# WPROWADZENIE DO PROJEKTOWANIA ELEMENTÓW STRUKTURALNYCH

"Traktujemy zmiany jako coś co powoduje nasze trudności. Lecz ja nie jestem pewien, czy nie jest to raczej coś z czym powinniśmy nauczyć się radzić sobie, niż coś na co mamy prawo się uskarżać."

/George Cox, Diebold UK/

## Specyfika podejścia projektowego w sferze realizacyjnej

W metodzie projektowania strukturalnego chodzi nie tyle o jednorazowe zaprojektowanie spójnego /"dobrze związanego"/ systemu, ile o ciągły proces budowy systemu poprzez dokładanie nowych pakietów problemowych. Założeniem metody jest dokonywanie wiązań elementów składowych nie podczas ich tworzenia lecz w trakcie eksploatacji, a ściślej rzecz biorąc, po ulokowaniu ich w prasystemie. Budowa systemu polega więc głównie na tworzeniu półfabrykatów i mechanizmów montażowych. Przesłanką montażu jest wyposażenie prefabrykatów w zmienne strukturalne, tj. modyfikatory procedur, odsyłacze adresowe /pointery/ danych, itp. Postać systemu zależy więc tutaj nie tylko od intuicji, kwalifikacji i doświadczenia poszczególnych projektantów, lecz wynika z reguł specyficznej inżynierii systemów.

Ze względu na wstępny etap rozwoju trudno jest w tej chwili zakwalifikować strukturalne projektowanie do jednej ze znanych kategorii metod projektowania. Na pewno nie należy ono do metod diagnostycznych, gdyż nie opiera swych rozwiązań na analizie stanu dotychczasowego. Mimo, iż preferuje obsługę problemów nieoczekiwanych, nie można go zaliczyć do metod prognostycznych typu Nadlera, gdyż nie stosuje techniki kolejnych przybliżeń jako podejścia podstawowego. Zapewne jest ono bliskie kategorii metod

aksjomatycznych, gdyż zakłada z góry szereg przesłanek /np. prasystem/ określających sposób realizacji systemu[1/].

Specyfiką działań w sferze realizacji jest projektowanie wielostrukturowe niekoniecznie wielowarstwowe. Zakłada się tutaj bowiem występowanie następujących kategorii struktur:

- informacyjnych /dotyczących struktur danych/,
- proceduralnych /dotyczących organizacji wewnętrznej procedur/,
- konstrukcyjnej /dotyczącej zespolenia powyższych struktur w jednostki eksploatacyjne typu program, zadanie lub dynamicznie sterowany pakiet obsługi transakcji/,
- użytkowej /dotyczącej układu treści dokumentów wejściowych, zawartości baz danych oraz możliwości wyszukiwania informacji/.

Pierwsze trzy kategorie prezentują głównie technologiczny punkt widzenia, zaś ostatnia charakteryzuje system od strony widocznej dla użytkowników. Dodajmy, że w sferze poznawczej występuje struktura funkcjonalna problemu /związana z dekompozycją problemu/ zorientowana na użytkownika, aczkolwiek zawierająca przesłanki dla technologii informatycznej.

Wyróżnienie tych kategorii struktur powinno w naszym odczuciu /i przekonaniu/ sprzyjać takim właściwościom systemu informatycznego jak selektywność, adaptacyjność,rozwijalność i zrozumiałość dla użytkownika. Właściwości te uważane są przez specjalistów, np. H.B. Hodsona /22/, za niezbędne atrybuty pakietów ogólnego stosowania. Selektywność umożliwia dobór do pakietu tylko tych składników, które odpowiadają potrzebom problemu, dzięki czemu osiągnąć można lepszą efektywność przetwarzania i uprościć konstrukcję pakietu. Adaptacyjność oznacza w wąskim sensie jedynie możliwość wprowadzania dodatkowych możliwości i modyfikacji w ramach istniejącego rozwiązania.Rozwijalność dotyczy poważniejszych prac rozwojowych składających się na ewolucję systemu. W potocznym rozumieniu wchodzi do rozszerzonego zakresu adaptacyjności.

