ACTA UNIVERSTIATIS LODZIENSIS
FOLIA OECONOMICA 183, 2004

*Konrad Woźniacki**

# ZALETY ZASTOSOWANIA UNIFIED MODELING LANGUAGE W ANALIZIE BANKOWYCH SYSTEMÓW INFORMATYCZNYCH

## Charakterystyka problemu

Przemiany zachodzące w bankach oraz w ich rynkowym otoczeniu zmieniły ich dotychczasowy, tradycyjny model działalności. Aktualnie prym wiedzie podejście zorientowane na klienta, w którym chęć skutecznej reakcji na jego potrzeby wymusza radykalną poprawę obsługi procesów decyzyjnych. Zasadnicze efekty w tej dziedzinie niesie wdrożenie nowoczesnych bankowych systemów informatycznych oraz koncepcji hurtowni danych. Wszystkie aktualnie stosowane rozwiązania wynikają z konieczności obsługi nowych, dynamicznych procesów biznesowych zachodzących wewnątrz organizacji. Są to procesy nie w pełni zrozumiałe dla informatyka, znane często jedynie specjalistom zatrudnionym w analizowanym banku. Zastosowanie podejścia strukturalnego związane jest z przedstawieniem klientowi opracowania dedykowanego programiście. Ten sposób widzenia banku skutkuje wyraźnym dysonansem między statyką systemu a sferą jego zachowań dynamicznych. Osobno przedstawia dane niezbędne informatykowi w procesach przetwarzania, osobno zaś ich przepływy w systemie. Jest to sprzeczne ze sposobem postrzegania systemu przez bank, który w miejsce tabel bazy danych widzi funkcje, których spełnienia oczekuje. Poprawność rozwiązań strukturalnych jest trudna do weryfikacji, sprowadzając się często do próby wmówienia klientowi, że to, co widzi w analizie, jest właśnie tym, czego potrzebuje.

Jak zatem pogodzić owe sprzeczności? Jako antidotum proponuję zastosowanie notacji bazującej na Unified Modeling Language, stanowiącej zdecydowany zwrot w stronę klienta. Celem niniejszej analizy jest stworzenie materiału koncentrującego się na procesach, a nie na strukturach danych, którego poprawność będzie łatwa do sprawdzenia w banku. Prawidłowe podejście do analizy sytemu bankowego wymaga stworzenia dokumentu czytelnego dla klienta, a nie tylko i wyłącznie dla wykwalifikowanego programisty. Tak prowadzona analiza staje się doskonałym materiałem dla testów akceptacyjnych systemu, stanowiąc jednocześnie czytelną formę dokumentacji użytkowej. Dopiero na podstawie takiego materiału, w zaciszu

* Dr, adiunkt, Zakład Informatyki Ekonomicznej, Uniwersytet Łódzki.

macierzystej firmy analityk tworzy specyfikację systemową, opisując klasy, atrybuty czy zachowania związków ternarnych[1].

## Jakość rozwiązania

Aby docenić ogromne korzyści płynące z zastosowania nowoczesnych metod wykorzystania kapitału informacyjnego przedsiębiorstwa, należy proces tworzenia aplikacji potraktować bardzo poważnie. Tylko rzetelnie przeprowadzona analiza wymagań klienta zaowocuje w efekcie spójnymi wynikami, mającymi przełożenie na wysokiej jakości system bankowy. Problem jakości rozwiązania można opisać grupą sześciu cech charakteryzujących jakość systemu bankowego[2]. Są to:

- orientacja na klienta,
- elastyczne rozwiązania projektowe – definiowalność,
- przetwarzanie scentralizowane,
- spójność i kompleksowość rozwiązań,
- szybki dostęp do informacji o pozycji klienta i banku[3],
- obsługa wielu walut.

