Problem stosowania wybranych kategorii składniowo-semantycznych języka angielskiego w języku programowania aplikacji baz danych 4GL

Leszek Szczepaniak
Akademia Techniczno-Rolnicza w Bydgoszczy,
Instytut Matematyki i Fizyki, Zakład Zastosowań Matematyki,
leszol@atr.bydgoszcz.pl

Streszczenie

Języki programowania aplikacji baz danych 4GL w coraz większym stopniu stosują rozwiązania składniowo-semantyczne zaczerpnięte z języka angielskiego. Niniejszy artykuł zwraca uwagę na pilną potrzebę dokonania systematycznej analizy i oceny rozwiązań zapożyczonych z języka naturalnego a występujących coraz powszechniej w językach programowania. W artykule zaprezentowano metodologię jaką przyjęto przy próbie dokonania takiej właśnie analizy oraz przedstawiono jedno z podstawowych kryteriów stosowane do oceny „zapożyczonych z języka naturalnego" rozwiązań składniowo-semantycznych występujących w języku programowania aplikacji baz danych Progress [1].

1. Potrzeba systematycznej analizy porównawczej języka programowania 4GL i języka angielskiego

Współczesne języki programowania aplikacji baz danych, określane terminem 4GL, w dużo większym stopniu niż ich poprzednicy – języki 3GL, wykorzystują wybrane elementy składniowo-semantyczne zapożyczone z języka angielskiego.

Przykładem zapożyczonych kategorii może być składnia komend, która w językach 3GL ograniczona była do identyfikatora procedury, po której występowała lista argumentów, podczas gdy w językach 4GL, postać komendy bardziej przypomina zdanie języka naturalnego. Przykładowo w języku Progress (którego wybrano jako reprezentanta języków programowania aplikacji baz danych – 4GL) komenda ukonstytuowana jest przez czasownik

wyrażający polecenie (np. Display), po którym występujące argumenty są pogrupowane w frazy, często wprowadzane przyimkiem – jak ma to miejsce w przypadku zdań języka naturalnego (np. with, when), a całość komendy zakończona jest kropką, np.:

```
DISPLAY name  address postal-code WHEN postal-code NE ""
        WITH NO-BOX NO-LABELS.
```

Podobna tendencja występuje w odniesieniu do deklaracji, w których autorzy języka starają się unikać symbolicznych notacji, na rzecz postaci zbliżonej do wypowiedzi języka naturalnego, np.:

```
DEFINE NEW GLOBAL SHARED VARIABLE first-time
        AS LOGICAL INITIAL TRUE.
```

Innym elementem zapożyczonym z języka naturalnego jest większa niż w językach 3GL kontekstowość języka. Sprawia ona, że wypowiedzi mogą być bardziej zwarte. Przykładem tego są komendy blokowe, których funkcja w językach 3GL ograniczała się zazwyczaj do grupowania instrukcji, podczas gdy przykładowo w języku Progress, komenda blokowa: FOR EACH, definiuje szeroki kontekst, w skład którego wchodzą między innymi: domyślne bufory relacji, obsługa sytuacji wyjątkowych, transakcje, blokowanie rekordów, domyślna formatka itp. Właściwości samego bloku zależą nie tylko od obecności lub braku jawnej deklaracji danego elementu ale też od instrukcji i operacji jakie mają miejsce wewnątrz danego bloku, oraz właściwości bloku, w którym dany blok występuje.

Idea szeroko pojętego kontekstu, wpływającego na sposób wykonania danej instrukcji jest zapożyczona z języka naturalnego, w którym właściwa interpretacja wypowiedzi mocno zależy od kontekstu w jakim została wypowiedziana.

Wszystkie te zabiegi sprawiły, iż języki programowania 4GL są bardziej czytelne, łatwiejsze do nauki i wygodniejsze w użyciu niż języki 3GL. Z drugie jednak strony elementy zapożyczone z języka naturalnego są często wprowadzane bardzo chaotycznie i niekonsekwentnie. Brak jest pogłębionej analizy odnoszącej przyjętą definicję składni języka programowania do języka naturalnego z którego zaczerpnięto dane rozwiązanie.

Wszystkie te negatywy, sprawiają, iż zarówno użytkownicy danego języka programowania jak i jego autorzy, widzą ów język jako twór całkowicie sztuczny nie mający z językiem naturalnym żadnego związku poza faktem użycia wybranych słów języka angielskiego.

