniem szerokiego spektrum elementów modelowania w języku SysML.

2.2. Elementy diagramu wymagań systemowych

2.2.1. Kategorie modelowania

Diagram wymagań systemowych umożliwia graficzne przedstawienie wymagań systemowych i ich związków z innymi kategoriami modelowania systemu. Wymagania specyfikuje się w odniesieniu do następujących **podstawowych kategorii modelowania** diagramów wymagań systemowych:

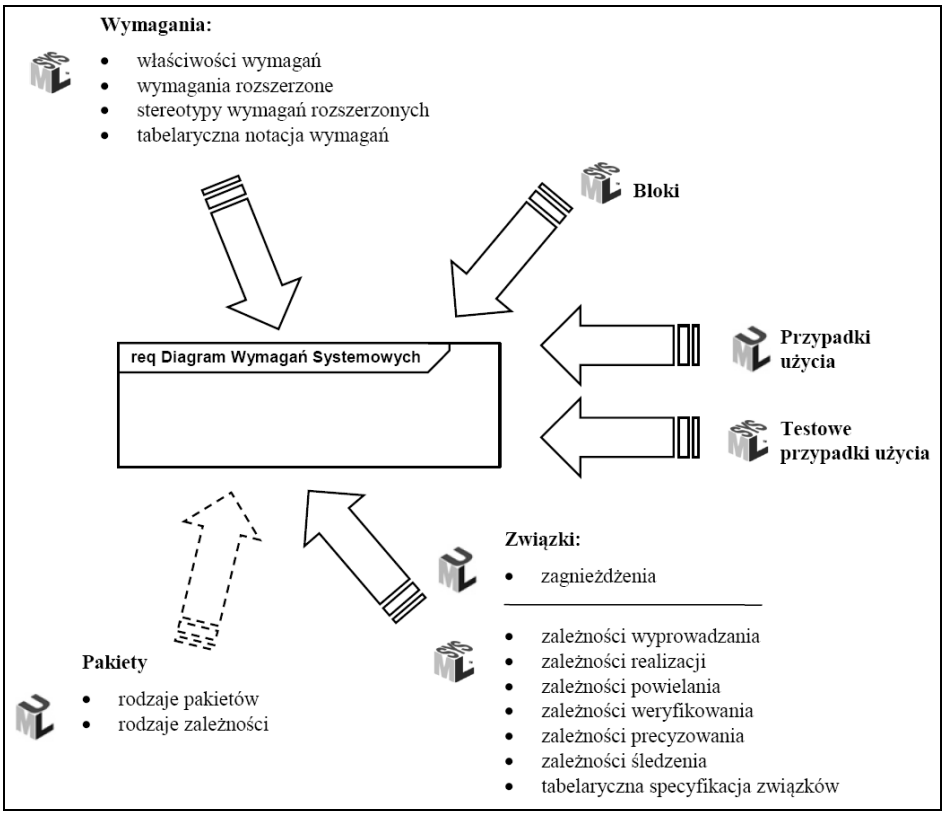
- wymaganie (ang. *requirement*),
- związek (ang. *relationship*),
- blok (ang. *block*),
- przypadek użycia (ang. *use case*),
- testowy przypadek użycia (ang. *test case*),
- pakiet (ang. *package*).

Wymienione kategorie oraz logiczne zależności między nimi przedstawiono na rysunku 2.3.

Punktem wyjścia w procesie tworzenia diagramu wymagań jest **identyfikacja** poszczególnych wymagań funkcjonalnych i pozafunkcjonalnych. Wymagania systemowe scharakteryzowano w punkcie 2.2.2. Z kolei zaawansowane aspekty wymagań ujęto w podrozdziale 2.3.

Wymagania wiąże się na diagramach wymagań systemowych języka SysML **związkami** zagnieżdżenia, umożliwiającymi tworzenie wielopoziomowej hierarchii wymagań, oraz szerokim zakresem zależności. Liczba stereotypów, które można przypisać zależnościom, wiążącym dane wymaganie systemowe z inną kategorią modelowania, wynika z wszechstronności i bogactwa interpretacyjnego pojęcia wymagania. Wpływa ono bowiem na wiele aspektów systemu. Związkom na diagramach wymagań systemowych poświęcono punkty od 2.2.3 do 2.2.11. Tabelaryczną notację związków omówiono w punkcie 2.3.2.

Bloki, przypadki użycia i testowe przypadki użycia są kategoriami modelowania, istotnymi z punktu widzenia ilustrowania **kontekstu** poszczególnych wymagań oraz nadzoru nad sposobem ich **implementacji** w systemie. Na diagramach wymagań systemowych stosuje się je w połączeniu z odpowiednimi rodzajami zależności. Wymienione kategorie modelowania wykorzystano w punktach 2.2.6, 2.2.8 i 2.2.9.



Rysunek 2.3. *Kategorie modelowania diagramu wymagań systemowych*

Pakiety stanowią mechanizm ogólnego zastosowania, służący do **organizowania** elementów, w tym wymagań i innych kategorii modelowania diagramu wymagań systemowych. Tym samym pakiety i omówione w podrozdziale 7.5 diagramy pakietów są użyteczne w zarządzaniu złożonością modelu wymagań systemowych.

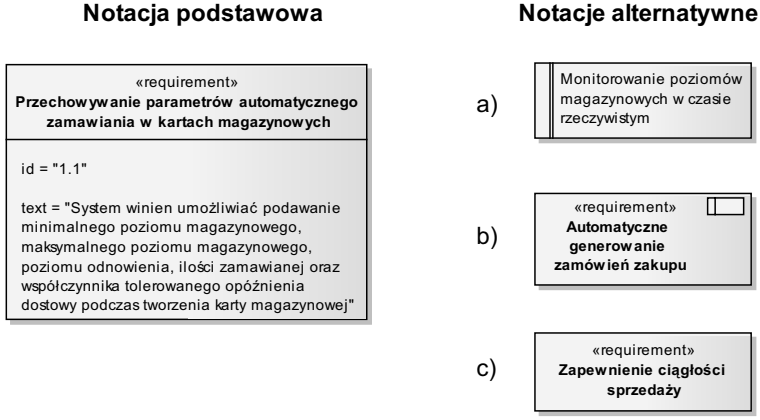
2.2.2. Wymagania

W języku SysML wymagania (ang. *requirements*) symbolizują **kontrakt** pomiędzy organizacją, zamawiającą system, a jego wykonawcami. W zależności od ustaleń zespołu projektowego i zastosowanego narzędzia można wykorzystywać podstawową notację wymagań bądź jedną z notacji alternatywnych, zaprezentowanych na rysunku 2.4. Podstawowymi charakterystykami każdego wymagania są:

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

W praktycznych zastosowaniach wymagania systemowe mają charakter hierarchiczny. W związku z tym wskazane jest nadawanie im numeracji porządkowej zgodnie z konwencją, podkreślającą umiejscowienie danego wymagania w wielopoziomowej strukturze

Rysunek 2.4. *Notacje wymagań*



wymagań systemowych. Szczególnie użyteczna w tym kontekście jest notacja Deweya. Z kolei treść wymagania zawiera spójny, syntetyczny opis właściwości, jaką system powinien się cechować. Jednym z parametrów oceny jakości tworzonego diagramu wymagań systemowych jest stosowanie konsekwentnego stylu treści poszczególnych wymagań, najwłaściwiej rozpoczynającego się od sformułowania „System winien...".

Wymienione cechy mogą zostać w sposób **bezpośredni** ujęte na diagramie, jak zaprezentowano na rysunku 2.4. Spotykane są także zapisy uproszczone (por. rysunek 2.4a, 2.4b i 2.4c). Zapisy te wynikają ze znacznej liczby wymagań, charakteryzujących współcześnie tworzone systemy — a w konsekwencji ze skali złożoności diagramów, składających się na kompletny model wymagań systemowych.

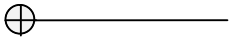
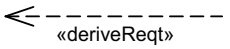
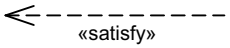
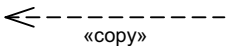
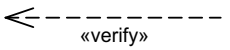
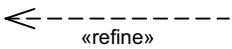
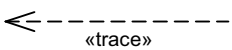
I tak na rysunku 2.4 przedstawiono cztery warianty opisu wymagań. Najbardziej pełna jest notacja wymagania *Przechowywanie parametrów automatycznego zamawiania w kartach magazynowych*, zawierająca oprócz nazwy także numer porządkowy i treść wymagania. Wymaganie *Zapewnienie ciągłości sprzedaży* (rysunek 2.4b) również jest stereotypowane tekstowo, lecz nie uwzględniono w jego przypadku charakterystyk szczegółowych. Z kolei wymagania *Monitorowanie poziomów magazynowych w czasie rzeczywistym* i *Automatyczne generowanie zamówień zakupu* są stereotypowane graficznie (rysunek 2.4a i 2.4b). Wymaganiu *Przechowywanie parametrów automatycznego zamawiania w kartach magazynowych* przydzielono numer porządkowy 1.1. Można z tego wywnioskować, że posiada ono wymaganie nadrzędne o numerze 1.

2.2.3. Rodzaje związków pomiędzy wymaganiami

W diagramie wymagań systemowych poszczególne wymagania łączy się różnymi rodzajami związków (ang. *relationships*). Wskazują one na charakter **logicznej zależności** pomiędzy poszczególnymi wymaganiami. Specyfikacja języka SysML ujmuje 7 podstawowych odmian związków pomiędzy wymaganiami. Związki te wyszczególnia tabela 2.1.