Stopień spełnienia przedstawionych wymagań daje podstawę do weryfikacji jakości rozwiązania. W jakim stopniu zapewnia ją Unified Modeling Language? Uważam, że w dużym. Nowa jakość systemu bankowego wynika z zagwarantowania powyższych elementów. Można zauważyć, że cechy takie jak obsługa wielowalutowa czy przetwarzanie scentralizowane związane są z fazą strategiczną systemu. Elementy te nie powinny „zaskoczyć" analityka na etapie procesu analizy. Jakkolwiek ich istnienie musi być podkreślone

---

[1] Znajomość tego typu elementów jest zbędna dla banku. Są one tworzone na potrzeby producenta oprogramowania, ale nie nadają się do przedstawienia klientowi celem weryfikacji.

[2] Rozwinięcie tematu zaprezentowano w rozprawie doktorskiej *Ocena możliwości aplikacyjnych języka UML w analizie bankowych systemów informatycznych* autorstwa Konrada Woźniackiego.

[3] Informacja taka związana jest z szacowaniem współczynnika zaangażowania klienta w fundusz własny banku. Ustalenie pozycji finansowej banku jest niezbędne dla realizacji transakcji dealerskich oraz podejmowania decyzji na międzynarodowych rynkach pieniężnych. Zapewnia to integracja systemu oraz wykorzystanie zaawansowanych technologii przetwarzania, umożliwiających bieżący przepływ danych z transakcji krajowych i zagranicznych. Niejednokrotnie wykorzystuje się tutaj mechanizm szacowania danych bieżących, wykorzystując tylko pozycje powyżej pewnej kwoty, zaś dopiero na koniec dnia licząc precyzję szacunku w stosunku do danych ostatecznych. Patrząc na powyższe dane bankowe przez pryzmat czasu, możemy nazwać je informacjami bieżącymi. Są one wykorzystywane głównie do zapewnienia sprawniejszego funkcjonowania banku i wynikają z jego działań operacyjnych.

w tworzonym opracowaniu, to jednak, moim zdaniem, jest to osiągalne również w podejściu strukturalnym. O wiele gorzej sytuacja wygląda w stosunku do pozostałych cech z zaprezentowanej grupy.

### Orientacja na klienta

Orientacja na klienta jest podstawowym założeniem przyświecającym tworzeniu nowoczesnego systemu bankowego. Zaletą UML jest zwrócenie uwagi na problem specyfikacji procesów zachodzących wewnątrz instytucji, dzięki czemu weryfikacja materiału nie wymaga głębokiej wiedzy informatycznej. Co więcej, owa weryfikacja powinna być wykonywana przez specjalistę z dziedziny bankowości, a nie zakładowego informatyka. Patrząc na cały proces z pewnej perspektywy, widoczna jest następująca zależność: bank zmienia formy swojej organizacji celem zadowolenia swoich klientów, dokonując parametryzacji swoich działań. Analogiczna sytuacja powinna mieć miejsce po drugiej stronie. Dla firmy software'owej to bank jest klientem i to jego oczekiwania powinny być spełniane[4]. Wymagania te przedstawiane są na spotkaniach i zostają przeniesione na diagramy przypadków użycia. Moje doświadczenia wykazały, że zastosowanie tych diagramów spotyka się z przychylnością ze strony klienta, wydatnie podnosząc jakość procesu analizy. Oto kilka przykładów:

1. Bank jest zainteresowany wykorzystaniem *use-case*, gdyż diagram ten w prostej graficznej formie specyfikuje żądane przez użytkownika funkcje aplikacji. Ukazuje osoby, które będą mogły z niego korzystać, jak również daje szanse na umieszczenie funkcjonalności, których przyszłą implementację bank ma na uwadze.
2. Zaangażowanie banku w proces analizy systemu wnosi nową jakość wynikającą z zaangażowania nie tylko jednostek decyzyjnych (kadra kierownicza), ale przede wszystkim szeregowych pracowników. Z uwagi na fakt, że to oni będą pracowali z tworzoną aplikacją, stają się źródłem cennych uwag dotyczących funkcjonalności rozwiązania.
3. Graficzna reprezentacja zastosowana na diagramie *use-case* przełamuje barierę wiedzy technicznej istniejącą między pracownikami banku a informatykami.
4. Opisy zamieszczone na diagramach wprowadzane są w języku użytkownika, dzięki czemu są dla niego bardziej czytelne i zrozumiałe. Bankowcy nie dają sobie rady z notacją techniczną, którą posługują się programiści.

