Dariusz OLCZYK[1]

MODELOWANIE STRUKTURALNE – DEFINICJE, NOTACJA, TECHNIKI I NARZĘDZIA

Streszczenie

W artykule wyróżnione zostały dwa zasadnicze podejścia do analizy i modelowania wykorzystywane obecnie w procesie budowy systemów informatycznych: strukturalne oraz obiektowe. Myśl przewodnia to próba odpowiedzi na pytania dotyczące istoty stosowania podejścia strukturalnego, wyjaśnienia podstawowych pojęć stosowanych w praktyce oraz przybliżenia notacji wbudowanych w różnego rodzaju narzędzia wspomagające proces projektowania systemów. Zasadnicza część artykułu zawiera szczegółowe informacje nt. sposobów tworzenia diagramów FHD, DFD oraz ERD.

Abstract

The article highlights two basic approaches to analysis and modeling currently used in the information systems development: structural and object-oriented. The keynote is an attempt to answer questions about the substance of the structural approach and to clarify key concepts applied in practice as well as to familiarize with notations embedded in different types of tools supporting the process of system design. The main part of the article provides detailed information about the ways of FHD, ERD and DFD diagrams creation.

1. WPROWADZENIE

Bezsprzecznie budowa nowych lub modernizacja istniejących systemów informatycznych to nadal trzon działalności informatycznej w kraju i na świecie. Ciągły postęp cywilizacyjny, a co za tym idzie rozwój wszelakich form aktywności człowieka wykorzystującego osiągnięcia świata komputerów, pociąga za sobą potrzebę tworzenia coraz to nowocześniejszych i coraz bardziej złożonych systemów informatycznych. Tym samym przed zespołami IT zajmującymi się wytwarzaniem oprogramowania stawiane są z roku na rok coraz trudniejsze zadania. Wyzwanie jakim jest budowa w pełni funkcjonalnego i akceptowalnego przez odbiorcę biznesowego systemu w określonych reżimach czasowych oraz kosztowych wymaga od specjalistów

[1] Mgr inż. Dariusz Olczyk jest wykładowcą w Warszawskiej Wyższej Szkole Informatyki

IT przede wszystkim metodycznego podejścia do procesu wytwarzania oprogramowania, a w szczególności przeprowadzenia rzetelnej analizy systemu informacyjnego organizacji. Gwałtowny rozwój takiej dziedziny informatycznej jaką jest inżynieria oprogramowania w pełni potwierdza powyższą tezę.

Pamiętajmy, iż w procesie analizy mamy do osiągnięcia zasadniczo dwa główne cele: po pierwsze chcemy stworzyć zbiór wymagań użytkownika, czyli odpowiedzieć na pytanie co klient potrzebuje, po drugie – zbudować logiczny model systemu, który stanie się osią spinającą wszystkie kolejne elementy projektu.

Gdyby spojrzeć na zagadnienie modelowania systemów informatycznych poprzez pryzmat historii rozwoju branży informatycznej na przestrzeni ostatnich 40 lat to wyraźnie widać kolejne fazy, przez które to modelowanie przeszło. Lata 70-te to w zasadzie dominacja sprzętu, gdzie oprogramowanie stanowi wyłącznie dodatek. Kolejne lata przynoszą stopniowe acz wyraźne odwrócenie tego stanu na korzyść oprogramowania. Przełom lat 70-tych i 80-tych oraz związany z nim kryzys oprogramowania spowodował narodziny wspomnianej już wcześniej inżynierii oprogramowania. W tym też czasie obserwujemy pojawienie się podejścia strukturalnego stanowiącego pierwsze prawdziwie inżynierskie podejście do modelowania informacji. Z kolei lata 90-te to dynamiczny rozwój podejścia obiektowego spowodowanego głównie popularnością pierwszego istotnego dla przemysłu języka obiektowego jakim był C++.