Wszystkie odmiany zależności, ujęte w specyfikacji języka SysML, przedstawiają powiązania pomiędzy parą wymagań: źródłowym i docelowym. Graficznie te dwa

Tabela 2.1. *Rodzaje związków na diagramach wymagań systemowych*

Nazwa	Notacja
zagnieżdżenie	
zależność wyprowadzania	
zależność realizacji	
zależność powielania	
zależność weryfikowania	
zależność precyzowania	
zależność śledzenia	

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.

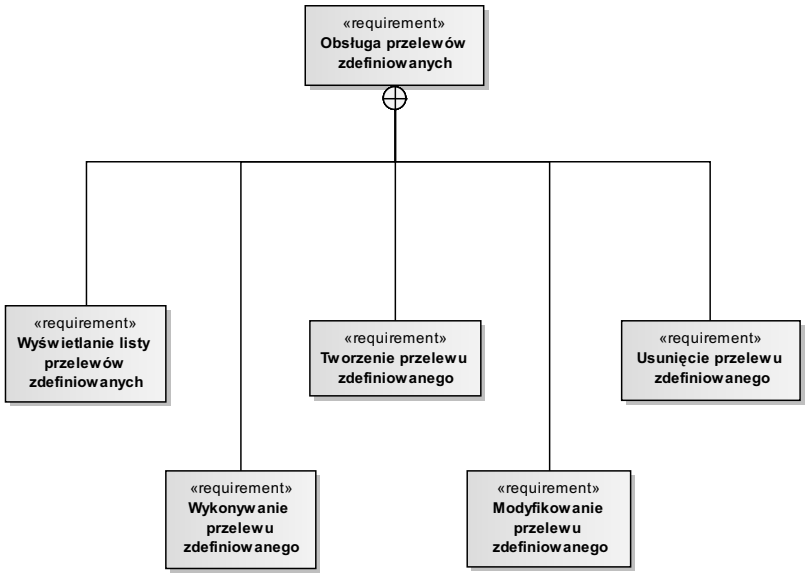
Należy zaznaczyć, iż zależności wyprowadzania, realizacji i precyzowania stanowią specyficzne formy alokacji bezpośredniej. Charakterystykę alokacji zawarto w punkcie 3.3.5.

2.2.4. Zagnieżdżenie

Związkiem najpowszechniej stosowanym w diagramach wymagań systemowych jest zagnieżdżenie (ang. *containment*). 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. Zasada ta nie dotyczy wymagania, zamieszczonego na szczycie hierarchii, które naturalnie nie posiada wymagań nadrzędnych. Wymaganie nadrzędne uznaje się za spełnione tylko i wyłącznie wtedy, gdy zostaną spełnione wszystkie jego wymagania podrzędne — czyli podwymagania powiązane z danym wymaganiem w hierarchię poprzez zastosowanie związków zagnieżdżenia.

Wielokrotne użycie danego wymagania pomiędzy różnymi gałęziami hierarchii wymagań jest możliwe dzięki zależności powielania «copy», omówionej w dalszej części rozdziału. Związek zagnieżdżenia przedstawia rysunek 2.5.

W hierarchii wymagań, zilustrowanej na rysunku 2.5, wymaganiem nadrzędnym jest wymaganie systemowe *Obsługa przelewów zdefiniowanych*. Wymaganie to posiada kilka podwymagań. Należą do nich:



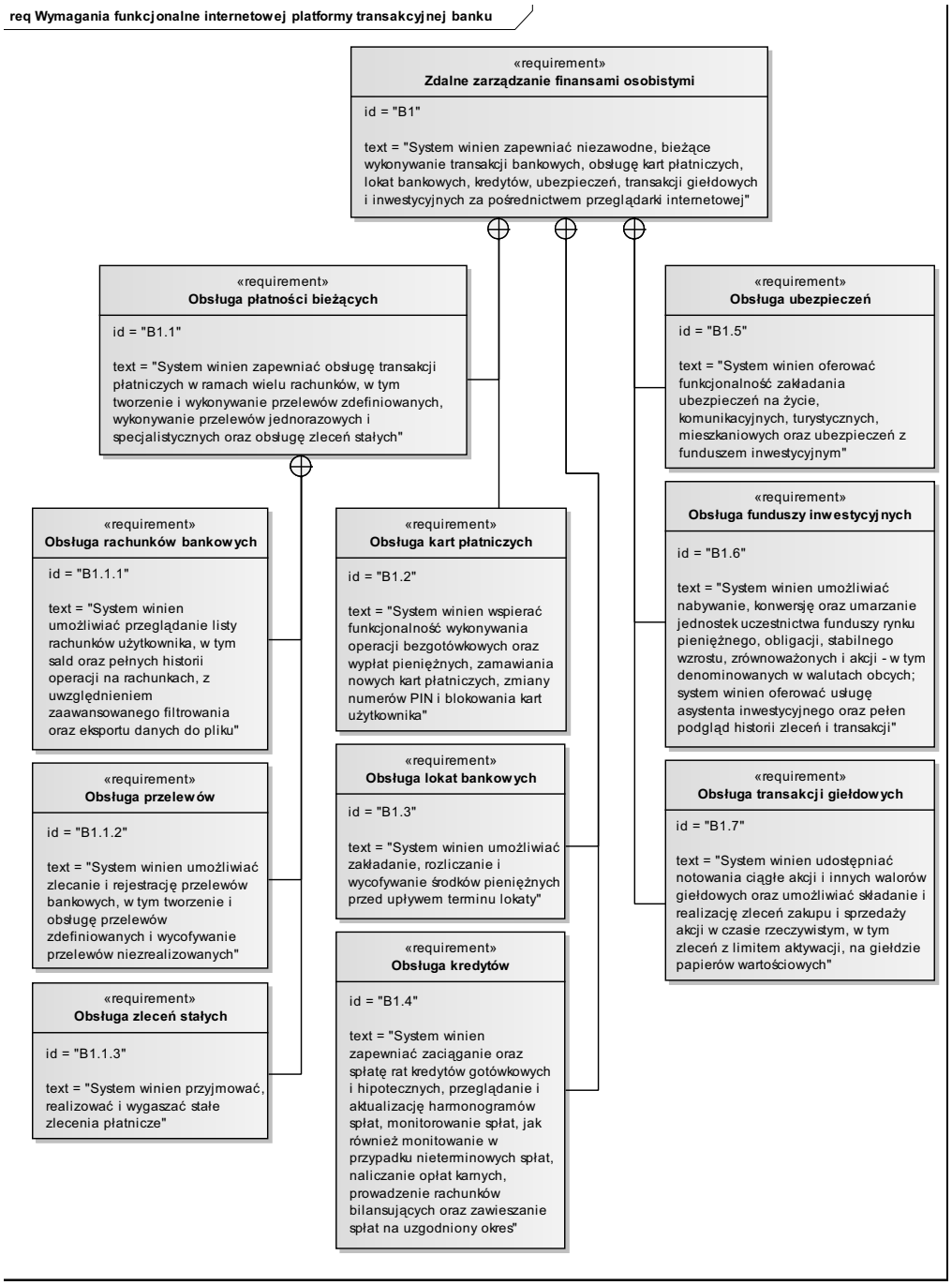
Rysunek 2.5. *Związek zagnieżdżenia w diagramie wymagań systemowych*

- *Wyświetlanie listy przelewów zdefiniowanych*,
- *Wykonywanie przelewu zdefiniowanego*,
- *Tworzenie przelewu zdefiniowanego*,
- *Modyfikowanie przelewu zdefiniowanego*,
- *Usunięcie przelewu zdefiniowanego*.

Kompletny diagram, ujmujący podstawowe wymagania funkcjonalne internetowej platformy transakcyjnej banku wraz z ich numerami porządkowymi oraz treściami, przedstawia rysunek 2.6.

Rysunek 2.6 ujmuje złożoną hierarchię wymagań. Na pierwszym poziomie hierarchii występują wymagania systemowe:

- *Obsługa płatności bieżących*,
- *Obsługa kart płatniczych*,
- *Obsługa lokat bankowych*,
- *Obsługa kredytów*,
- *Obsługa funduszy inwestycyjnych*,
- *Obsługa ubezpieczeń*,
- *Obsługa transakcji giełdowych*.



Rysunek 2.6. *Wymagania funkcjonalne internetowej platformy transakcyjnej banku*

Każde z tych wymagań może podlegać dalszej hierarchizacji z wykorzystaniem związku zagnieżdżenia. I tak na przykład wymaganie *Obsługa płatności bieżących* jest wymaganiem nadrzędnym dla wymagań systemowych:

- *Obsługa rachunków bankowych*,
- *Obsługa przelewów*,
- *Obsługa zleceń stałych*.

2.2.5. Zależność wyprowadzania

Zależność wyprowadzania, wykorzystująca stereotyp «deriveReqt», pozwala na **wyprowadzenie** wymagania docelowego z wymagania źródłowego. Cechy systemu, reprezentowane przez wymaganie docelowe, są pochodnymi cech systemu, reprezentowanych przez wymaganie źródłowe. Zazwyczaj pojedyncze wymaganie źródłowe wspierane jest przez szereg wymagań docelowych, powiązanych osobnymi zależnościami wyprowadzania. I tak, jak przedstawiono na rysunku 2.7, z wymagania *Wykonywanie przelewu jednorazowego* wyprowadzono dwa wymagania: *Wykonywanie przelewu do Urzędu Skarbowego* oraz *Wykonywanie przelewu do Zakładu Ubezpieczeń Społecznych*. Z drugiej strony pojedyncze wymaganie docelowe może korzystać z funkcjonalności kilku różnych wymagań źródłowych.