---

[1/] W projektowaniu struktury pakietów problemowych zastosowanie znaleźć może również metoda morfologiczna, należąca do kategorii metod analityczno-kombinacyjnych.

Cechy elementu strukturalnego

Przez elementy strukturalne rozumiemy tutaj moduły proceduralne, zestawy danych, zestawy zdań sterujących systemu operacyjnego, itp. Każdy element powinien posiadać określone łącze komunikacyjne oraz być podatny na strojenie /modyfikację przedprocesorową/. Ponadto treść każdego elementu powinna podlegać opisowi w języku specyfikacyjnym, który to opis może być katalogowany na równi z treścią, dostarczając projektantom i programistom informacji niezbędnych do sformułowania dyrektyw strojenia. Dalsze cechy zależne są od rodzaju elementu.

Najbogatsze właściwości strukturalne występują w modułach proceduralnych, a mianowicie:

- posiadają one określone granice, zarówno fizyczne /zawarte pomiędzy formalnymi ogranicznikami/ jak i logiczne /wyznaczające zakres zadania logicznego, funkcjonalnego itp./,
- wyposażone są w łącza komunikacyjne /jedno wejściowe i jedno wyjściowe/ do łączności z innymi modułami proceduralnymi,
- są podatne na strojenie /noszą charakter szkieletowy/,
- posiadają przejrzysty przepływ sterowania logicznego /między innymi dzięki zasadzie ograniczonego zaufania do GO TO/,
- zbudowane są z mechanizmów strukturalnych wynikających z metody programowania strukturalnego /a więc sekwencji, selekcji i repetycji/,
- nadają się do niezależnej kompilacji i testowania /czemu towarzyszy odpowiednie podparcie narzędziowe zdolne do symulacji brakujących modułów/,
- są opisywalne w języku specyfikacyjnym, zwroty którego służą zarówno do opisu dokumentacyjnego jak i do strojenia procedur,
- opis modułu znajduje się poza jego tekstem /co między innymi ułatwia wyszukiwanie informacji o procedurach/.

Na charakterystykę zestawów danych /strukturalnych modułów danych/ składają się:

- opis struktury fizycznej /układu danych w pamięci/,
- opis struktury logicznej /w tym opis typu struktury - np. sieciowej binarnej typu CODASYL-owskiego - oraz opis techniki wiązań - np. poprzez pointery wbudowane lub tablice pointerów/,

dane /treść danych/.

Z punktu widzenia wymogów eksploatacyjnych strukturalne moduły danych rozpatrywane zwykle jako suma modułów specjalizowanych powinny zapewnić następujące możliwości:

- przetwarzanie danych formatowych,
- przetwarzanie danych tekstowych,
- efektywne aktualizowanie danych,
- efektywne wyszukiwanie danych,
- możliwość tworzenia różnorodnych struktur danych,
- niezależność danych oparta na wewnętrznym słowniku danych, udostępniającym adresy danych podczas przetwarzania,
- efektywną reorganizację i restrukturalizację bazy danych,
- zabezpieczenie danych przed niepowołanym dostępem i zniszczeniem.

Kluczową kwestią projektowania strukturalnego jest łączenie strukturalnych modułów danych i strukturalnych modułów proceduralnych w jednostki programowe oraz opatrywanie ich w gotowe zestawy zdań sterujących niezbędne do komunikacji z systemem operacyjnym komputera. Generowanie zdań sterujących oparte być może na uzupełnianiu wzorców zdań parametrami wprowadzanymi za pomocą zwrotów języka specyfikacyjnego.

Pełny zakres projektowania strukturalnego obejmuje więc kompleks elementów niezbędnych do budowy systemu informatycznego. Jest on co prawda trudny do zrealizowania przy obecnym stadium rozwoju informatyki krajowej, niemniej jednak stanowi perspektywę, do której należy dążyć w jednostkach organizacyjnych zajmujących się przemysłową produkcją oprogramowania. W szczególności dotyczy to ewolucji systemów zarządzania bazą danych w kierunku, szerszego niż dotąd, uwzględnienia komputerowego wspomagania projektowania.