„W badaniach lingwistycznych istnieje nurt określany terminem gramatyki kontrastywnej, zajmujący się opisem dwóch lub więcej języków pod kątem podobieństw i różnic występujących między nimi. Badania kontrastywne prowadzone w Ameryce wyrosły głównie z potrzeb praktycznych i służyły przede wszystkim do doskonalenia metod nauczania języków obcych. Zakładano, że trudności w nauce języków obcych wywołane są głównie różnicami między językiem ojczystym a językiem nauczanym. Różnicom tym przypisywano decydującą rolę w powstawaniu błędów u uczniów. Spodziewano się, że opis kontrastywny pozwoli przewidzieć takie błędy i ułatwi ich zwalczanie. Materiały do nauczania języków oparte na takim opisie uważano za najlepsze i najbardziej skuteczne” [3].

Analiza kontrastywna powinna mieć również zastosowanie w odniesieniu do języka sztucznego i języka naturalnego, z którego zaczerpnięto znaczną ilość elementów

składniowych. Duża ilość zapożyczonych z języka naturalnego rozwiązań składniowo-semantycznych sprawia, iż osoba ucząca się danego języka programowania odnosi go do języka naturalnego, z którego pochodzą zapożyczenia.

Źródłem języków programowania 4GL są z jednej strony języki programowania 3GL, a z drugiej język angielski. O ile odniesienia przyjętych rozwiązań składniowo-semantycznych do języków 3GL są w pełni uwzględnione, o tyle odniesienia do języka naturalnego są praktycznie całkowicie pomijane.

Zadaniem opisanej niniejszym artykułem pracy jest uzupełnienie powstałej luki. Praca ta będzie próbą analizy porównawczej języka programowania aplikacji baz danych Progress oraz języka angielskiego. Będzie to próba spojrzenia na język programowania z punktu widzenia języka naturalnego, próba uchwycenia podobieństw i różnic, próba „zobaczenia" języka programowania w świetle kategorii składniowo-semantycznych języka naturalnego.

2. Potrzeba wyboru kategorii składniowo-semantycznych języka angielskiego mogących mieć zastosowanie przy konstruowaniu języków programowania 4GL

Aktualnie powstaje wiele systemów informatycznych, w których językiem komunikacji użytkownika z systemem jest kontrolowany język angielski. Są wśród nich także systemy programowania [5]. Niestety dokonany w nich wybór podzbioru języka angielskiego nie jest oparty o dogłębną analizę potrzeb i ich powiązania z mechanizmami jakimi dysponuje język naturalny. Wybór ten jest zazwyczaj podyktowany tylko względami pragmatycznymi.

Przy konstruowaniu języków kontrolowanych nie wypracowano dotychczas metody pozwalającej na świadomy dobór środków składniowo-semantycznych pozwalających na taką konstrukcję języka by siła wyrazu zastosowanych środków była adekwatna do potrzeb wynikających z dziedziny zastosowań danego języka [4].

Brak takiej metodologii jest odczuwalny także w dziedzinie języków programowania aplikacji baz danych - 4GL. Z jednej strony czerpią one z wybranych kategorii języka naturalnego, z drugiej jednak, brak w nich jasnego wyodrębnienia tychże kategorii, oraz brak ich pogłębionej analizy w odniesieniu do potrzeb jakie występują w odniesieniu do języków programowania aplikacji baz danych. Taka sytuacja sprzyja temu, że wybrane kategorie składniowo-semantyczne są niejednokrotnie stosowane chaotycznie, niekonsekwentnie a nawet sprzecznie z intuicją wynikającą z kompetencji językowej użytkownika języka naturalnego.

Termin „kategorie składniowo-semantyczne", odnosi się do tych kategorii semantycznych w danym języku (naturalnym lub sztucznym), które są wyodrębniane czy wyróżniane na poziomie składni języka. Są to zazwyczaj kategorie semantyczne, które ze względu na swoją wagę czy powszechność dysponują własnymi środkami wyrazu obecnymi w warstwie składniowej języka.

Pogłębiona analiza kontrastywna konkretnego języka programowania 4GL i stosunkowo małego podzbioru języka angielskiego dostarczy szeregu spostrzeżeń pozwalających na wyodrębnienie kategorii języka naturalnego, które mogą mieć zastosowanie w językach programowania aplikacji baz danych, oraz na zrezygnowanie z wybranych kategorii języka programowania i zastosowanie w ich miejsce kategorii zapożyczonych z języka naturalnego, bardziej intuicyjnych, i wygodniejszych w użyciu.