Zależność wyprowadzania ma charakter bardziej uniwersalny od zagnieżdżenia, gdyż może specyfikować związki pomiędzy wymaganiami, zamieszczonymi w **różnych gałęziach** hierarchicznej struktury wymagań. Obejmuje to także wymagania ujęte na tym samym poziomie hierarchii.

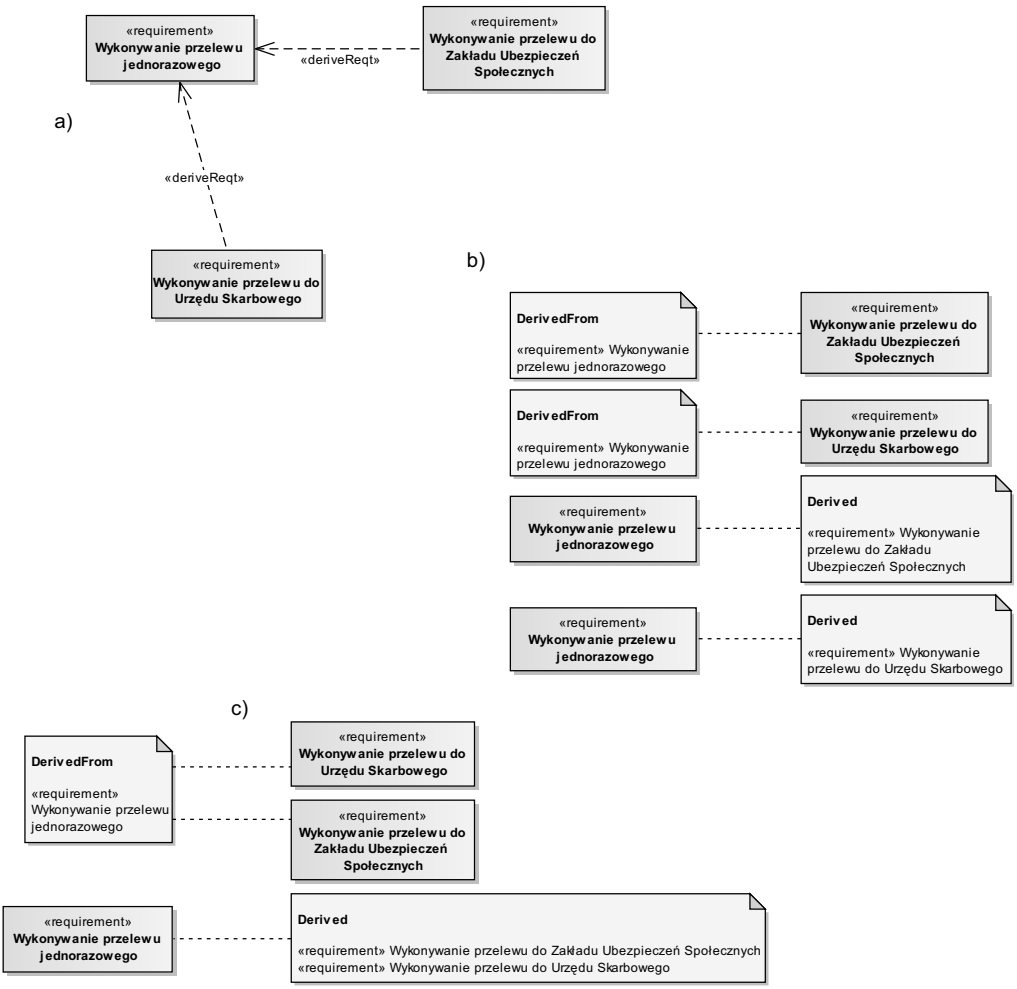
Język SysML oferuje kilka alternatywnych **konwencji** graficznej prezentacji poszczególnych odmian zależności pomiędzy wymaganiami. Wyróżnić można konwencje:

- bezpośrednią,
- notatkową,
- notatkową zagregowaną.

Egzemplifikacja poszczególnych rodzajów zależności będzie konsekwentnie prowadzona we wszystkich wymienionych konwencjach. I tak bezpośrednią notację zależności «deriveReqt» przedstawiono na rysunku 2.7a, notatkową — rysunku 2.7b, a notatkową zagregowaną — na rysunku 2.7c.

2.2.6. Zależność realizacji

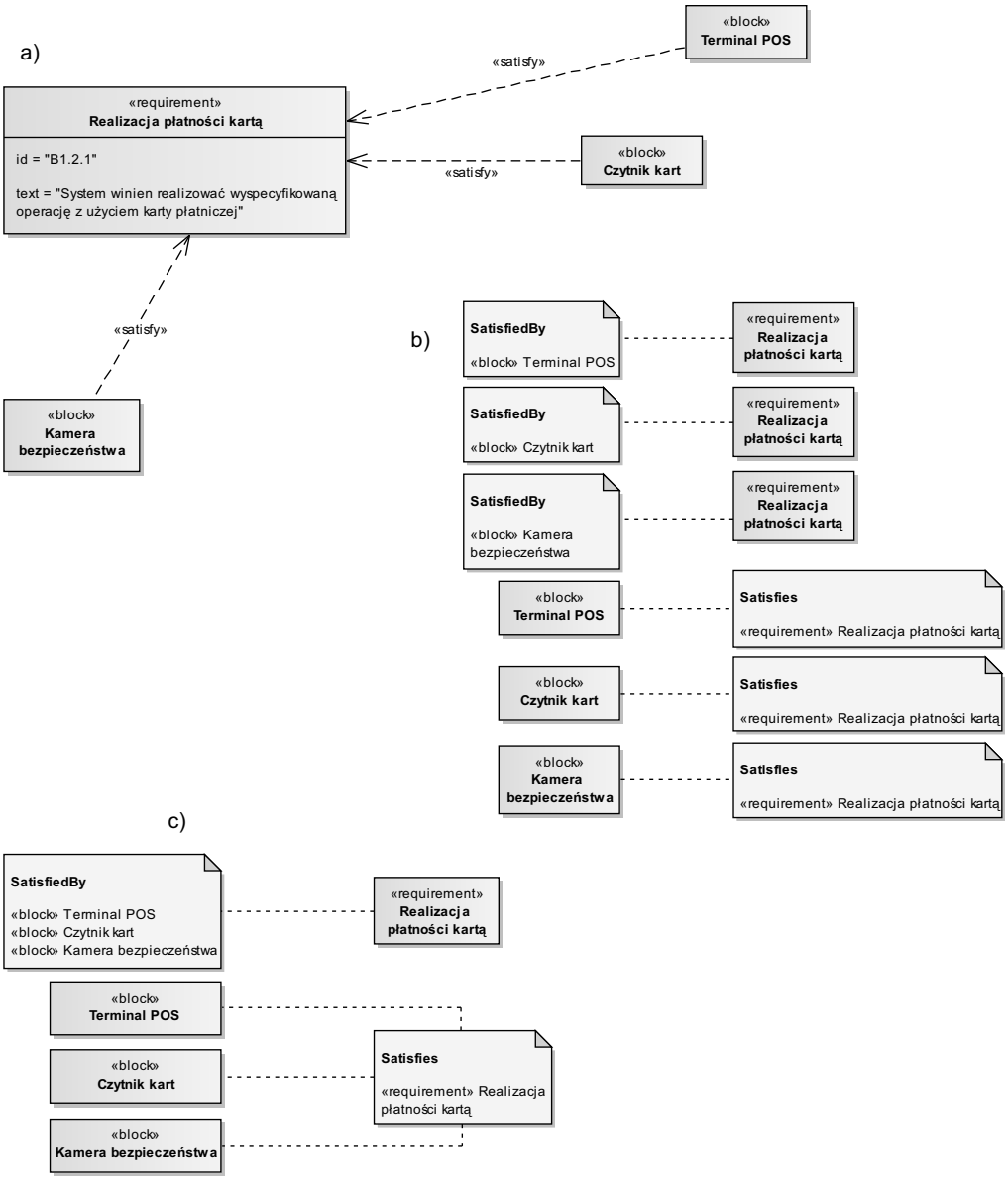
Każde wymaganie systemowe musi zostać **spełnione** poprzez konkretne kategorie modelowania, składające się na ten system. Mogą być to zarówno elementy o charakterze programowym (np. płatność, podsystem zamawiania), jak i sprzętowym (czytnik kart, przełącznik sieciowy itd.). Obie wskazane odmiany najczęściej przyjmują na diagramach postać bloków lub kategorii modelowania grupujących bloki (systemy, podsystemy itp.).



Rysunek 2.7. *Różne konwencje graficznej prezentacji zależności wyprowadzania*

Zadaniem projektanta systemu jest identyfikacja kluczowych elementów, niezbędnych do spełnienia poszczególnych wymagań, oraz wskazanie, które konkretnie wymagania są spełniane przez wyspecyfikowane elementy.