Śmiało można zaryzykować twierdzenie, iż informatycy, a w szczególności projektanci/analitycy systemów informatycznych to jedna z nielicznych grup zawodowych, zapewne tuż obok filozofów, zastanawiająca się nad złożonością otaczającego nas świata. Zapewne właśnie ta różnorodność rzeczywistości z jaką styka się informatyk stanowi o atrakcyjności tego zawodu z drugiej jednak strony jest podstawowym i zdecydowanie niebanalnym problemem do rozwiązania w trakcie projektu. Modelowanie, które w swojej istocie ma ten świat zewnętrzny opisać, uporządkować i wyjaśnić reguły jego działania jest tworzeniem swego rodzaju mapy drogowej dla projektu. Tworząc tą mapę szukamy rzeczy ważnych, a pomijamy nieistotne, generalizujemy pewne informacje, ale przede wszystkim identyfikujemy obiekty i szukamy zależności między nimi. Trzeba pamiętać, że tak stworzony obraz świata zewnętrznego musi być zrozumiały dla wszystkich interesariuszy projektu, zarówno przedstawicieli biznesu jak i obszaru IT. Znaczenie modelowania w cyklu projektowym cały czas wyraźnie rośnie, a wraz z pojawieniem się narzędzi klasy CASE nastąpiło wyraźne przesunięcie ciężaru na elementy tworzenia opisu operującego

pojęciami bliskimi „zwykłym ludziom" w stosunku do tradycyjnego kodowania. Na tym gruncie popularność zdobyło podejście obiektowe. Tutaj bowiem widzimy bezpośrednie korelacje między elementami tworzonych diagramów a elementami ze świata rzeczywistego.

Pytanie jakie stawiamy sobie na wstępie niniejszego wykładu to pytanie o zasadność stosowania podejścia strukturalnego w nowych projektach. Czy ten sposób modelowania nie powinien już odejść do lamusa, czy nadal są obszary zastosowań dla których to podejście możne być najlepsze lub chociaż nie gorsze?

2. PODEJŚCIE STRUKTURALNE vs. OBIEKTOWE

Podejście strukturalne

Cechą charakterystyczną podejścia strukturalnego jest traktowanie świata zewnętrznego jako istoty dualnej składającej się z dwu bytów:

- danych,
- procesów przetwarzania.

Klasyczna analiza strukturalna zaproponowana przez Toma DeMarco (dzisiaj już rzadko stosowana w praktyce) proponowała kaskadowy, zorientowany na kolejne fazy model budowy systemu informatycznego. Na gruncie krytyki klasycznego podejścia DeMarco powstały nowoczesne metodyki strukturalne, np. nowoczesna analiza strukturalna Yourdona (Modern Structured Analysis) czy metoda SSADM (Structural System Analysis and design Method). Zrezygnowano w nich z modelu kaskadowego na rzecz modeli umożliwiających szybką analizę i budowę oprogramowania, przede wszystkim modelu przyrostowego. Istotną cechą modelowania strukturalnego nadal pozostaje rozdzielenie pojęcia danych i funkcji, pomimo faktu iż nowoczesna analiza strukturalna zbliżyła oba modele wprowadzając mechanizmy zdarzeń i transakcji, które starały się łączyć strumienie danych i procesy.

Jeśli zatem chcemy wymienić podstawowe cechy modelowania strukturalnego to będą to na pewno:

- orientacja na funkcje i procesy,
- orientacja na dane,
- analiza zstępująca „z góry na dół" (top-down),
- priorytet analizy logicznej,
- odrębność modelowania danych i procesów,
- ustrukturalizowane narzędzia i techniki.

Głównym celem analizy strukturalnej systemów informatycznych jest stworzenie specyfikacji funkcjonalnych systemu, które będą:

- graficzne – diagramy uzupełnione o szczegółowy materiał tekstowy,

- podzielone – poszczególne części specyfikacji można czytać i analizować niezależnie od innych,
- minimalnie nadmiarowe – zmiany w wymaganiach użytkownika powinny powodować zmiany tylko w jednej części specyfikacji.