---

[4] Oczywiście zakres wymagań musi pozostawać w realnych granicach wykreślonych umową kontraktową i możliwościami technicznymi sprzętu oraz aplikacji.

5. Dla analityka i projektanta model przypadków użycia jest wyznacznikiem dalszego postępowania warunkującego listę funkcjonalności systemu. Obrany kierunek wskazuje, jakie diagramy czynności musi zbudować analityk oraz które z funkcji elementarnych ma przedstawić do zakodowania programiście.

6. Zespoły integratorów i testerów określają wykorzystanie diagramów przypadków użycia za celowe, podnoszące jakość procesu testowania. Będąc jasną i czytelną formą dokumentacji – stanowią, czy system w prawidłowy sposób spełnia wymagania stawiane przez użytkownika.

7. Również inne osoby zaangażowane w procesach biznesowych wyniosą wymierne korzyści z lektury diagramu *use-case*. Do tej grupy zaliczyć można dział marketingu bankowego, sprzedaży produktów, dział obsługi klienta oraz departamenty odpowiedzialne za gromadzenie różnego rodzaju dokumentacji bankowej. Wszyscy oni będą chcieli „przemycić" do systemu pewne rozwiązania ułatwiające im życie, o których zapewne nie pomyśli programista korzystając z Data Flow Diagram na poziomie kontekstowym w podejściu strukturalnym.

Powyższe elementy wykazują wyższość zastosowania diagramów przypadków użycia w stosunku do strukturalnego diagramu kontekstowego. Również późniejsze działanie, polegające na wykorzystaniu diagramów czynności do rozpisania procesów zachodzących w banku, wzmacnia ten pozytywny wymiar. Dzieje się tak za sprawą przeniesienia powiązań realizowanych przez system funkcji z aktorami, co jest nowością w stosunku do metod strukturalnych. Dotychczasowe podejście klasyczne koncentrowało się na systemie, jego otoczeniu i przepływach danych[5]. UML skupia swoją uwagę na obsłudze procesów biznesowych, dla których system jest tylko platformą realizacji.

## Elastyczność rozwiazania

Wspomniane powyżej zastąpienie diagramu kontekstowego i DFD zespołem diagramów użycia i czynności owocuje kolejnym elementem jakości – zachowaniem elastyczności rozwiązań. Tworząc strukturalny diagram bąbli należy, już na starcie, znać pełną listę procesów zachodzących w systemie. Związane to jest z hierarchiczną strukturą zrównoważonych diagramów przepływu danych. Jeśli na jakimś poziomie analizy krystalizuje się nowy proces, pojawia się konieczność zmiany lub stworzenia odpowiadających mu nadrzędnych DFD. W rzeczywistości nakłady czasu i pracy poniesione na

[5] Grupa tych elementów opisywana jest poprzez zestawienie diagramu kontekstowego ze zrównoważonymi diagramami przepływu danych.

zmiany diagramów są wysokie, co niejednokrotnie owocuje odejściem od koncepcji zaproponowanych zmian[6]. Dzieje się to, niestety, kosztem jakości końcowego rozwiązania.

Celem osiągnięcia wysokiej jakości systemu, analityk musi mieć na uwadze pewne zagadnienia podstawowe, takie jak:

- wydajność,
- niezawodność,
- pielęgnowalność systemu.