Przypisanie wymagania do spełniającego je elementu można osiągnąć dzięki zależności realizacji, bazującej na stereotypie «`satisfy`». Zależność ta pozwala określać **skutki zmian** w obrębie wymagania wobec elementów od niego zależnych i zarazem wskazywać, które z kluczowych elementów projektu i implementacji systemu są podatne na zmiany w obrębie danego wymagania i jakie ma to implikacje dla tego wymagania. Jak zaprezentowano na rysunku 2.8a, zależność realizacji na diagramie jest skierowana od elementu docelowego, odpowiedzialnego za realizację wymagania (*Terminal POS*, *Czytnik kart*, *Kamera bezpieczeństwa*), do wymagania źródłowego (*Realizacja płatności kartą*). Alternatywne konwencje graficznej prezentacji omawianego związku przedstawia odpowiednio rysunek 2.8b oraz rysunek 2.8c.



Rysunek 2.8. *Różne konwencje graficznej prezentacji zależności realizacji*

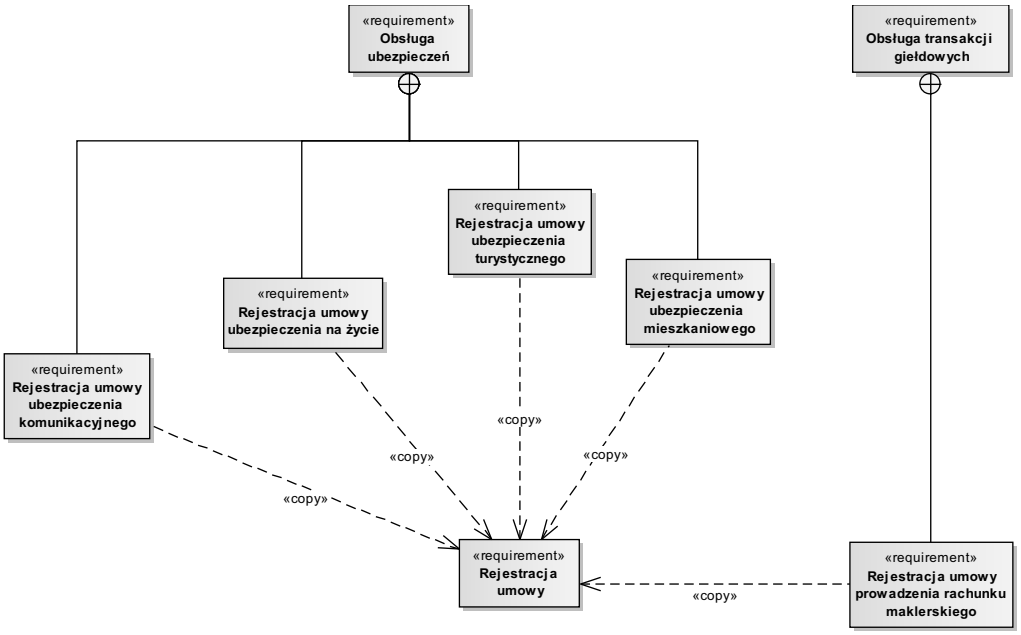
2.2.7. Zależność powielania

W różnych modułach złożonych systemów bardzo często obowiązuje szereg tożsamych wymagań. Praktyka niezależnego definiowania takich samych wymagań w różnych modułach jest niewskazana ze względu na utratę możliwości śledzenia zmian w modelu wymagań oraz ryzyko powstania niespójności modelu wymagań. Jest ona także sprzeczna

z uniwersalną zasadą **ponownego wykorzystania** (ang. *reuse*), konsekwentnie stosowaną w systemach obiektowych. Z tego względu twórcy języka SysML zaproponowali możliwość powielania treści wymagań. Przyjmuje ona formę zależności powielania, oznaczanej stereotypem «copy».

W momencie poprowadzenia zależności powielania pomiędzy dwoma wymaganiami przyjmuje się, że wymaganie docelowe ma charakter **tylko do odczytu**, tj. przyjmuje ono tożsamą treść w stosunku do wymagania źródłowego. Stąd jeśli zamawiający system przeformułuje treść wcześniej zdefiniowanego wymagania, zostanie ona automatycznie zaktualizowana we wszystkich powiązanych wymaganiach docelowych. Zastosowanie tego związku eliminuje zatem konieczność wyszukiwania wszystkich wymagań o analogicznej treści i stosownego wprowadzania korekt. Z kolei numery porządkowe oraz nazwy wymagań źródłowych i docelowych mogą się różnić.

Przykład zastosowania bezpośredniej konwencji graficznej prezentacji związków powielania zamieszczono na rysunku 2.9.



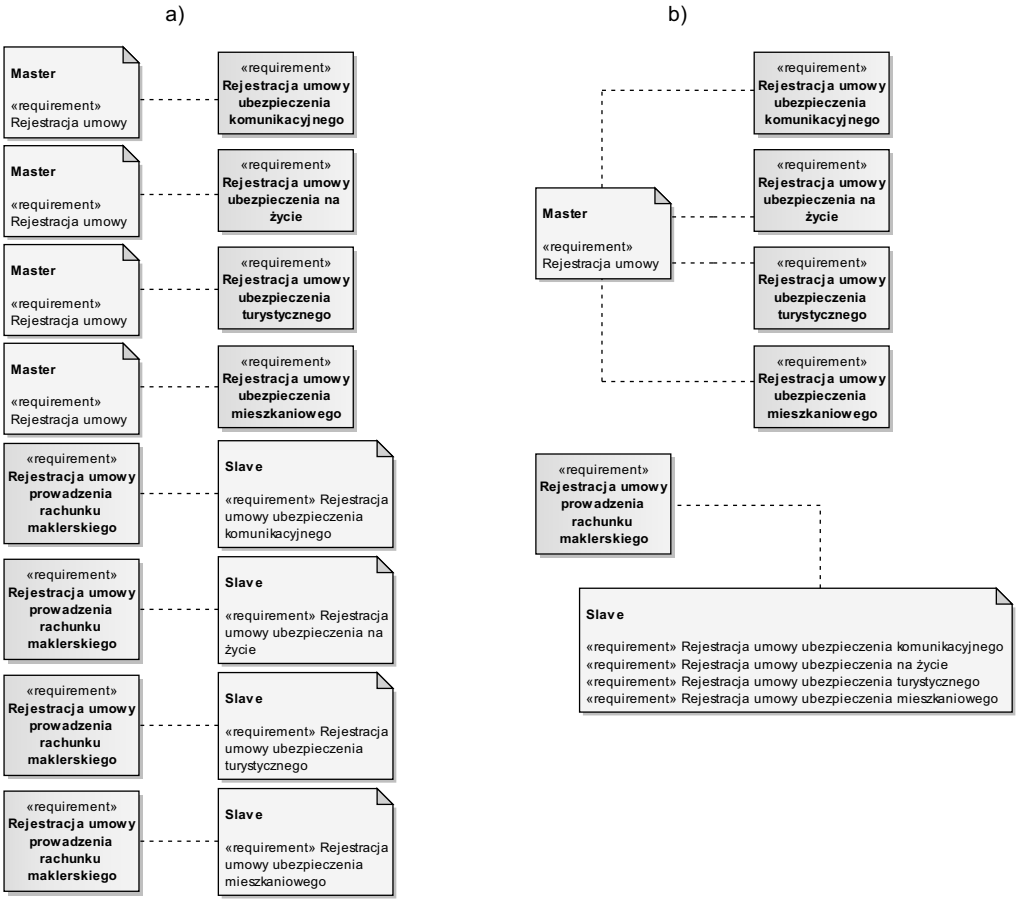
Rysunek 2.9. *Bezpośrednia konwencja graficznej prezentacji zależności powielania*

I tak zaprezentowane na rysunku 2.9 wymaganie systemowe *Obsługa ubezpieczeń* posiada kilka podwymagań. Obejmują one:

- *Rejestrację umowy ubezpieczenia komunikacyjnego*,
- *Rejestrację umowy ubezpieczenia na życie*,
- *Rejestrację umowy ubezpieczenia turystycznego*,
- *Rejestrację umowy ubezpieczenia mieszkaniowego*.

Wszystkie te wymagania szczegółowe powielają treść wymagania *Rejestracja umowy*. Wymaganie *Rejestracja umowy* stanowi także wzorzec dla wymagania systemowego *Rejestracja umowy prowadzenia rachunku maklerskiego*. To ostatnie jest podwymaganiem *Obsługi transakcji giełdowych*.

Z kolei konwencje notatkowe graficznej prezentacji zależności powielania zilustrowano na rysunku 2.10.



Rysunek 2.10. *Konwencje notatkowe graficznej prezentacji zależności powielania*

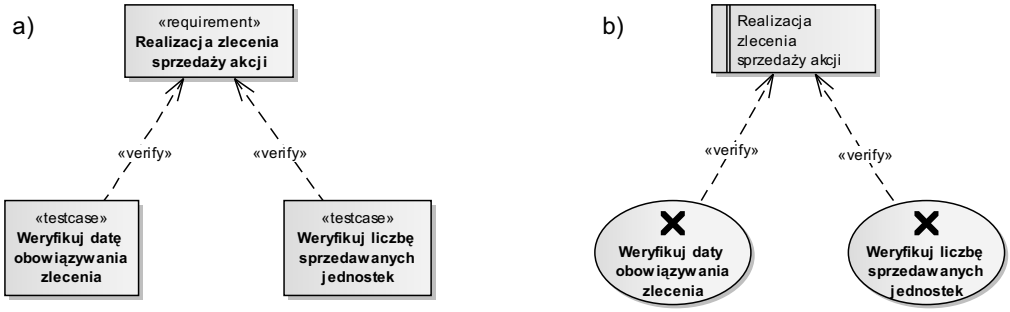
2.2.8. Zależność weryfikowania

Dobrą praktyką modelowania systemów informatycznych jest weryfikowanie poszczególnych wymagań pod kątem tego, czy zostały one **zrealizowane** podczas kodowania systemu. W tym celu tworzy się stosowne testy. Testy formalne charakteryzują się posiadaniem konkretnego zestawu warunków wejściowych oraz oczekiwanego efektu testu po jego wywołaniu. W języku SysML testy formalne wiąże się z wymaganiami z wykorzystaniem zależności weryfikowania, oznaczonej stereotypem «`verify`».