Analiza strukturalna koncentruje się na stworzeniu dwóch modeli dziedziny problemu:

- modelu danych – część pasywna systemu,
- modelu funkcji – aktywna część modelu.

Wynika z tego faktu proste ale i niezwykle istotne spostrzeżenie, że metody strukturalne mogą być szczególnie przydatne w przypadkach kiedy jeden z aspektów systemu – pasywny (dane) lub aktywny (funkcje) jest zdecydowanie dominujący. A zatem jeśli chcemy zbudować system informatyczny obsługujący przechowywanie i wyszukiwanie złożonych danych, a nie realizujący złożone funkcje (dominacja modelu danych) lub system realizujący złożone funkcje związane np. z obliczeniami naukowymi na prostych danych (dominacja modelu funkcjonalnego) wybór podejścia strukturalnego będzie zdecydowanie właściwy.

Podejście obiektowe

Podejście obiektowe w modelowaniu systemów powstało w połowie lat 80-tych w wyniku gwałtownego rozwoju i wzrostu popularności obiektowych języków programowania (Java, VisualBasic, C+). Nowe koncepcje, które wówczas zaproponowano, stały się głównym trendem w latach następnych w czasie powstawania zarówno nowych języków programowania obiektowego jak i specjalizowanych języków analizy projektowania obiektowego. Myśl o połączeniu dwu modeli: danych i funkcji, których niezależność stanowiła istotną wadę analizy strukturalnej, jest najbardziej widoczną zmianą, jaka przyniosła analiza obiektowa. Najważniejsze zalety nowego podejścia to pojęcie klasy i obiektu łączącego w sobie z definicji dane i operacje oraz naturalna orientacja na przyrostowy model rozwoju oprogramowania.

Tym samym łatwo jest wymienić podstawowe cechy modelowania obiektowego:

- integracja modelu danych i procesów,
- powiązanie danych i funkcji w ramach obiektów,
- hermetyzacja (inkapsulacja) – zmiana dotyczy tylko danego (jednego) obiektu,
- grupowanie obiektów w klasy,
- dziedziczenie danych i procedur w ramach hierarchii klas,
- komunikacja miedzy obiektami za pomocą komunikatów.

W analizie obiektowej podstawową strukturą modelowania jest obiekt (klasa), a system jest przedstawiany jako kolekcja obiektów o określonych cechach realizujących

zadania z wykorzystaniem metod (funkcji). Dynamiczna strona systemu opisana jest przez model interakcji pomiędzy obiektami, które zachodzą w wyniku wysyłania i odbierania przez obiekty komunikatów.

W podsumowaniu rozważań na temat podobieństw i różnic obu podejść poniżej zaprezentowane zostało zestawienie najistotniejszych cech.

PODEJŚCIE STRUKTURALNE

- dane i procesy modelowane osobno,
- wykorzystanie w zasadzie tylko prostych typów danych,
- dobrze dostosowane do relacyjnego modelu danych,
- podstawowe techniki
  - diagramy związków encji (ERD),
  - hierarchie funkcji (FHD),
  - diagramy przepływu danych (DFD).

PODEJŚCIE OBIEKTOWE

- dane i procesy modelowane łącznie,
- wykorzystują złożone typy danych,
- dostosowane do obiektowego modelu danych,
- podstawowe techniki:
  - diagramy klas UML,
  - przypadki użycia i modele dynamiczne UML

3. DIAGRAMY W ANALIZIE STRUKTURALNEJ

W przypadku realizacji zadania projektowego z wykorzystaniem modelowania strukturalnego wykonywane są zazwyczaj trzy podstawowe diagramy w następującej kolejności:

- Diagram Hierarchii Funkcji (Function Hierarchy Diagram) jako reprezentacja wymagań klienta w zakresie realizacji przez system określonych operacji,
- Diagram Związków Encji (Entity Relationship Diagram) jako reprezentacja danych jakie muszą być przez system przetwarzane,
- Diagram Przepływu Danych (Data Flow Diagram) jako element spinający część statyczną (dane) i dynamiczną (funkcje).