Elementy te są ściśle związane z definiowalnością i elastycznością, co argumentuję tym, że system wydajny musi być systemem definiowalnym. Żadna szanująca się firma nie może sobie pozwolić na stworzenie aplikacji sprawdzającej się tylko „tu i teraz", czyli w warunkach zachodzących w czasie analizy[7]. Przy ogromnej dynamice zmian procesów i produktów bankowych, system taki przestanie spełniać swoje zadania w przeciągu roku, a więc prawdopodobnie jeszcze przed zakończeniem fazy wdrożenia. Stanie się tak dlatego, że aktualne sposoby funkcjonowania produktów bankowych nie będą możliwe do obsłużenia przy użyciu stworzonych „na sztywno" elementów. Podejście takie zaowocuje kolejnym problemem – niską niezawodnością systemu. Obsługa produktu za pomocą niedefiniowalnych, statycznych modułów wymusi tworzenie pewnych o wiele bardziej zawodnych rozwiązań tymczasowych[8]. Z kolei brak elastyczności systemu wpłynie na możliwość jego przyszłej pielęgnacji. W przypadku aplikacji bankowych nie możemy dopuścić do zamrożenia obsługi funkcji systemu na jakimś poziomie, gdyż produkt bankowy żyje swoim własnym, coraz bardziej dynamicznym życiem. Aktualnie banki posiadają możliwość parametryzowania warunków udzielania kredytów i przyjmowania depozytów stosowanie do potrzeb różnych grup klientów. Takie

---

[6] Rezygnacja następuje często z obu stron. Zarówno analityk, jak i współpracujący z nim przedstawiciel banku nie są zainteresowani dodatkową pracą. W takiej sytuacji nowe, konieczne do obsługi procesy „podpinane" są do już istniejących grup w modelu systemu. Skutkiem takiej niefrasobliwości otrzymujemy późniejsze obarczanie specyfikacji procesów dużą grupą nadmiarowych danych, koniecznych do obsłużenia dopisanej na siłę funkcji systemu.

[7] Jest to jedna z zasad systemu pakietowego. System taki jest opracowaniem ramowym (szkieletowym/pakietowym), akceptowanym przez wiele banków. Zawiera skumulowaną i zweryfikowaną wiedzę projektantów oraz doświadczenie wdrożeniowe poprzednich użytkowników. Tym samym, jakość sytemu przy wdrażaniu u każdego kolejnego użytkownika powinna być lepsza.

[8] Zazwyczaj, kiedy z uwagi na zbyt sztywne określenie założeń systemu (często spotykane przy strukturalnym modelu kontekstowym) nie jesteśmy w stanie obsłużyć jakiegoś procesu bankowego, tworzona jest pewna forma obejścia problemu. Określana mianem łaty (*patch*) ma za zadanie naprawić uchybienia. Niestety, umieszczanie takich poprawek na już istniejącej strukturze aplikacji jest o wiele mniej bezpieczne i wydajne od rozwiązań zaprojektowanych na początku tworzenia sytemu.

podejście możliwe jest tylko przy zachowaniu definiowalności i elastyczności zaimplementowanych rozwiązań.

Oba te postulaty spełnia notacja bazująca na UML. Konieczność dodania nowej oczekiwanej funkcji w aplikacji realizowana jest zależnie od jej wagi. Elementy „uszczegółowiające" dopisywane są do drzewa hierarchii funkcji, a następnie umieszczane na konkretnym diagramie czynności. Jeśli mamy do czynienia z zupełnie nową funkcją systemu, to tworzymy dla niej nowy przypadek użycia. Z uwagi na fakt, że UML nie niesie ograniczeń co do liczby diagramów *use-case* (w przeciwieństwie do pojedynczego diagramu kontekstowego), nie spowoduje to znaczących poprawek w dotychczasowym opracowaniu. Dla nowo powstałego przypadku użycia dorysowuję opisujący go diagram czynności (ewentualnie również sekwencji), korzystając z wyszczególnionych dotychczas funkcji elementarnych. W przypadku braku odpowiednich elementów zawsze istnieje możliwość ich dopisania w formie nowej gałęzi drzewa hierarchii. Nie powoduje to oczywiście zaburzenia jego dotychczasowej struktury.