Wybrane problemy projektowania elementów strukturalnych omówimy w następnych rozdziałach.

Wymogi stawiane systemom zarządzania bazą danych przez metodę projektowania strukturalnego

Jak już podkreślaliśmy, projektowanie strukturalne oparte jest na możności elastycznego doboru wszelakich elementów potrzebnych do rozwiązania problemu. O ile w stosunku do elementów proceduralnych oznacza to zwykle ich fizyczną dystrybucję do określonych pakietów problemowych, to w zakresie treści danych pociągać to powinno jedynie logiczne przyporządkowania do pakietów /dane pozostają cały czas we wspólnych bazach danych pod kontrolą softwareowego administratora baz danych/. Wynika z tego, że tylko niektóre elementy systemu zarządzania bazą danych będą stale rezydować podczas przetwarzania dowolnego pakietu problemowego lub będą przywoływane nakładkowo. Do rezydentnych elementów systemu zarządzania bazą danych zaliczyć należy co najmniej wewnętrzny słownik danych /internal data dictionary/ dostarczający informacji adresowych oraz procedury kontroli dostępu do danych. Pozostałe elementy proceduralne mogą być włączane do pakietów problemowych zgodnie z zapotrzebowaniem problemu. Pozwoli to zmniejszyć zapotrzebowanie na konfigurację komputerów, zwiększyć efektywność przetwarzania i ułatwić konserwację /utrzymanie/ pakietów. Na poparcie tego postulatu /aby większość oprogramowania baz danych traktować jako elementy prasystemu/ przytoczymy pogląd T. Gilba wyrażony w /64, s.21/: "Jest możliwe, że argumenty na korzyść bazy danych są, ogólnie rzecz biorąc, ważne do pewnego punktu krytycznego, następnie zaś stają się nieistotne w rezultacie wysokich kosztów administrowania i utrzymywania złożonych funkcji centralnych. Jak dotychczas, nikt nie zadał pytania, w którym miejscu krzywej opłacalności znajduje się ten punkt. Całe nasze doświadczenie mówi, że większość systemów posiada taki punkt optymalny i że po przekroczeniu pewnego poziomu złożoności zaczynają się one dezintegrować."

Wykorzystanie systemów zarządzania bazą danych w charakterze tworzywa strukturalnego /jako części prasystemu/ jest uwarunkowane spełnieniem przez nie następujących wymogów:

- występowanie kilkupoziomowych schematów danych, umożliwiających projektowanie docelowej bazy danych /conceptual schema/ oraz implementację programową jej fragmentów /external and internal schema/,
- zdolność do obsługi procedur aplikacyjnych pisanych w różnych językach programowania,
- możliwość nawiązania komunikacji z innymi bazami lub systemami zarządzania bazą danych,
- zabezpieczenie złożonych logicznych struktur danych /np. sieciowych/,
- akceptowanie różnorodnych fizycznych struktur danych i zabezpieczenie odpowiednich procedur fizycznego dostępu,
- możliwość katalogowania procedur szkieletowych oraz ich opisu,
- sprawny system dystrybucji danych do baz danych /oparty np.na metodzie przetwarzania sterowanego danymi opisanej w /39// umożliwiający zachowanie jednolitych zasad zasilania baz niezależnie od tego w jakim pakiecie problemowym wystąpiły dane,
- wyposażenie w aparat komputerowego wspomagania projektantów i programistów, tak w zakresie dokumentacji jak i generowania programów.

Obecne tendencje rozwoju systemów zarządzania bazą danych zdają się przybliżać realizację tych wymogów. Systemy te często wbudowywane są w duże kompleksy programowe realizujące nie tylko zadania związane z bazami danych /np. kompleks TIME z systemem TOTAL, CDMS z IDMS, PROTEE + APPEL IV/. Ponadto pojawiają się pakiety programowe tzw. niezależnego słownika danych /np.DATAMANAGER/, pełniące funkcję interface'u pomiędzy różnymi systemami zarządzania bazą danych /w tym generowania kodów słownika oraz definicji danych/ /13/. Niektóre systemy wyposażone są w łącza dostępu do innych baz danych, np. MARK-IV posiada łącze nieproceduralnego dostępu do IMS lub do TOTAL. Istnieją też konwertory baz danych, np.Multics Data Base Manager firmy Honeywell realizuje konwersję sieciowego lub hierarchicznego modelu danych na model relacyjny /18/.