Poniżej przedstawiona zostanie charakterystyka ww. diagramów a w tym zasady tworzenia, notacja, przykłady oraz wskazówki dla osób wykonujących modele z wykorzystaniem podejścia strukturalnego.

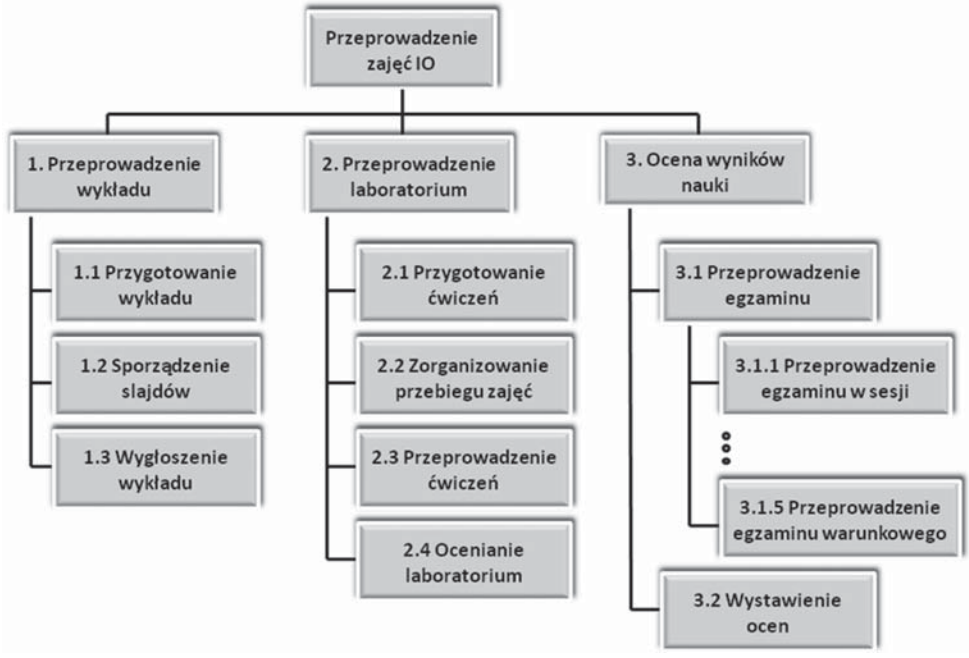
Diagram Hierarchii Funkcji

Diagram ten jest narzędziem modelowania, pozwalającym opisać w postaci modelu pojęciowego (konceptualnego) czym zajmuje się organizacja, dla której chcemy stworzyć system informatyczny. Jest to pierwszy etap w procesie określania wymagań funkcjonalnych systemu.

Tworząc FHD patrzymy na dziedzinę problemu z perspektywy funkcji, które są realizowane w kontekście celów, które firma chce osiągnąć.

W utworzonej hierarchii, funkcje wyższego poziomu opisują istotę działania firmy, natomiast dekompozycja w dół (rozkład funkcji na bardziej elementarne) pokazuje jak podfunkcje przyczyniają się do realizacji funkcji nadrzędnych. Podstawą dobrego modelu firmy jest odwzorowanie w nim wszystkich potrzeb funkcjonalnych przedsiębiorstwa, niezależnie czy mają one związek z informatyzacją czy też nie.