3. Przyjęta metodologia

Do przeprowadzenia analizy kontrastywnej języka programowania aplikacji baz danych 4GL wybrano język Progress, a analizę dokonano w dwóch zasadniczych etapach:

> Etap I. Obejmuje wyłącznie analizę wewnętrzną języka programowania 4GL w oderwaniu od języka naturalnego.
>
> Etap II. Obejmuje analizę języka programowania w kontekście kontrolowanego języka naturalnego - Angielskiego

Poniżej przedstawiamy poszczególne kroki analizy:

Etap I.

3.1. Specyfikacja składni języka Progress 4GL

W oparciu o dokumentację języka „PROGRESS On-line Reference" wyspecyfikowano wszystkie pozycje opisu składniowego języka Progress w układzie takim jaki przyjęli autorzy języka. Każda pozycja identyfikowana jest przez unikalna nazwę i/lub numer oraz zawiera opis składniowy tak jak jest on przedstawiony w dokumentacji. Jeśli opis parametrów danej pozycji opisu składniowego zawiera dodatkowe elementy składniowe są one także przytoczone.

3.2. Specyfikacja kategorii składniowo-semantycznych języka Progress

W oparciu o pozycje opisu składniowego języka Progress dokonano specyfikacji kategorii semantyczno-składniowych jakimi posługują się autorzy do opisu języka Progress.

W specyfikacji wydzielono:

- kategorie klasyfikacji podstawowej, w oparciu o które autorzy dokonali rozłącznej klasyfikacji wszystkich pozycji opisujących język Progress – 15 pozycji;
- kategorie opisu składniowego, którymi autorzy języka posługują się w opisie składni poszczególnych elementów języka – 6 pozycji;
- kategorie elementarne, wyznaczające cechy składniowo-semantyczne wyrażeń języka Progress zgrupowane gł. według poszczególnych komend, funkcji itd.;
- kategorie wyznaczone przez symbole nieterminalne użyte w opisie składniowym.

3.3. Analiza kategorii składniowo-semantycznych języka Progress

W kolejnym kroku dokonano analizy wyspecyfikowanych w poprzednim punkcie kategorii języka Progress. Analizę dokonano w oderwaniu od języka naturalnego.

Ma to na celu wyłonienie pryncypiów wynikających z przyjętej kategoryzacji składniowo-semantycznej języka Progress (np. cel istnienia odrębnej kategorii związanej z komendami i nazwami preprocesora).

Etap II.

3.4. Zestawienie odpowiedników w języku angielskim dla wszystkich elementów opisu składniowego języka Progress

Dla każdej pozycji opisu składniowego podano jej odpowiedniki wyrażone w języku naturalnym - w kontrolowanym języku angielskim.

- Dla każdej pozycji opisu składniowego podano jeden lub kilka odpowiedników tak by obejmowały one wszystkie warianty składniowe danej pozycji, oraz wszystkie możliwe parametry jakie mogą w danej pozycji wystąpić.
- Odpowiedniki nie zawierają opisu semantycznego danej pozycji czy jakiegoś parametru, ale wyrażają dany element w kontrolowanym języku angielskim. Zostały one utworzone w oparciu o zdania bądź zwroty używane w dokumentacji „PROGRESS On-line Reference" opisującej dany element.
- Odpowiednik w języku naturalnym starano się wyrazić w możliwie najprostszej i najkrótszej postaci, tak by uwzględnić wszystkie możliwości wystąpienia lub pominięcia danego parametru i parametrów z nim związanych. Ze zdań lub fraz, w oparciu o które utworzono dany odpowiednik w języku naturalnym, pominięto frazy, które zawierają opis semantyczny.
- Odpowiedniki związane z daną pozycją opisu składniowego składają się zazwyczaj z jednego lub kilku możliwie najprostszych zdań (lub fraz nominalnych), które wyrażają zależności pomiędzy wszystkimi elementami.
- Dla każdej pozycji w oparciu o podane odpowiedniki podany jest przykład wyrażenia tejże pozycji w formie skondensowanej – w formie jednego zdania lub frazy.

3.5. Specyfikacja używanych kategorii składniowo-semantycznych języka angielskiego

W kolejnym kroku dokonano analizy językowej angielskich odpowiedników poszczególnych pozycji opisu składniowego, w oparciu o którą wyodrębniono kategorie semantyczno-składniowe użytego w opisie kontrolowanego języka angielskiego.