W skład prasystemu wchodzić powinien system zarządzania bazą danych zdolny do gromadzenia wielu różnorodnych informacji, a w tym opisu dokumentacyjnego elementów prasystemu.W tym wzglę-

dzie na uwagę zasługuje system IDMS /13, s.77-80/ wyposażony w tzw. Zintegrowany Słownik Danych /Integrated Data Dictionary - IDD/,który z kolei poza językami DDL /Data Description Language/ i DML /Data Manipulation Language/ wymaga stosowania języka DDDL /Data Dictionary Definition Language/. Za pomocą tego ostatniego języka definiowane są takie elementy jak: dane, zapisy, zbiory, moduły, programy, systemy, użytkownicy, jednostki organizacyjne. Podobne możliwości zawiera wspomniany już pakiet niezależnego słownika danych DATAMANAGER firmy Management Systems and Programming.

Poza możliwością katalogowania opisów łącze software'owe powinno być zdolne do strojenia procedur szkieletowych,przekazywania parametrów pomiędzy procedurami pisanymi w różnych językach programowania oraz montażu procedur w większe jednostki proceduralne. Są to czynności znacznie szersze i bardziej skomplikowane niż pointerowe wiązanie danych w klasycznych systemach zarządzania bazą danych.

Istotną właściwością łącza jest łatwość powiązań z programami użytkowymi i użytkownikami. Dla potrzeb programów automatycznie generowane są kody źródłowe /w przypadku IDD - w COBOL-u lub PL/1/ ze zwrotów języka specyfikacyjnego, zaś z "naturalnego" języka żądań informacji /którym posługuje się użytkownik nieprogramista/ za pomocą tzw. makroprzedprocesora generowany jest /np.w rozwiniętej wersji ADABAS-u/ formalny język wyszukiwania, a ściśle rzecz biorąc procedury w tym języku.

Jak można się zorientować z podanych wymogów, projektowanie strukturalne w istocie swej stanowi złożony kompleks działań w trakcie których niezbędne jest posługiwanie się nie tylko zasadami metodycznymi, lecz również narzędziami software'owymi. Narzędzia te wspomagają projektanta, dając mu wgląd w skatalogowane zasoby danych i procedur, umożliwiając ich przekształcenie oraz powiązania z innymi składnikami technologicznymi systemu. Innymi słowami, narzędzia te - w sumie - powinny umożliwić projektantowi dokonanie opisu problemu w języku specyfikacyjnym wyższego rzędu, wyszukać w katalogach elementy strukturalne odpowiednie do rozwiązania problemu, uzupełnić je o elementy specyficzne problemu, połączyć wszystkie elementy wybrane w pakiety

problemowe, wygenerować opisy dokumentacyjne pakietów, sprawdzić ich funkcjonowanie na danych testowych, utrzymywać powiązania pomiędzy składnikami, rejestrować użytkowników i pakiety problemowe oraz każdorazowe użycie elementów prasystemu, modyfikować elementy, wariantować rozwiązania, itp.

W warunkach tych projektant przekształca się z rękodzielnika manufaktury w dyspozytora nowoczesnej wytwórni oprogramowania użytkowego. Zapewne zrodzi to pewne problemy psychologiczne, stworzy rygory kwalifikacyjne /pod względem znajomości narzędzi projektowo-programowych/ i zapewne pozbawi projektanta pewnej cząstki satysfakcji osobistej /wskutek użycia wielu elementów pochodzenia "obcego"/ oraz może go oddalić od technologii komputerowego przetwarzania danych /gdyż staje się bardziej montażystą niż twórcą/. Być może będzie to cena, jaką przyjdzie zapłacić za postęp techniczny w procesie projektowania i programowania, za postęp który umożliwi lepsze spożytkowanie społecznego wkładu twórców oprogramowania.