Poniżej pokazany jest przykład prostego diagramu FHD opisującego funkcjonalność związaną z przeprowadzeniem zajęć z przedmiotu Inżynieria Oprogramowania. Przy tworzeniu tego typu diagramu należy pamiętać, że w nazwach funkcji powinny znaleźć się pojęcia (terminy), które są używane w organizacji. Ponadto opis powinien być precyzyjny tzn. określać dokładnie to, co dana funkcja realizuje. Dobrą praktyką jest rozpoczynanie nazwy funkcji od czasownika. Etykieta funkcji jest jej nazwą skróconą stosowaną dla wygody. Umożliwia ona szybkie odwoływanie się do danej funkcji podczas analizy. Często jest ona liczbą określającą pozycje funkcji w hierarchii. Już teraz można także wspomnieć, iż te same funkcje wymienione w diagramie FHD pojawia się również w diagramie przepływu danych. Oznaczenia liczbowe tutaj naniesione okażą się wtedy nieoceniona informacją decydującą o czytelności diagramu DFD.



Przykład FHD

Aby stworzyć dobry Diagram Hierarchii Funkcji należy pamiętać aby hierarchia była:

- dokładna i spójna na każdym poziomie,
- kompletna, czyli powinna pokazywać zamierzony zakres,
- spójna z modelem informacyjnym i zgodna z przyjętymi konwencjami nazewniczymi,
- zrównoważona, co ułatwi czytanie.

Diagram Związków Encji

Diagram ten tworzony jest w celu dokładnego opisu potrzeb informacyjnych organizacji. Główny cel to zrozumienie struktury danych przetwarzanych w przedsiębiorstwie oraz dostarczenie modelu niezależnego od sposobu przetwarzania danych i dostępu do nich, umożliwiającego podjęcie decyzji o metodzie implementacji i współdziałaniu z istniejącymi systemami.

Model struktury danych jest najczęściej tworzony z wykorzystaniem diagramów pojęciowych (konceptualnych), z których najpopularniejszym jest właśnie ERD. Diagram ten spotyka się w wielu różnych notacjach, do których zaliczamy m.in. notacje: Chena, Martina, Bachmana czy Bakera.

Diagram ERD przedstawia:

- obiekty, o których informacje są istotne z punktu widzenia realizacji celów strategicznych firmy,
- atrybuty obiektów,
- związki pomiędzy obiektami.

Wyodrębnione obiekty mogą być zarówno rzeczywiste jak i mogą być pojęciami abstrakcyjnymi. W diagramie ERD zasadniczą rolę odgrywają trzy pojęcia, których zrozumienie jest kluczowe dla opanowania umiejętności tworzenia tego diagramu. Są to:

- **ENCJA**, rozumiana jako rzecz lub obiekt mający dla nas znaczenie, rzeczywisty lub wyobrażony, o którym informacje muszą być znane i przechowywane,

Przykłady encji

- **ATRYBUT**, który jest dowolnym opisem mającym znaczenie dla encji. Atrybut może być tekstem, liczbą, wartością logiczną lub obrazem,
- **ZWIĄZEK**, będący nazwanym, istotnym powiązaniem pomiędzy dwiema encjami. Związki przedstawiają zależności zachodzące pomiędzy modelowanymi

obiektami. Każdy związek ma dwa końce, z których każdy ma przypisane następujące atrybuty:

- nazwę,
- liczebność,
- opcjonalność.

Związek jest reprezentowany za pomocą linii łączącej dwie encje.

Przykład związku między encjami

Oczywiście ze względu na liczebność możemy wyróżnić kilka typów związków. Oto one:

Związek Jeden do Wielu (1:n)

Związek Wiele do Wielu (m:n)

Związek Jeden do Jeden (1:1)

Zdecydowanie rzadziej stosowanymi związkami są:

– związek nadtyp-podtyp,

– związek wykluczający czyli łuk.