Zależność weryfikowania wyprowadzana jest od elementu docelowego, którym jest tzw. **testowy przypadek użycia**. Charakteryzuje on procedurę testu. Do pojedynczego wymagania przydzielić można szereg testowych przypadków użycia. Każdy przypadek można osobno udokumentować diagramami dynamiki języka SysML, na przykład diagramem sekwencji lub czynności. Testowe przypadki użycia można na diagramach wymagań systemowych stereotypować zarówno tekstowo (rysunek 2.11a), jak i graficznie (rysunek 2.11b). Testowemu przypadkowi użycia można także przypisać dodatkowe stereotypy testowania, odpowiadające różnym metodom testowania. Przykładami takich stereotypów mogą być:

- «*analyze*»,
- «*inspect*»,
- «*demonstrate*»,
- «*test*».

Zróżnicowanie testowych przypadków użycia pozwala na realizację różnych **procedur weryfikacyjnych**. Bezpośrednią konwencję graficznej prezentacji zależności weryfikowania przedstawiono na rysunku 2.11.



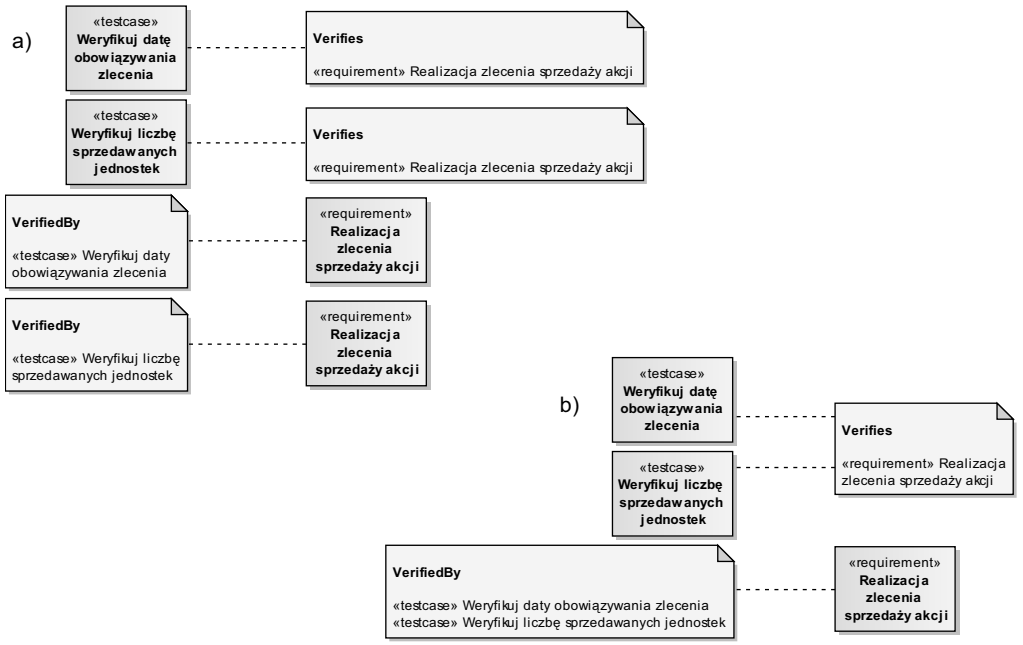
Rysunek 2.11. *Bezpośrednia konwencja graficznej prezentacji zależności weryfikowania*

I tak zamieszczone na rysunku 2.11 wymaganie systemowe *Realizacja zlecenia sprzedaży akcji* podlega weryfikacji przez testowe przypadki użycia *Weryfikuj datę obowiązywania zlecenia* oraz *Weryfikuj liczbę sprzedawanych jednostek*. Rysunek 2.11b ujmuje alternatywną, graficzną notację poszczególnych kategorii modelowania.

Zależność weryfikowania nabiera szczególnego znaczenia w iteracyjno-przyrostowym cyklu życia systemu — w jego kolejnych **iteracjach** (minicyklach życia systemu) wykonuje się bowiem zadania z zakresu poszczególnych dyscyplin, od modelowania biznesowego do wdrożenia i testowania (por. rysunek 2.1). Konwencje notatkowe graficznej prezentacji zależności weryfikowania zilustrowano na rysunku 2.12.

2.2.9. Zależność precyzowania

Zależność precyzowania, oznaczana stereotypem «`refine`», pozwala na wprowadzenie do diagramu wymagań systemowych licznych detali, reprezentowanych poprzez docelowe elementy związku. Detale te mają na celu ułatwienie zrozumienia **sensu wymagania**



Rysunek 2.12. *Konwencje notatkowe graficznej prezentacji zależności weryfikowania*

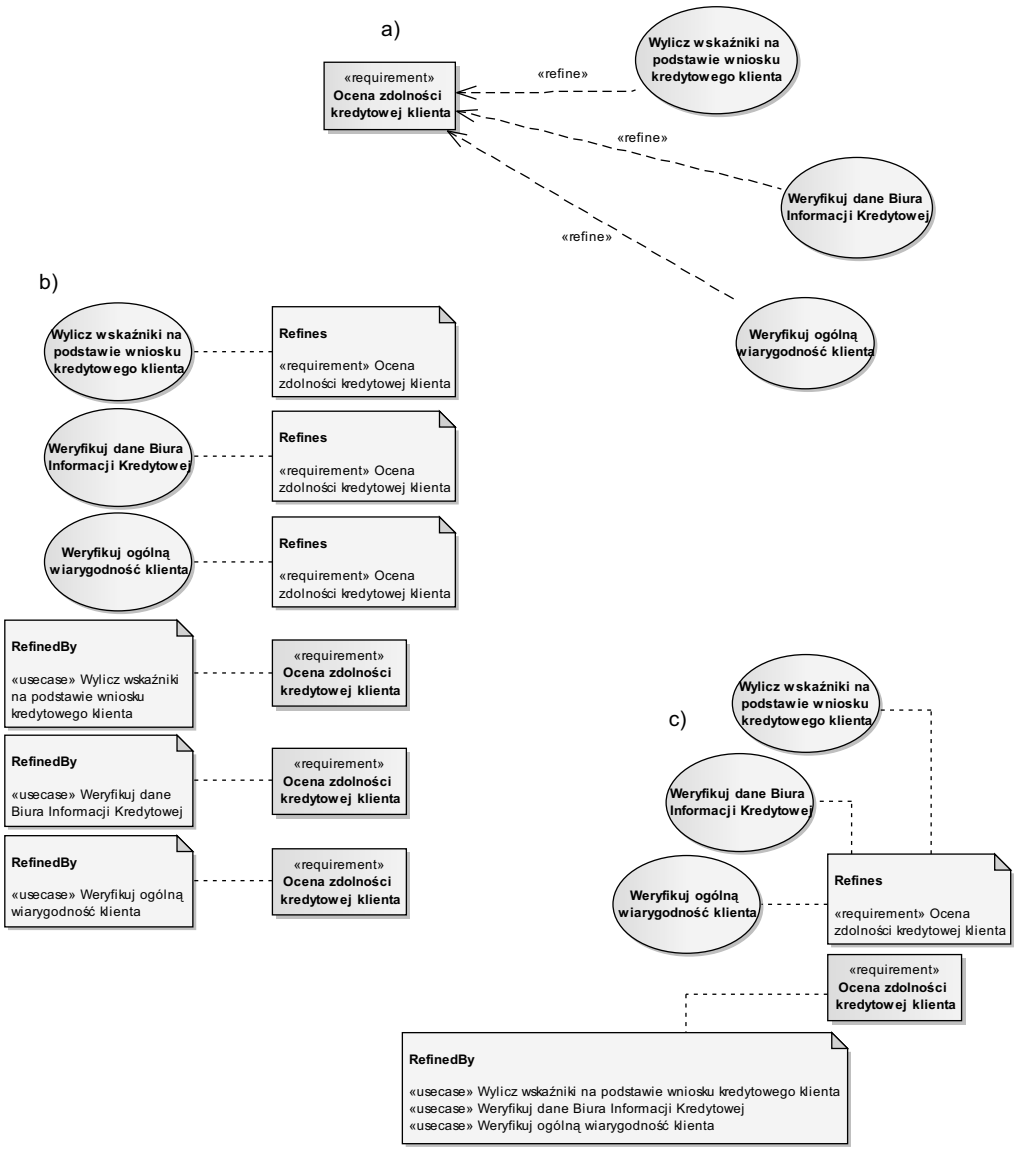
poprzez wzbogacenie go o elementy modelowania lub ich zestawy, takie jak kategorie modelowania języka SysML, projektowania interfejsu, osobne specyfikacje tekstowe lub standardy. Zwięzła treść wymagania źródłowego może być tym samym sformalizowana lub bardziej precyzyjnie zdefiniowana. Zależność ta stwarza możliwość wyjaśnienia ogólnego tekstu zapisanego w wymaganiu źródłowym. Przykłady zastosowania zależności precyzowania zaprezentowano na rysunku 2.13.

