AUTOMATYKA • 2010 • Tom 14 • Zeszyt 3/2

Radosław Klimek*, Paweł Skrzyński*, Michał Turek*

Rozszerzenie języka BPQL na potrzeby interaktywnego modelowania i formalnej weryfikacji diagramów aktywności UML

1. Wprowadzenie

Różnorodne rozszerzenia metodyk wspierających wytwarzanie oprogramowania w kierunku tzw. rozwiązań biznesowych są coraz częściej praktykowane. Klasyczny przykład takich działań zakłada wprowadzenie graficznego języka symboli określających encje lub usługi biznesowe. Bardziej zawansowane rozwiązania idą wyłącznie w kierunku wspomagania modelowania procesów biznesowych. Tam podstawowym trendem jest tworzenie reprezentacji graficznych, obrazujących przepływy sterowania w ramach procesu biznesowego. Proces taki (będący usługą) można na etapie projektowania przetwarzać metodami innymi niż jedynie modelowanie graficzne. Prowadzone przez autorów rozległe prace badawcze (których część została przedstawiona w niniejszym artykule) dążą do zdefiniowania mechanizmów wspomagających modelowanie procesów biznesowych – głównie z wykorzystaniem metod formalnych, języków zapytań oraz złożonych procesów konwersji diagramów procesów biznesowych. W trakcie prac zaistniała potrzeba stworzenia języka zapytań, umożliwiającego rozbudowę zadanego formalnie diagramu procesu [1]. Prace badawcze poruszyły zagadnienia automatycznej translacji treści diagramu do postaci zbioru formuł złożonych z symboli procesów elementarnych oraz spójników logicznych, następnie formalnej walidacji poprawności formuł (a tym samym – złożonego procesu biznesowego), finalnie rozbudowy treści diagramu (a tym samym formuł) w trybie interakcyjnym. W niniejszym artykule zostanie przedstawiona propozycja stworzenia języka zapytań, umożliwiającego interaktywną rozbudowę diagramów opisujących procesy biznesowe. Głównym problemem, jaki należy postawić, jest tutaj opracowanie zminimalizowanej puli zapytań umożliwiających wykonywanie wszystkich niezbędnych operacji na diagramie oraz algorytmów realizacji tych zapytań. Utrzymanie więzów integralności zawartych pomiędzy poszczególnymi elementami diagramu determinuje jego poprawność po wykonaniu

* Katedra Automatyki, Akademia Górniczo-Hutnicza w Krakowie

danego zapytania. Razem z diagramem aktywności modyfikowana będzie formuła walidacji tego diagramu. Język zapytań powinien nawiązywać do już utworzonych rozwiązań, umożliwiających projektowanie procesów biznesowych oraz sterowanie nimi. Jednym z nich jest BPQL (*Business Process Query Language*) [2]. Język ten umożliwia zapisanie treści procesu w postaci pseudokodu – z zaangażowaniem zmiennych i prostych instrukcji sterujących. Jest przeznaczony wyrażenia treści poszczególnych procesów elementarnych, oraz w wąskim zakresie – przepływów sterowania pomiędzy nimi. Nie definiuje jednak więzów integralności pomiędzy elementarnymi procesami, które można by wyrazić przy użyciu notacji BPMN na diagramach lub przetwarzać do postaci formuł.

Artykuł proponuje dalsze rozwinięcie koncepcji modelowania przepływu sterowania opartej na kreowaniu tzw. wzorców procesowych. Do podstawowych wzorców procesowych zaliczamy [3]:

- Sekwencję: aktywność w procesie pracy jest wykonywana po zakończeniu aktywności, która ją poprzedza.
- Rozwidlenie: punkt w którym pojedynczy wątek wykonania się rozwidla na kilka wątków wykonywanych równocześnie lub w jakimś porządku.
- Synchronizację: punkt, w którym wiele równoległych wątków łączy się w jeden z założeniem, że każda z nadchodzących gałęzi wykona się dokładnie raz.
- Wybór wyłączny: punkt, w którym jedna lub wiele gałęzi zostaje wybrana do dalszego wykonania.
- Złączenie: punkt, w którym kilka alternatywnych gałęzi się łączy z założeniem, że żadna z równoległych gałęzi nie jest wykonywana równolegle.