Ciekawą sprawą jeśli chodzi o Diagram Związków Encji jest ilość powszechnie używanych notacji. Mnogość tą powiększają jeszcze bardziej liczne narzędzia

wspomagające rysowanie ERD. Trzeba przy tym zaznaczyć, iż jedynie w przypadku dużych narzędzi klasy CASE stworzonych przez renomowane firmy informatyczne możemy mówić o określonej notacji. W większości mniejszych rozwiązań mamy do czynienia z różnego rodzaju hybrydami lub całkowicie autorskimi rozwiązaniami, których trudno jest doszukiwać się w literaturze. Wprowadza to niemały zamęt w tej kwestii i utrudnia jednoznaczność interpretacji wyników.

Do najbardziej znanych notacji stosowanych w procesie tworzenia diagramów ERD zaliczamy:

- Notacja Chena

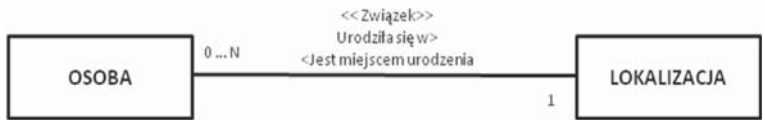
- Notacja IDEF1X

- Notacja Bachmana

- Notacja Martina (Crow's foot)

- Notacja Min-Max (ISO) Jean-Raymond Abriala

- Notacja UML



- Notacja Bakera

Diagram Przepływu Danych

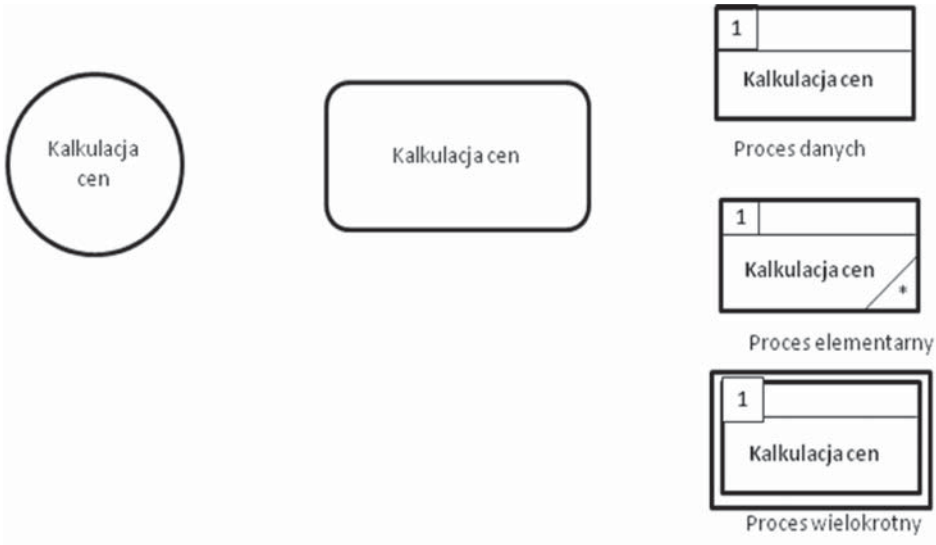
Jedną z metod wykorzystywanych na etapie analizy i projektowania służących do modelowania funkcji systemu jest Diagram Przepływu Danych (ang. *Data Flow Diagram – DFD*). Przedstawia on, w jaki sposób dane przepływają w systemie oraz opisuje procesy przetwarzające dane. Tworzenie diagramu DFD opiera się na następujących kategoriach pojęciowych:

- proces,
- przepływ danych,
- magazyn danych,
- terminator

i odpowiadających im symbolach graficznych.

- **PROCES** (ang. *process*) oznacza transformację danych wejściowych w wynikowe i odpowiada tym składnikom systemu, które przetwarzają dane. Procesy otrzymują i przesyłają dane za pośrednictwem przepływów danych. Kojarzą się one z procedurą, której specyfikacja jest przedstawiona przy użyciu innych technik strukturalnych.