## Spójność i kompleksowość

Niezwykle ważna jest spójność i kompleksowość rozwiązań, gdzie w przypadku zastosowania UML bank „widzi, co dostaje", gdyż opracowanie analityczne przybiera czytelną i zrozumiałą formę. Dzięki temu Unified Modeling Language odwzorowuje potrzeby klienta a nie producenta oprogramowania, oferując niespotykaną łatwość weryfikacji wyników analizy. Graficzna forma jej reprezentacji ułatwia zrozumienie zapisu, a co za tym idzie ułatwia zrozumienie tematu oraz zwiększa efektywności procesu weryfikacji. Z uwagi na fakt, że jakość weryfikacji wpływa na spójność opracowania, wszelkie formy czytelniejsze dla klienta są jak najbardziej pożądane. W modelu strukturalnym mieliśmy do czynienia z oderwaniem statyki od dynamicznych zachowań systemu, co powodowało trudności w weryfikacji kompleksowości rozwiązania. Problemem było ustalenie, czy informacje zapisane na poszczególnych diagramach nie są wzajemnie sprzeczne, co wynikało z braku bezpośredniego przełożenia pomiędzy diagramami. Dodatkowo następowało rozerwanie między przepływem danych (obrazowanym DFD) a zachodzącymi w systemie procesami biznesowymi. Diagram specyfikacji procesów ma bowiem za zadanie opisanie procesu na najniższym poziomie diagramu przepływu danych, co nie jest równoznaczne z opisem rzeczywistego bankowego procesu biznesowego.

## Dostęp do informacji

Specyfiką systemów bankowych jest konieczność szybkiego dostępu do informacji. Mam tu na myśli zarówno dane o pozycji klienta, jak i informacje dotyczące aktualnej pozycji banku. Pozornie sprawa ta zależy od zaprojektowania struktur bazy danych, jednak mówiąc o informacji pojawia się pytanie, jakiego rodzaju dane mają być udostępniane. W tym miejscu należy cofnąć się do fazy analizy. Informacja to dane przechowywane w systemie, lecz błędem jest wyjście od ich struktury statycznej. Ich dostępność wynika z przebiegu procesów biznesowych oraz procesów wewnętrznych organizacji, generujących informacje niezbędne dla prawidłowej działalności operacyjnej banku. Rozpoczęcie projektowania sposobów pobierania danych od tworzenia ich struktur, skutkuje w większości przypadków pojawieniem się luk informacyjnych. Takie podejście jest tożsame z analizą skutków, a nie przyczyn. Jako przykładem posłużę się podsystemem SWIFT-owym, gdzie dla prawidłowego zarządzania depeszami w operacjach zagranicznych korzystne jest automatyczne ściąganie kursów walut z monitoringu Reutersa oraz informacji ze swiftowej bazy banków. Pierwsze z nich ładowane są do tablic kursów wymiany, drugie zaś zapisywane do kartoteki podmiotów finansowych w projektowanej aplikacji. W przypadku wykorzystania podejścia strukturalnego nie uchwycimy momentu pobierania tej informacji, określanego procesem biznesowym, a nie strukturą danych. Co za tym idzie, w najlepszym wypadku otrzymamy strukturę bazodanową pozbawioną zasilającego ją przypadku użycia. Wówczas z czynności o przebiegu automatycznym przejdziemy na proces ręczny, nie wnoszący żadnego usprawnienia.

Powyższe problemy wykorzystania podejścia strukturalnego związane są ze sposobem uzyskiwania danych zapisanych w architekturze systemu komputerowego. Uzyskiwanie informacji o pozycji finansowej banku leży w gestii zaplecza operacyjnego kierownictwa. Otrzymanie wsparcia informacyjnego w trybie *on-line* wymaga integracji danych pochodzących z wielu procesów, a nie jedynie ze struktur hurtowni danych, które tworzone są na zamknięcie dnia. Jak zatem widać, wykorzystanie Unified Modeling Language ma duży wpływ na kolejną cechę jakości systemu bankowego, skutkując dostarczaniem w czasie rzeczywistym informacji o pozycji finansowej klienta i banku.