Wśród wyłonionych kategorii języka naturalnego znajdują się przykładowo następujące kategorie:

- wyrażanie przynależności gatunkowej (nazwy gatunkowe, predykat *is a*)
- wyrażanie przynależności: dopełniacz *'s*, przydawka przedrzeczownikowa, przyimki *of, with, without,* predykat *has*
- wyrażanie mnogości: liczba mnoga rzeczowników, liczebniki główne, wyliczenie
- kwantyfikatory: *all, each, every*

- wyrażanie właściwości: przymiotniki atrybutywne i predykatywne, przymiotniki z dopełnieniami przyimkowymi, stopniowanie przymiotników, zaimki względne w zdaniu nieograniczającym
- wskazywanie: zaimki osobowe, dzierżawcze i wskazujące, liczebniki porządkowe, zaimki względne w zdaniu ograniczającym, przedimek nieokreślony *a/an*, przedimek określony *the*

3.6. Analiza porównawcza kategorii składniowo-semantycznych języka Progress i języka angielskiego

W oparciu o wyspecyfikowane w krokach 3 i 5 kategorie składniowo-semantyczne języka Progress i języka angielskiego dokonuje się analizy porównawczej obu języków pod kątem występujących podobieństw, różnic i sprzeczności. W kroku tym poddana została szczególnej analizie możliwość zastosowania wybranej kategorii języka naturalnego w języku programowania 4GL.

Jednym z istotnych problemów jaki wyłonił się podczas prowadzenia analizy porównawczej był problem zgodności składniowo-semantycznej kategorii języka programowania i języka naturalnego. Problem ten dokładniej przedstawimy w punkcie 5, ale zanim to uczynimy musimy wpierw dokonać pewne dodatkowe wyjaśnienia terminologiczne.

4. Dwa poziomy desygnatów języka programowania - desygnaty wirtualne i realne

Desygnacja (ang. reference) – czyli oznaczanie jest najważniejszym elementem funkcjonowania języka. Desygnacja jest relacją między formami językowymi (zdaniami, grupami wyrazów, wyrazami, morfemami) a tym o czym się mówi i pisze, co jest odzwierciedlone w umyśle uczestników aktu porozumiewania się. Każdy byt, czy sytuację znajdujące się w umyśle użytkownika języka nazywać będziemy desygnatem (ang. referent) [2](str. 172).

Język programowania posługuje się dwoma poziomami desygnatów. Pierwszy poziom desygnatów można by określić mianem desygnatów wirtualnych, które powstają w umyśle programisty pod wpływem danego języka programowania. Można je nazwać bytami programistycznymi. Składają się na nie desygnaty występujące w danym języku programowania, takie jak: zmienne, tablice, pola rekordów, procedury, obiekty ekranowe, bufory, pytania, formatki, klasy, typy. Występują one w kodzie programu i zarazem funkcjonują one w umyśle programisty posługującego się danym językiem programowania. Podstawową metodą wskazywania takich desygnatów jest posługiwanie się identyfikatorami. W momencie deklarowania danego desygnatu wiążemy z nim unikalny w danym kontekście identyfikator, którego używamy następnie w celu wskazania danego desygnatu.

Drugi poziom desygnatów można nazwać desygnatami realnymi, istniejącymi w umyśle użytkownika języka naturalnego, odzwierciedlające zazwyczaj byty świata rzeczywistego. Przykładem mogą być tu desygnaty występujące w realnym świecie, takie jak: instytucje, osoby, dokumenty, produkty, procesy produkcyjne. Do desygnatów realnych należą nie tylko byty ale także atrybuty z nimi związane: właściwości, cechy itp., jak na przykład: data wystawienia dokumentu, numer, rozmiar, robocizna, nazwa. Ogólnie, z punktu widzenia

języka programowania, przyjmujemy, że desygnatami świata realnego są byty abstrakcyjne stanowiące atrybuty bytów rzeczywistych.

Desygnaty wirtualne, czyli te z poziomu języka programowania, reprezentują desygnaty realne, występujące w opisywanym fragmencie świata. Przykładowo: tablica bazy danych reprezentuje grupę kontrahentów, produktów, dokumentów, rodzaje płatności, rodzaje warunków kredytowych; rekord reprezentuje pojedynczego kontrahenta, produkt, dokument, sposób płatności, warunek kredytowy a pytanie (ang. query) reprezentuje podzbiór jakiejś zbiorowości desygnatów realnych o określonych właściwościach.