I tak zamieszczone na rysunku 2.13 wymaganie *Ocena zdolności kredytowej klienta* precyzowane jest poprzez wiele kategorii modelowania. W tym konkretnym zastosowaniu są to następujące przypadki użycia:

- *Weryfikuj ogólną wiarygodność klienta*,
- *Weryfikuj dane Biura Informacji Kredytowej*,
- *Wylicz wskaźniki na podstawie wniosku kredytowego klienta*.

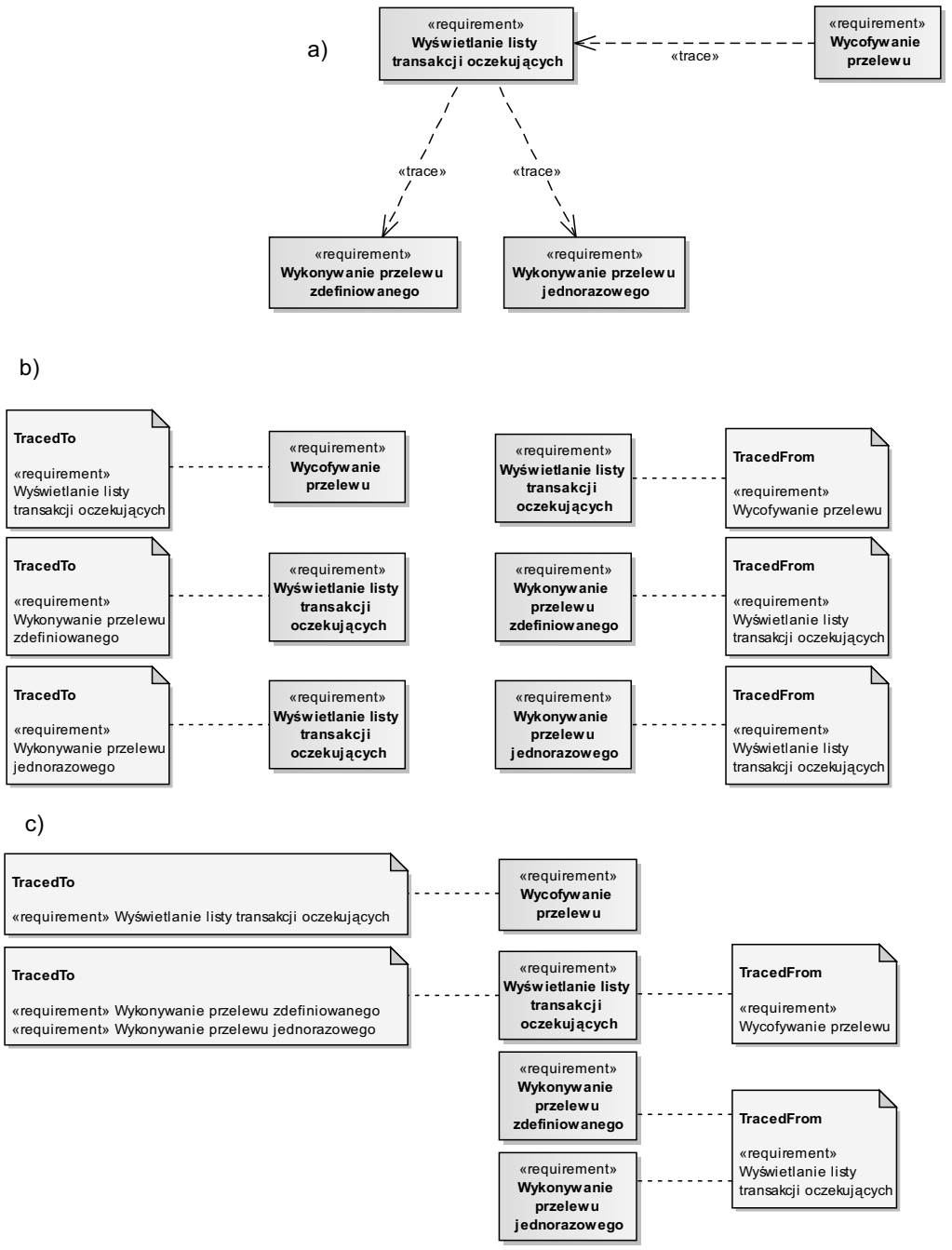
2.2.10. Zależność śledzenia

Zależność śledzenia, oznaczana stereotypem «`trace`», umożliwia zaprezentowanie **nieformalnego związku** pomiędzy wymaganiem a dowolnym elementem modelu systemu, w tym innym wymaganiem. Dokumentacja języka SysML pozostawia znaczną swobodę stosowania tego typu związków. Na rysunku 2.14 wykorzystano zależność śledzenia do podkreślenia następstwa czasowego realizacji usług systemowych, wynikających z zaprezentowanych wymagań.



Rysunek 2.13. *Różne konwencje graficznej prezentacji zależności precyzowania*

I tak, zgodnie z rysunkiem 2.14, *Wycofanie przelewu* jest możliwe tylko za pośrednictwem listy transakcji oczekujących. Jeśli jakieś wymaganie znajduje się na wspomnianej liście, jego obsługę zapewnia funkcjonalność systemu, zaimplementowana w konsekwencji wniesienia wymagania *Wyświetlanie listy transakcji oczekujących*. Z kolei aby transakcja znalazła się na tej liście, wcześniej należy wykonać funkcjonalność wynikającą z wymagania *Wykonywanie przelewu jednorazowego* bądź *Wykonywanie przelewu zdefiniowanego*.



Rysunek 2.14. *Różne konwencje graficznej prezentacji zależności śledzenia*

2.2.11. Analiza porównawcza zależności

Jak wynika z dokonanej w powyższych punktach charakterystyki sześciu odmian zależności, opisują one związki pomiędzy wymaganiami źródłowymi oraz wymaganiami docelowymi bądź docelowymi elementami modelowania. Syntetycznie zagadnienie to **podsumowuje** tabela 2.2.

Tabela 2.2. *Porównanie zależności na diagramach wymagań systemowych*

Rodzaj zależności	Stereotyp	Wymaganie źródłowe	Wymaganie docelowe	Docelowa kategoria modelowania
zależność wyprowadzania	«deriveReqt»	x	x	
zależność realizacji	«satisfy»	x		x
zależność powielania	«copy»	x	x	
zależność weryfikowania	«verify»	x		x
zależność precyzowania	«refine»	x		x
zależność śledzenia	«trace»	x	x	x

Zastosowanie większości z zaprezentowanych w tabeli 2.2 rodzajów zależności zilustrowano na rysunku 2.15.

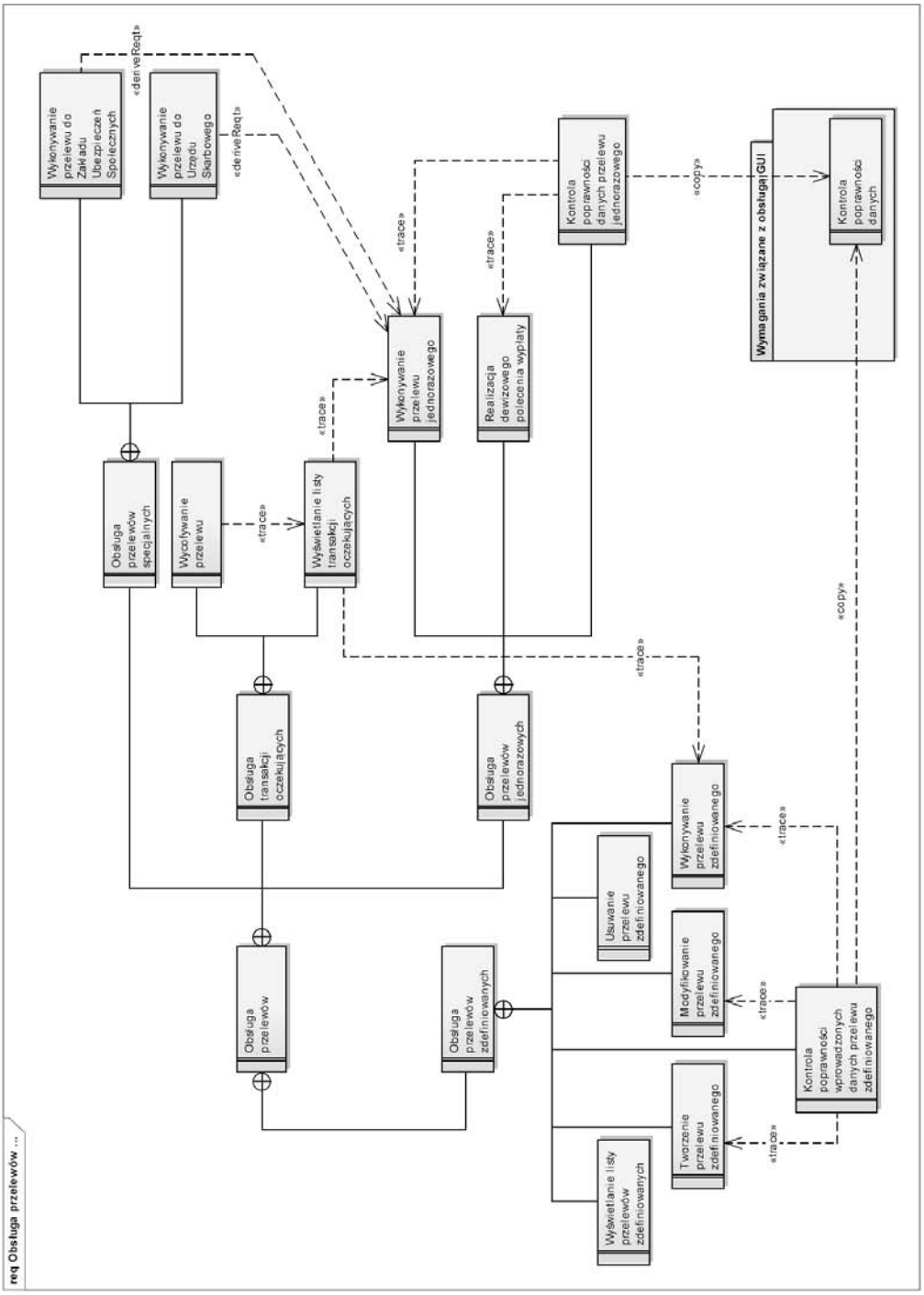
I tak rysunek 2.15 przedstawia zestaw wymagań systemowych w odniesieniu do systemu transakcyjnego banku. Przedmiotem wymagań jest obsługa przelewów. Nadrzędnym wymaganiem w hierarchii jest właśnie wymaganie *Obsługa przelewów*. Posiada ono cztery bezpośrednie podwymagania:

- *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 trzema ostatnimi wymaganiami zależnością śledzenia powiązane jest wymaganie *Kontrola poprawności wprowadzonych danych przelewu zdefiniowanego*. Oznacza to, że wykonanie, tworzenie lub modyfikowanie przelewu wiąże się z wywołaniem funkcjonalności, kontrolującej poprawność danych, wprowadzonych do poszczególnych formatek.



Rysunek 2.15. *Studium diagramu wymagań systemowych banku internetowego*

Z realizacją *Obsługi przelewów jednorazowych* wiążą się następujące wymagania szczegółowe:

- *Wykonywanie przelewu jednorazowego*,
- *Realizacja dewizowego polecenia wypłaty*,
- *Kontrola poprawności danych przelewu jednorazowego*.

Funkcjonalność zaimplementowana wskutek sformułowania tego ostatniego wymagania jest automatycznie wywoływana zarówno przy *Wykonywaniu przelewu jednorazowego*, jak i *Realizacji dewizowego polecenia wypłaty*. Oba wymienione wyżej wymagania, związane z weryfikacją spójności danych, powielają wzorcową funkcjonalność wymagania *Kontrola poprawności danych*, zamieszczonego z kolei w pakiecie *Wymagania związane z obsługą GUI*.

Obsługa transakcji oczekujących dzieli się na podwymagania *Wycofywanie przelewu* i *Wyświetlanie listy transakcji oczekujących*, przy czym wycofywanie jest uzależnione od wyświetlania. Wspomnianą relację obrazuje zależność «*trace*», zamieszczona pomiędzy tymi wymaganiami. Analogiczna zależność łączy wymaganie *Wyświetlanie listy transakcji oczekujących* z wymaganiami *Wykonywanie przelewu zdefiniowanego* i *Wykonywanie przelewu jednorazowego*.

Ostatnim z wymagań, podlegających dalszej hierarchizacji, jest wymaganie systemowe *Obsługa przelewów specjalnych*. Dzieli się ono na *Wykonywanie przelewu do Zakładu Ubezpieczeń Społecznych* i *Wykonywanie przelewu do Urzędu Skarbowego*. Obydwa te wymagania wywodzą się z wymagania *Wykonywanie przelewu jednorazowego*, co zobrazowano zależnością wyprowadzania «`deriveReqt`».

2.3. Zaawansowana specyfikacja wymagań oraz związków

Poza kluczowymi właściwościami podstawowych kategorii modelowania diagramu wymagań systemowych, tj. wymagań oraz związków, istnieją następujące **zaawansowane koncepcje**, związane w języku SysML ze specyfikowaniem wymagań:

- tabelaryczna specyfikacja wymagań,
- tabelaryczna specyfikacja związków,
- rozszerzone wymagania systemowe,
- stereotypowanie rozszerzonych wymagań systemowych.

Wymienione koncepcje omówiono w kolejnych punktach niniejszego rozdziału.

2.3.1. Tabelaryczna specyfikacja wymagań

W przypadku obszernego opisu treści wymagań użyteczna staje się tabelaryczna reprezentacja wymagań systemowych. Obligatoryjnie musi zawierać ona co najmniej trzy kolumny, tj. numer porządkowy, nazwę oraz treść wymagania. Numeracja porządkowa może opierać się na **systemie klasyfikacji dziesiętnej Deweya**, wiernie oddającym hierarchiczne zależności pomiędzy poszczególnymi wymaganiami systemowymi. W kolumnie *Nazwa* umieszcza się naturalnie nazwę wymagania, podczas gdy *Treść* może zawierać nawet bardzo drobiazgowy opis wymagania. W miarę potrzeb wprowadza się dodatkowe kolumny tabeli. Przykład tabelarycznej specyfikacji wymagań systemowych zaprezentowano w tabeli 2.3.

Tabela 2.3. *Tabelaryczna reprezentacja wymagań systemowych*

Numer porządkowy	Nazwa	Treść
B1	*Zdalne zarządzanie finansami osobistymi*	System winien zapewniać niezawodne, bieżące wykonywanie transakcji bankowych, obsługę kart płatniczych, lokat bankowych, kredytów, ubezpieczeń, transakcji giełdowych i inwestycyjnych za pośrednictwem przeglądarki internetowej
B1.1	*Obsługa płatności bieżących*	System winien zapewniać obsługę transakcji płatniczych w ramach wielu rachunków, w tym tworzenie i wykonywanie przelewów zdefiniowanych, wykonywanie przelewów jednorazowych i specjalistycznych oraz obsługę zleceń stałych
B1.2	*Obsługa kart płatniczych*	System winien wspierać funkcjonalność wykonywania operacji bezgotówkowych oraz wypłat pieniężnych, zamawiania nowych kart płatniczych, zmiany numerów PIN i blokowania kart użytkownika
B1.3	*Obsługa lokat bankowych*	System winien umożliwiać zakładanie, rozliczanie i wycofywanie środków pieniężnych przed upływem terminu lokaty
B1.4	*Obsługa kredytów*	System winien zapewniać zaciąganie oraz spłatę rat kredytów gotówkowych i hipotecznych, przeglądanie i aktualizację harmonogramów spłat, monitorowanie spłat, jak również monitowanie w przypadku nieterminowych spłat, naliczanie opłat karnych, prowadzenie rachunków bilansujących oraz zawieszanie spłat na uzgodniony okres
B1.5	*Obsługa ubezpieczeń*	System winien oferować funkcjonalność zakładania ubezpieczeń na życie, komunikacyjnych, turystycznych, mieszkaniowych oraz ubezpieczeń z funduszem inwestycyjnym
B1.6	*Obsługa funduszy inwestycyjnych*	System winien umożliwiać nabywanie, konwersję oraz umarzanie jednostek uczestnictwa funduszy rynku pieniężnego, obligacji, stabilnego wzrostu, zrównoważonych i akcji — w tym denominowanych w walutach obcych; system winien oferować usługę asystenta inwestycyjnego oraz pełen podgląd historii zleceń i transakcji
B1.7	*Obsługa transakcji giełdowych*	System winien udostępniać notowania ciągłe akcji i innych walorów giełdowych oraz umożliwiać składanie i realizację zleceń zakupu i sprzedaży akcji w czasie rzeczywistym, w tym zleceń z limitem aktywacji, na giełdzie papierów wartościowych

2.3.2. Tabelaryczna specyfikacja związków

Podobnie jak w przypadku wymagań, istnieje możliwość posługiwania się **tabelaryczną specyfikacją związków**. Tabela taka jest bardziej złożona, gdyż prócz podstawowych cech wymagań źródłowych i docelowych — tj. numeru porządkowego i treści — wymaga wyspecyfikowania rodzaju związku. Tabelaryczną reprezentację związków ilustruje tabela 2.4.

Tabela 2.4. *Tabelaryczna reprezentacja związków*

Wymaganie docelowe		Rodzaj związku	Wymaganie źródłowe	
Numer porządkowy	**Nazwa**		**Numer porządkowy**	**Nazwa**
B1.1.2.1.5	*Wykonywanie przelewu zdefiniowanego*	zagnieżdżenie	B1.1.2.1	*Obsługa przelewów zdefiniowanych*
B1.1.2.2.1	*Wykonywanie przelewu jednorazowego*	zagnieżdżenie	B1.1.2.2	*Obsługa przelewów jednorazowych*
B1.1.2.3.1	*Wyświetlanie listy transakcji oczekujących*	zależność śledzenia	B1.1.2.1.5	*Wykonywanie przelewu zdefiniowanego*
B1.1.2.3.1	*Wyświetlanie listy transakcji oczekujących*	zależność śledzenia	B1.1.2.2.1	*Wykonywanie przelewu jednorazowego*
B1.1.2.3.2	*Wycofywanie przelewu*	zależność śledzenia	B1.1.2.3.1	*Wyświetlanie listy transakcji oczekujących*
B1.1.2.2.2	*Wykonywanie przelewu do Zakładu Ubezpieczeń Społecznych*	zależność wyprowadzania	B1.1.2.2.1	*Wykonywanie przelewu jednorazowego*
B1.1.2.2.3	*Wykonywanie przelewu do Urzędu Skarbowego*	zależność wyprowadzania	B1.1.2.2.1	*Wykonywanie przelewu jednorazowego*

2.3.3. Rozszerzone wymagania systemowe

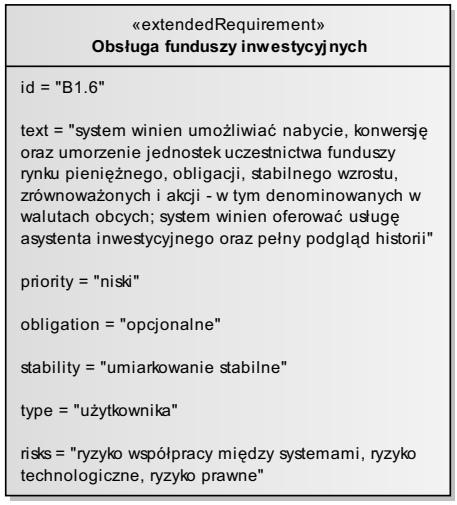
Dotychczasowe rozważania skupiały się na opisie ogólnych cech wymagań oraz związków. Cechy te w języku SysML można swobodnie **rozszerzać** poprzez przypisanie wymaganiom lub związkom dodatkowych stereotypów, a w przypadku wymagania — także właściwości i sekcji.