Zalety UML w stosunku do metod strukturalnych w fazie analizy systemów bankowych zostały zebrane w tab. 1.

Tabela 1

Zestawienie cech notacji strukturalnej w konfrontacji z cechami notacji bazującej na UML

| ELEMENT | PODEJŚCIE STRUKTURALNE | NOTACJA BAZUJĄCA NA UML |
|---|---|---|
| Spojrzenie na system bankowy | Dysonans między statyką systemu a jego dynamicznym zachowaniem, zwiększający się wraz ze wzrostem komplikacji systemów bankowych | Zgodnie z zasadami modeli obiektowych statyka i dynamika systemu są ściśle ze sobą związane |
| Punkt wyjścia notacji | Orientacja na struktury danych | Orientacja na procesy w organizacji |
| Widok ogólny systemu | Wymiarowany pojedynczym diagramem kontekstowym | Szerokie spektrum widzenia systemu obrazowane szeregiem diagramów przypadków użycia |
| Modelowanie dynamiki systemu | Jednowymiarowe, poprzez Data Flow Diagram | Wielowymiarowe, przez zastosowanie połączenia *use - case* z diagramami czynności i sekwencji |
| Opis czynności elementarnych | Rozumiane jako diagram specyfikacji procesów i opisane językiem naturalnym lub innymi elementami notacji | Opisane drzewem hierarchii funkcji oraz diagramami funkcji elementarnych, niosącymi dodatkową informację na temat sposobu zasilenia funkcji i wymagalności danych |
| Modelowanie statyki sytemu | W podejściu strukturalnym ERD jest wykorzystywany prawie wyłącznie do modelowania danych trwałych, co jest sprzeczne z dynamiką systemów bankowych | Dzięki elastyczności pojęcia klasy istnieje możliwość modelowania struktur dynamicznych |
| Modelowanie procesów biznesowych zachodzących w banku | Spojrzenie strukturalne jest zbyt wąskie dla odwzorowania całości procesów biznesowych realizowanych w banku | Szerokie spojrzenie UML daje zarówno możliwość odwzorowania procesów niezbędnych dla producenta oprogramowania, jak i opisuje model banku jako przedsiębiorstwa. Jest to bardzo cenny wkład analizy w zarządzanie strukturami i obiegiem informacji w banku |
| Dokumentowanie prac | Dokumentacja powstająca osobno i tworzona zazwyczaj w języku naturalnym plus słowniki danych | Zastosowanie notacji opartej na UML jest dobrą dokumentacją wykorzystywaną zarówno na etapie testów, jak i przez odbiorcę oprogramowania w fazie eksploatacji |
| Spójność rozwiązania | Trudna do weryfikacji poprzez słabo widoczne związki między diagramami | Przez swą czytelność umożliwia wychwycenie sytuacji błędnych nie tylko w projektowanej aplikacji, ale i w organizacji przetwarzania i obiegu informacji w banku |

Tabela 1 cd.