Wzorce te nie będą jednak jedynie komponowane graficznie przez projektanta na diagramach BPMN (*Business Process Management Notation*), a potem ewentualnie podawane dalej do procesów transformacji. Będą wykorzystywane do dynamicznej rozbudowy modelu BPMN w oparciu o specjalnie opracowany język zapytań. Utrzymując kontynuację znaczeniową akronimu BPQL, autorzy zdecydowali się na utworzenie warstwy wyższej dla tego języka operującej w systemie zapytań i zarządzającej przepływem sterowania pomiędzy całymi podprocesami definiowanymi przez BPQL. Podejście takie, będąc zdecydowanie bardziej wysoko poziomowym, zakłada zdefiniowanie nowego języka – będącego hybrydą powszechnie znanych SQL (*Structured Query Language*) i BPQL (Języka zapisu dla procesów). Interpretacja komend takiego języka umożliwi edycję już istniejącego modelu przepływów w procesie. Umożliwi to tworzenie języków skryptowych, które obok zakresu DDL (*Data Definition Language*) języków SQL (przeznaczonych do definiowania struktur danych, na których operuje system) będą także w znacznym stopniu przetwarzały jego zachowanie [4].

Podsumowując informacje wstępne, warto zauważyć, że BPQL posiada także swój odpowiednik w notacji XML zwany XPDL (*XML Process Definition Language*) [5]. Wyrażenie BPSQL za pomocą wspomnianego XPDL (będącego już opracowanym językiem) jest celowe, gdyż dostarcza bardziej przenośnej notacji języka. Nie powinno także przysporzyć problemów.

2. Problematyka projektowania języka zapytań wspomagającego edycję modelu sterowania przepływem

Już we wcześniejszych pracach ustalono, że ekstrakcja formuł z modeli BPMN następuje przy założeniu, że model taki jest zbudowany od początku do końca z wzorców procesowych opracowanych dla procesów biznesowych. Udowodniono też, że dla każdego z tych wzorców będzie możliwe wygenerowanie zestawu formuł logiki temporalnej, który specyfikuje własności danego wzorca projektowego. Problem obecnie poruszany to modyfikacja tego wzorca z chwilą wykonania zapytania przetwarzającego model zadany reprezentacją graficzną (BPMN). Każda z tych modyfikacji, będąca realizacją zapytania, musi być możliwa do przeprowadzenia na diagramie BPMN oraz formułach. Ponadto – diagram musi zachować swoją poprawność [6]. Przy realizacji czynności edycyjnych na modelu BPMN potrzebna jest oczywista pula operacji elementarnych, gwarantująca możliwość jego swobodnej edycji. Ustabilizowanie puli koniecznych do implementacji zapytań oraz wyrażenie odpowiadających im procedur realizujących operacje na diagramie BPMN stało się głównym problemem napotkanym podczas prac. Interpreter komend BPSQL musi umożliwić (oprócz definiowania i modyfikacji linii przepływu) także nanoszenie i likwidację pozostałych komponentów notacji (BPMN), generując tym samym poprawną kompozycję wzorców procesowych. Poza walidacją składni samych zapytań BPSQL, konieczne staje się prowadzenie ciągłej kontroli procesu tworzenia modelu przepływów zapisanego w notacji BPMN. Będzie to odpowiednikiem utrzymywania reguł więzów integralności przy interpretacji zapytań języka SQL modyfikującego strukturę bazy danych w SZBD (Systemu zarządzania bazą danych). Następny podrozdział rozliczy szczegółowo te właśnie kwestie.

3. Proponowana metodyka rozwiązania

Język BPSQL powinien wspierać wszystkie czynności edycyjne niezbędne do modyfikowania grafu przepływów pomiędzy procesami elementarnymi wyrażonego za pomocą notacji BPMN. Modelowany proces, rozpoczynając realizację w punkcje startowym, powinien prowadzić do jednego z wielu symboli zakończenia. Przepływ sterownia może być wielokrotnie rozdzielany bez wprowadzania konieczności późniejszego łączenia rozdzielonych linii przepływu (każda może prowadzić także do symbolu zakończenia). Rozdzielenie przepływu możliwe jest w jednym z dwóch wariantów:

- jako rozgałęzienie (alternative) będące w przypadku notacji BPMN wzorcem *Gateway* w wariancie XOR, OR lub Complex),
- jako rozwidlenie czyli tzw. *fork* inicjujący przetwarzanie współbieżne, będące w notacji BPMN wzorcem typu *Gateway* w wariancie *Pararell*.