Yordon – DeMarco Gane-Sarson SSADM



- **PRZEPŁYW DANYCH** (ang. *data flow*) – opisuje zbiór danych przepływający pomiędzy dwoma obiektami w systemie. Przedstawia się go za pomocą linii ze strzałkami określającymi kierunek przesyłania informacji. Linie są skierowane najczęściej jednostronnie. Jeśli przekazywana informacja jest zwrotna używa się kolejnych linii lub strzałek dwukierunkowych.

Yordon – DeMarco Gane-Sarson SSADM

- **MAGAZYN DANYCH** (ang. *data store*) – inaczej składnica danych służy do przechowywania danych w postaci jednorodnych kolekcji. Zaistnienie magazynu danych w diagramie ma sens, gdy przechowywane dane służą do realizacji, co najmniej dwóch procesów. Charakter magazynu danych zależy od stopnia szczegółowości diagramu.

Yordon – DeMarco Gane-Sarson SSADM

- **TERMINATOR** (ang. *terminator*) – obiekt zewnętrzny w stosunku do systemu reprezentujący źródła lub miejsca przeznaczenia informacji. Terminatorami są obiekty, z którymi system komunikuje się.

Yordon – DeMarco Gane-Sarson SSADM

Podobnie jak to miało miejsce w przypadku diagramu ERD tak i tutaj mamy do czynienia z kilkoma notacjami. Sposoby reprezentacji poszczególnych elementów pokazano powyżej.

Diagram kontekstowy jest specjalnym graficznym schematem przepływu danych, który definiuje zakres i granice systemu. System przedstawiony jest na diagramie jako pojedynczy proces powiązany bezpośrednio przepływami danych z terminatorami. Całość ma na celu przedstawienie powiązań systemu ze środowiskiem zewnętrznym.

Następnie tworzony jest **diagram systemowy** ukazujący główne funkcje systemu. Każda funkcja jest przedstawiona w postaci hierarchicznej grupy diagramów niższego poziomu.

Diagramy niższych poziomów to **diagramy szczegółowe**. Zbiór diagramów DFD dla systemu wraz z opisem elementów występujących na diagramie w słowniku danych stanowi model funkcji systemu. Jest to graficzna mapa procesów ukazująca przepływ danych miedzy procesami w systemie oraz między światem zewnętrznym a systemem.

4. PODSUMOWANIE

W artykule przedstawione zostały cechy charakterystyczne oraz elementy notacji modelowania strukturalnego jako przeciwwagi modelowania obiektowego. Zaprezentowano obszary zastosowań a także główne tendencje rozwoju myśli informatycznej w tym obszarze. Niniejszy artykuł nie wyczerpuje oczywiście tematu i stanowi niejako wprowadzenie do rozważań bardziej szczegółowych zawierających opisy najlepszych praktyk, wskazówek praktycznych oraz przykładów najczęściej popełnianych błędów przy tworzeniu poszczególnych diagramów. Konkludując powyższe rozważania można stwierdzić, iż techniki modelowania strukturalnego nadal znajdują swoje obszary zastosowań i stanowią ważne uzupełnienie technik obiektowych. Natomiast co do jakości wytwarzanych diagramów przez zespoły analityków i projektantów, jak w wielu przypadkach tak i tutaj można przywołać powszechne powiedzenie, że praktyka czyni mistrza.

Literatura

J. Roszkowski, *Analiza i projektowanie strukturalne.* Wydanie III; Wydawnictwo Helion, Gliwice 2004

R. Barker, C. Longman, *CASE*Method. Modelowanie funkcji i procesów*, Wydawnictwo Naukowo-Techniczne, Warszawa 1996

R. Dumnicki, A. Kasprzyk, M. Kozłowski, *Analiza i projektowanie obiektowe*, Wydawnictwo Helion, Gliwice 1998

A. Jaszkiewicz, *Inżynieria oprogramowania*, Wydawnictwo Helion, Gliwice 1997