Przykładowo pytanie:

```
DEFINE QUERY qi FOR item.
OPEN QUERY qi FOR EACH item WHERE item-num > 40.
```

definiuje zbiorowość rzeczy, których numery są większe od 40,

a zmienna:

```
DEFINE VARIABLE  order-date AS date.
order-date = '01-02-2001'.
```

reprezentuje datę jakiegoś zamówienia.

Wśród bytów wirtualnych można wydzielić byty wirtualne własne, które istnieją jedynie w danym języku programowania, i nie mają bezpośredniego odniesienia do bytów realnych. Przykładem może być instrukcja bloku, która grupuje instrukcje składające się na procedurę i sama w sobie pełni rolę pomocniczą wyłącznie w języku programowania nie mając bezpośredniego odniesienia do świata realnego.

Wprowadzone tu rozróżnienie desygnatów na wirtualne i realne jest podstawą do wprowadzenia pojęcia zgodności lub braku zgodności składniowo-semantycznej kategorii języka programowania i języka naturalnego.

5. Zgodność składniowo-semantyczna kategorii języka programowania i języka naturalnego

Język programowania wprowadza środki wyrazu pozwalające wyrażać zależności semantyczne pomiędzy bytami wirtualnymi. Określamy je terminem: „kategorie składniowo-semantyczne języka programowania". Przykładami takich kategorii mogą być: deklarowanie zmiennej czy procedury; wyliczanie czy wskazywanie.

Podobnie i język naturalny wprowadza środki wyrazu pozwalające wyrażać zależności pomiędzy bytami realnymi. Określamy je terminem „kategorie składniowo-semantyczne języka naturalnego". Przykładami takich kategorii mogą być: wyrażanie przynależności, wyrażanie mnogości, nazwy własne, ograniczanie zakresu oznaczania przy pomocy zdań względnych.

Zgodność składniowo-semantyczna kategorii języka programowania i języka naturalnego dokonuje się na dwóch poziomach: wewnętrznym i zewnętrznym. Omówimy je poniżej w dwóch punktach.

5.1. Zgodność wewnętrzna

Rozważmy następujący przykład:

Byty wirtualne: widget oraz event z poziomu języka programowania są związane ze sobą w relacją przynależności, gdyż język Progress wiąże z poszczególnymi obiektami ekranowymi (ang. widgets) zdarzenia, które są przez dany obiekt obsługiwane. Tak zdefiniowany przez język programowania związek pomiędzy zdarzeniem a obiektem ekranowym, jest przez użytkownika języka naturalnego postrzegany w kategorii „przynależność", która to kategoria w języku angielskim może być wyrażona przy pomocy przyimka *of* np.:

the event of the widget, the selection of the menu

Ponieważ język Progress także wyraża tę zależność przy pomocy przyimka *of* mówimy o wewnętrznej zgodności składniowo-semantycznej danej kategorii języka programowania i języka angielskiego.

Rozważmy inny przykład:

Tablica bazy danych składa się z pól, a pole jest częścią składową tablicy. Język Progress dla wyrażenia relacji przynależności pola do tablicy używa operatora „." (kropka), np.:

Display order.number

W tym przypadku brak jest zgodności wewnętrznej języka programowania i języka naturalnego, gdyż relacja pomiędzy nazwą tablicy i jej składowym elementem – polem, tak jak jest ona zdefiniowana w języku Progress, jest przez użytkownika języka naturalnego także postrzegana w kategorii „przynależności" i z tego też względu winna być wyrażona przy użyciu przyimka *of* lub w formie przydawki przed-rzeczownikowej w postaci rzeczownika *order*:

Display number of order.
Display order number.

5.2. Zgodność zewnętrzna

Kwestię zgodności zewnętrznej danej kategorii składniowo-semantycznej języka programowania i języka naturalnego wyjaśnimy na pierwszym przykładzie z poprzedniego punktu. Wirtualne byty określane terminem *widgets* mają swoje realne odniesienie, gdyż funkcjonują w umyśle użytkownika języka naturalnego jako obiekty występujące na ekranie. Podobnie i zdarzenia, jak np. wybranie danej opcji menu, mają swoje realne odzwierciedlenie w umyśle użytkownika języka naturalnego.

Niestety, programista – użytkownik języka programowania postrzega zależność pomiędzy obiektem ekranowym a obsługiwanym przez niego zdarzeniem inaczej niż użytkownik języka naturalnego.