| ELEMENT | PODEJŚCIE STRUKTURALNE | NOTACJA BAZUJĄCA NA UML |
|---|---|---|
| Specyfikacja stanowisk pracy | W notacji strukturalnej nie ma bezpośredniego przełożenia na podmioty odpowiadające za realizację procesu. Zazwyczaj stosowny opis jest rozwinięciem specyfikacji procesu i zapisu o ręcznym lub automatycznym sposobie przetwarzania | Dzięki czytelnej specyfikacji aktorów istnieje możliwość przyporządkowania ról dla poszczególnych użytkowników systemu (dostępność do poszczególnych funkcji) |
| Łatwość weryfikacji opracowania | Weryfikacja utrudniona z uwagi na niską czytelność notacji strukturalnej dla pracowników banku | W czasie weryfikacji efektów analizy klient koncentruje się na tym, na czym się zna, a więc na realizacji procesów w banku. Nie musi przyswajać i weryfikować zbędnych dla niego informacji zapisanych w obcym mu języku |
| Kompletność notacji | Notacja tworzona z myślą o twórcy systemu przedstawia tylko stronę informatyczną projektu (bez aspektów „biznesowych"). Dodatkowo niektóre elementy zapisu strukturalnego, np. ERD, są zbyt trudne dla specjalisty z banku | Notacja zawiera zarówno wszystkie elementy niezbędne dla programisty, jak i dla użytkownika |
| Rozmiary opracowania analitycznego | Tworzone opracowanie przybiera formę wielostronicowego dokumentu, w którym trudno odnaleźć szukane informacje i wprowadzić ewentualne poprawki. Wynika to z mnogości zapisów opartych na formach języka naturalnego | Elementy notacji strukturalnej o największej objętości zastąpione zostały serią diagramów czynności i funkcji elementarnych. Taka forma umożliwia łatwe poprawianie i szybki dostęp do informacji |
| Obszar modelowania | Modele charakteryzują wnętrze systemu z niskim poziomem opisu otoczenia ograniczonym do prostego diagramu kontekstowego | Notacja UML ukazuje nie tylko wewnętrzną strukturę banku, ale również dynamikę zmian w instytucji i jej otoczeniu (wartość dostarczana przez procesy) |
| Poprawność semantyczna | Poprawność semantyczna rozwiązania zależy w głównej mierze od doświadczenia zespołu analitycznego – jest wysoce subiektywna | Poprawność semantyczna uzyskiwana jest dzięki wykorzystaniu metamodeli uniwersalnego języka modelowania |
| Nakłady pracy ponoszone przy projekcie | Mniejsze zaangażowanie klienta na wstępnych etapach analizy skutkujące dużym obciążeniem banku podczas weryfikacji opracowania | Duże zaangażowanie banku w cały proces analizy skutkujące częściowym przeniesieniem nakładów pracy na firmę realizującą projekt |

Tabela 1 cd.

| ELEMENT | PODEJŚCIE STRUKTURALNE | NOTACJA BAZUJĄCA NA UML |
|---|---|---|
| Możliwość zastosowania podejścia w stosunku do rozmiarów systemu | Bez ograniczeń – podejście dobre zarówno dla systemów małej, jak i dużej wielkości. Jedyny minus to zdecydowany rozrost objętości dokumentacji w miarę wzrostu rozmiarów systemu | Dopiero na etapie UML uniknięto zarzutu zbyt małej przydatności notacji obiektowej dla dużych systemów bankowych (wada niskiej skalowalności rozwiązań od czasów OOA – OOD)* |
| Możliwość rozwoju w kierunku samodzielnych metod definiowania założeń systemu przez klienta | Sposób notacji strukturalnej w znacznej mierze ogranicza możliwość samodzielnej pracy klienta z aplikacją, na podstawie której miałyby powstać założenia systemu | Możliwe przyszłe zastosowania definiowania wymagań klienta poprzez aplikacje portali WEB-owych |
| Generalne ujęcie problemu | Typowo systemowe, zorientowane na informacje niezbędne producentowi systemu bankowego | Zgodne z przyzwyczajeniami odbiorcy, co wynika z orientacji na procesy biznesowe instytucji |

* Notacja strukturalna dla dużego systemu bankowego osiągała tak gigantyczne rozmiary, że jej aktualizacja i utrzymanie wewnętrznej spójności graniczyły z cudem

Źródło: opracowanie własne.