Poszczególne właściwości rozszerzonego wymagania systemowego (ang. *extended requirement*) mogą przybierać następujące wartości:

- **priorytet wymagania** (ang. *priority*) — wskazujący na kolejność implementowania wymagań, na przykład niski/średni/wysoki;
- **istotność wymagania** (ang. *obligation*) — czyli wyspecyfikowanie, czy dane wymaganie można pominąć w dalszych fazach tworzenia systemu ze względu na czas lub koszty, na przykład obligatoryjne/opcjonalne;

- **stabilność wymagania** (ang. *stability*) — prawdopodobieństwo redefinicji wymagania systemowego w trakcie implementowania systemu, na przykład niestabilne/mało stabilne/umiarkowanie stabilne/wysoce stabilne/całkowicie stabilne;
- **typ wymagania** (ang. *type*) — źródło pochodzenia wymagania, na przykład użytkownika/wzorcowe/własne;
- **różne rodzaje ryzyka implementacji wymagania** (ang. *risks*) — zwięzła lista podstawowych zagrożeń, związanych z danym wymaganiem; poszczególne typy ryzyka charakteryzuje się opisowo.

Na rysunku 2.16 przedstawiono rozszerzone wymaganie systemowe. Dla odróżnienia od wymagania standardowego zostało ono oznaczone stereotypem «extendedRequirement». Oprócz standardowych właściwości, tj. numeru porządkowego i treści wymagania, zawiera ono wiele właściwości dodatkowych.



Rysunek 2.16. *Rozszerzone wymaganie systemowe*

2.3.4. Stereotypowanie rozszerzonych wymagań systemowych

Do diagramów wymagań systemowych powszechnie wprowadza się dodatkowe stereotypy, rozszerzające funkcjonalność wymagań bazowych. Dobór dodatkowych stereotypów jest ściśle uzależniony od dziedziny przedmiotowej i preferencji zespołu, projektującego dany system. Każde wymaganie z przydzielonym niestandardowym stereotypem klasyfikuje się jako rozszerzone wymaganie systemowe. Dodatek C dokumentacji języka SysML w wersji 1.1 proponuje zastosowanie stereotypów, wskazujących na **specyfikę** wymagań systemowych. I tak dodatkowe odmiany rozszerzonych wymagań systemowych obejmują:

- wymagania funkcjonalne,
- wymagania interfejsowe,
- wymagania wydajnościowe,
- wymagania fizyczne,
- ograniczenia projektowe.

Zostały one scharakteryzowane i zilustrowane w tabeli 2.5.

Tabela 2.5. *Dodatkowe stereotypy wymagania zaawansowanego*

Nazwa	Stereotyp	Charakterystyka	Przykład
Wymaganie funkcjonalne	«functional ↪Requirement»	Reprezentuje usługi, które system musi oferować bez uwzględniania uwarunkowań technologicznych	«functionalRequirement» **Zamieszczenie apletu czatu w systemie e-learningowym** id = "F2.1.6" text = "System winien umożliwiać przeprowadzanie czatu, dostępnego dla wszystkich uczestników zdefiniowanej grupy studenckiej oraz prowadzącego przy stopniu dostępności szacowanym na 99%"
Wymaganie interfejsowe	«interface ↪Requirement»	Dotyczy wejściowych i wyjściowych interfejsów systemu	«interfaceRequirements» **Transmisja danych w czasie rzeczywistym** id = "I4.1.3" text = "System winien zapewniać transmisję danych tekstowych, dźwiękowych i graficznych w czasie rzeczywistym w systemie e-learningowym zgodnie z wewnętrzną specyfikacją AFS/23/2009"
Wymaganie wydajnościowe	«performance ↪Requirement»	Dotyczy wolumenu pracy wykonanej przez system w określonym czasie i przy zaangażowaniu określonych zasobów	«performanceRequirement» **Krótki czas nawiązywania połączenia z systemem bankowym** id = "P12.1.3" text = "System winien zapewnić maksymalny czas autoryzacji karty płatniczej nie wyższy niż 5 sekund"
Wymaganie fizyczne	«physical ↪Requirement»	Dotyczy charakterystyki sprzętowej oraz fizycznych ograniczeń systemu lub jego elementów składowych	«physicalRequirement» **Dostępność terminali bankowych** id = "F1.2.1" text = "System winien bazować na infrastrukturze sieciowej, umożliwiającej podłączenie co najmniej 2500 terminali bankowych w regionie"

Tabela 2.5. *Dodatkowe stereotypy wymagania zaawansowanego — ciąg dalszy*

Nazwa	Stereotyp	Charakterystyka	Przykład
Ograniczenie projektowe	«design ↪Constraint»	Dotyczy ograniczeń projektowych, takich jak wykorzystanie technologii własnościowych	«designRequirement» **Wykorzystanie bazy danych Oracle** id = "D1.1.23" text = "Ze względu na integrację z innymi systemami firmy, system winien być oparty na bazie danych Oracle"

W języku SysML można w zależności od potrzeb wprowadzać własne, autorskie stereotypy, tak jak w języku UML.

Profile UML 2.x w praktyce

Język inżynierii systemów

SysML

Architektura i zastosowania

Stanisław Wrycza
Bartosz Marcinkowski

SysML, czyli Systems Modeling Language, to nowy obiektowy język modelowania systemów. W prostej linii wywodzi się on z języka UML, który stanowił do tej pory swego rodzaju standard w inżynierii oprogramowania. SysML został dostosowany do specyficznych potrzeb inżynierów systemowych, zajmujących się projektami w sposób całościowy. Pozwala na specyfikację, analizę, projektowanie i weryfikację złożonych systemów technicznych i gospodarczych, a dzięki swoim dużym możliwościom i elastyczności w ciągu kilku lat zdołał zdobyć liczną rzeszę profesjonalnych użytkowników.

Opanowanie arkanów posługiwania się tym narzędziem ułatwi książka „Język inżynierii systemów SysML. Architektura i zastosowania. Profile UML 2.x w praktyce". Pierwsza na polskim rynku pozycja poświęcona SysML stanowi jednocześnie doskonałe wprowadzenie w zagadnienia inżynierii systemowej, zawiera szczegółowy opis architektury języka oraz prezentuje najważniejsze koncepcje związane z jego zastosowaniem. Książka niemal w całości przedstawia różnego typu diagramy, a zamieszczone w niej liczne przykłady ułatwią zrozumienie nawet najbardziej skomplikowanych zagadnień i umożliwią sprawne poruszanie się po treści oraz uzupełnienie wiedzy w oparciu o publikacje innych autorów.

Poznaj język SysML, opierając się na wiedzy najlepszych specjalistów w tej dziedzinie

- Struktura, historia i zastosowania języka SysML
- Diagram wymagań systemowych
- Diagram definiowania bloków
- Diagram bloków wewnętrznych
- Diagram parametryczny
- Rozszerzony diagram czynności
- Diagramy UML4SysML

Stanisław Wrycza i **Bartosz Marcinkowski** są pracownikami Katedry Informatyki Ekonomicznej na Wydziale Zarządzania Uniwersytetu Gdańskiego. Zajmują się problematyką teorii i zastosowań współczesnych metodyk, technik i narzędzi modelowania systemów informatycznych. Są współautorami takich książek z tej tematyki, jak *Język UML 2.0 w modelowaniu systemów informatycznych*, *Język UML 2.1. Ćwiczenia*, *Tablice informatyczne*, *UML czy też Systems Analysis and Design for Advanced Modeling Methods. Best Practices*. Ich prace były prezentowane i publikowane na licznych konferencjach – w tym międzynarodowych ISECON, BIR, ISD i AIS SIGSAND Symposium.

Cena 34,00 zł

helion.pl
księgarnia internetowa

Nr katalogowy: 5347

Księgarnia internetowa:
http://helion.pl

Zamówienia telefoniczne:
0 801 339900
0 601 339900

Zamów najnowszy katalog:
http://helion.pl/katalog
Zamów informacje o nowościach:
http://helion.pl/nowosci
Zamów cennik:
http://helion.pl/cennik

Wydawnictwo Helion
ul. Kościuszki 1c, 44-100 Gliwice
44-100 Gliwice, skr. poczt. 462
032 230 98 63
http://helion.pl
e-mail: helion@helion.pl

ISBN 978-83-246-2541-3

Informatyka w najlepszym wydaniu