Zdarzenia w umyśle użytkownika języka naturalnego są wyrażane zasadniczo przy pomocy czasowników i form łączących funkcje czasowników z innymi częściami mowy (np.

imiesłowu). W takim ujęciu, pomiędzy określonym zdarzeniem a obiektem ekranowym będącym uczestnikiem tego zdarzenia nie występuje relacja „przynależności". Programista natomiast, zależność pomiędzy obiektem ekranowym a zdarzeniem postrzega i wyraża w kategorii „przynależności".

Mamy tu więc przypadek braku zgodności zewnętrznej. Środki wyrazu zastosowane do wyrażenia przykładowej referencji zapisanej w języku Progress, pomimo zgodności wewnętrznej, nie pokrywają się ze środkami wyrazu jakie zastosowałby użytkownik języka naturalnego:

```
on choose of button do ...
```

Przytoczona powyżej fraza języka Progress definiująca wyzwalacz (ang. trigger) obsługujący wystąpienie zdarzenia wybrania (przyciśnięcia) obiektu ekranowego- przycisku, nie jest poprawną frazą języka naturalnego, gdyż użytkownik języka naturalnego postrzega relację pomiędzy tymi bytami w innych kategoriach składniowo-semantycznych. Poprawną frazą języka angielskiego byłoby:

```
On choosing the button do ...
```

Wprowadzone tu pojęcie zgodności wewnętrznej i zewnętrznej jest bardzo istotnym rozróżnieniem z punktu widzenia analizy porównawczej języka naturalnego i języka programowania i pod jego kątem powinny być oceniane wszystkie kategorie składniowo-semantyczne występujące w języku programowania, a zapożyczone z języka naturalnego.

6. Uwagi końcowe

W trakcie pierwszego etapu pracy wyłonił się cały szereg problemów które wymagają oddzielnej bliższej analizy. Oto ważniejsze z nich.

Wyłonienie kategorii składniowo-semantycznych samego języka programowania nastręcza duże trudności. Spora część języków 4GL powstaje przez szereg lat w drodze ewolucji, w której autorzy języka rozbudowują zarówno warstwę składniową jak i semantyczną (np. wprowadzając nowe paradygmaty programowania), dbając jednocześnie o „kompatybilność" z wcześniejszymi wersjami języka. W ten sposób powstaje twór, w którym istnieje spora ilość rozwiązań obecnych w definicji języka ale nie używanych w praktyce programistycznej w kolejnych wersjach języka, gdyż zostały one „wyparte" nowocześniejszymi rozwiązaniami.

Dokumentacja języka programowania zawiera zazwyczaj wystarczająco bogaty zestaw wypowiedzi w języku naturalnym opisującym właściwości składniowo-semantyczne poszczególnych elementów języka programowania. Opisuje ona jednakże najczęściej zależności pomiędzy „desygnatami wirtualnymi" – a więc zależności wewnętrzne które pozwalają ocenić przyjęte rozwiązanie pod kątem zgodności wewnętrznej języka programowania i języka angielskiego. Niestety materiał wypowiedzi w języku angielskim pozwalający na ocenę pod kątem zgodności zewnętrznej - a więc jak poszczególne elementy języka sztucznego odwzorowują „desygnaty realne" - jest najczęściej bardzo ubogi i zazwyczaj niekompletny. Stąd w aktualnej wersji pracy przyjęto ograniczenie, w którym nie analizuje się problemu zgodności zewnętrznej. Dodatkowo z podzbioru analizowanego języka 4GL wyłączono elementy stanowiące interfejs do innych języków programowania jak np. SQL czy DDE.

Bibliografia

[1]. Kassabgi G., Wood W., Jannery M., Feinstein S., Seidl G.: Using Progress V8, *QUE*, 1996.

[2]. Krzeszowski T.P.: Gramatyka angielska dla Polaków, *PWN*, 1980

[3]. Polański K. (red.): Encyklopedia Językoznawstwa ogólnego, *Osolineum,* 1999, str. 214

[4]. Szczepaniak L., Królikowski Z.: Kontrolowane języki naturalne – przegląd rozwiązań i zastosowań, *Pro Dialog nr 11, str. 47-67, Nakom,* Poznań 2000.

[5]. Szczepaniak L.: Specyfikacja z zastosowaniem ograniczonego języka naturalnego, *II Krajowa Konferencja Inżynierii Oprogramowania KKIO 2000,* Zakopane 2000