## Podsumowanie

Uwypuklone znaczenie procesów biznesowych zachodzących w banku znajduje oddźwięk w orientacji Unified Modeling Language nakierowanej na użytkownika. Szerokie uczestnictwo odbiorców aplikacji w procesie analizy i projektowania umożliwia dobre poznanie systemu, powodując, że nie jawi się on jako „narzucony z zewnątrz"[9]. Samodzielne decyzje użytkownika w kluczowych kwestiach analizy skutkują przychylną atmosferą i pozytywnym nastawieniem odbiorcy do projektu. Jest to jedna z wielkich zalet notacji wykonanej na podstawie UML. W tabeli 2 ukazuję różnice między tworzeniem systemu opartego na podejściu „tradycyjnym" a projektowaniem przy współudziale użytkowników.

[9] Znana jest prawda, że najlepsze programy to te, które sami napisaliśmy. Jeżeli użytkownik systemu poczuje się autorem programu, to szkolenie wdrożeniowe staje się zbędne, zaś utrzymanie programu w stanie aktualnym o wiele łatwiejsze.

Tabela 2

Porównanie projektowania tradycyjnego z partycypacyjnym

| PROJEKTOWANIE TRADYCYJNE | PROJEKTOWANIE PARTYCYPACYJNE |
|---|---|
| Tempo realizowania prac projektowych wynika z rutynowych przyzwyczajeń zespołu informatyków | Zaangażowanie użytkowników jest czynnikiem motywującym informatyków, przyczynia się do wzrostu wydajności pracy |
| Wyniki projektowania zależą głównie od umiejętności projektantów systemów | Odpowiedzialni za wyniki są ludzie najlepiej znający problem |
| Użytkownikom „sprzedaje się" system wraz z jego efektami | Użytkownicy biorą na siebie realizację efektów |

Źródło: [4].

Dodatkową zaletą partycypacji użytkowników na etapie analizy systemu jest uwzględnienie perspektywy rozwojowej aplikacji w przyszłych okresach czasu. Jedynie użytkownik jest bowiem w stanie stwierdzić nie tylko to, co potrzebuje dziś, ale również to, co prawdopodobnie będzie mu potrzebne jutro.

Jak widać, wykorzystanie Unified Modeling Language w przypadku analizy systemu bankowego jest przedsięwzięciem złożonym i wymagającym wiedzy i doświadczenia. Prawidłowość wykonania analizy w znacznej mierze wpływa zarówno na koszty projektu informatycznego, jak i na samo jego powodzenie. Popełnione na tym etapie błędy przenoszone są dalej w „głąb" projektu, ujawniając się na prawie każdym etapie cyklu życia systemu. Tym samym wykorzystanie rozwiązań najlepszych z możliwych jest jak najbardziej wskazane. Moim zdaniem rozwiązaniem tym jest UML.

## Literatura


[1] Baker R., *CASE method. Modelowanie związków encji*, WNT, Warszawa 1996.

[2] De Marco T., *Structured analysis and System Specification*, Yourdon Press 1978.

[3] Hawryszkiewycz I. T., *Introduction to Systems Analysis and Design*, Prentice Hall, 1994.

[4] Niedźwiedziński M., *Ocena zamierzeń informatyzacyjnych przedsiębiorstwa*, Wyd. UŁ, Łódź 1989.

[5] Roszkowski J., *Analiza i projektowanie strukturalne*, Helion, Gliwice 1998.

[6] Ryznar Z., *Informatyka bankowa – próba syntezy*, Wyd. Wyższej Szkoły Bankowej, Poznań 1998.

[7] Szyjewski Z., *Metoda spiralna tworzenia oprogramowania*, Infogryf, KNJORK, Szczecin 1986.

[8] Woźniacki K., *Ocena możliwości aplikacyjnych języka UML w analizie bankowych systemów informatycznych*, rozprawa doktorska, Uniw. Łódzki, Łódź 2002.

[9] Yourdon E., *Współczesna analiza strukturalna*, WNT, Warszawa 1996.

*Konrad Woźniacki*

**Actual function of the software engineering in the modern IT environment**

**Summary**

The paper presents considerations on the subject of analytical solution quality referring to the requirements of modern banking system. It explores key advantages of using UML in phase of analyzing bank's system lifecycle in comparison with sample structural methods.