Po wykonaniu każdego zapytania model procesu biznesowego (tym samym formuła będąca jego odpowiednikiem) muszą być poprawne. Zapytania muszą zatem modyfikować proces tak, aby w każdej chwili możliwe było wykonanie obecnego wariantu procesu lub jego walidacja.

Podstawowym symbolem opisującym czynności elementarne procesu będzie węzeł identyfikujący metodę, gotowy proces lub jakąkolwiek inną aktywność modelowanego systemu w ramach podejmowanej przez niego obsługi biznesowej klientów.

Sam język skryptowy zapisany notacji BPSQL umożliwi tworzenie kodu modyfikującego przepływy sterowania – konstruując tym samym model zachowania systemu. Operacje elementarne związane z realizacją zapytań BPSQL będą realizowane w trybie analogicznym do zapytań SQL – operując na strukturze danych, którą w tym przypadku stanowi zbiór procesów elementarnych oraz wzorców procesowych. Jak łatwo się domyśleć – w podstawowej (obecnie teoretycznej) puli zapytań będzie BPSQL będzie figurowało dodawanie, usuwanie i modyfikacja wzorców projektowych oraz procesów elementarnych prowadzona na diagramie BPSQL. Dodawanie nowych elementów do modelu procesu przy pomocy BPSQL można będzie w pełni zautomatyzować – utrzymując ciągłą poprawność modelu po każdorazowym wykonaniu zapytania dodającego. Istnieją generalnie trzy rodzaje symboli, które należy dodać w sposób jawny:

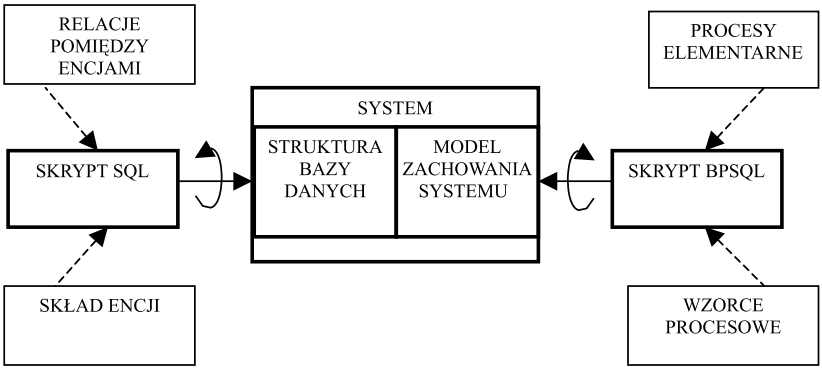
- Węzeł (*node*) będący elementem sekwencji instrukcji.
- Węzeł lub znak zakończenia (stop) będący akcją alternatywną (np. utworzenie rozgałęzienia, ang. *alternative*).
- Węzeł lub znak zakończenia (stop) będący akcją równoległą (np. utworzenie rozwidlenia, ang. *fork*).

Pozostałe symbole (nie licząc tzw. przepływu podstawowego, o którym niżej) towarzyszą wyżej wspomnianym i można będzie generować je automatycznie w trakcie wykonywania zapytań dodających któryś z elementów tej trójki lub zapytań modyfikujących elementy modelu.

Danymi wejściowymi dla skryptów SQL i BPSQL będą kolejno:

- Model encji oraz relacji pomiędzy encjami stanowiący podstawę do projektowania i tworzenia baz danych przyszłego systemu – za pośrednictwem generującego je skryptu SQL.
- Zbiory procesów elementarnych stanowiące materiał do modelowania zachowania systemu przez rozbudowę modelu procesów BPMN – za pośrednictwem skryptu BPSQL operującego na zestawie wzorców procesowych.

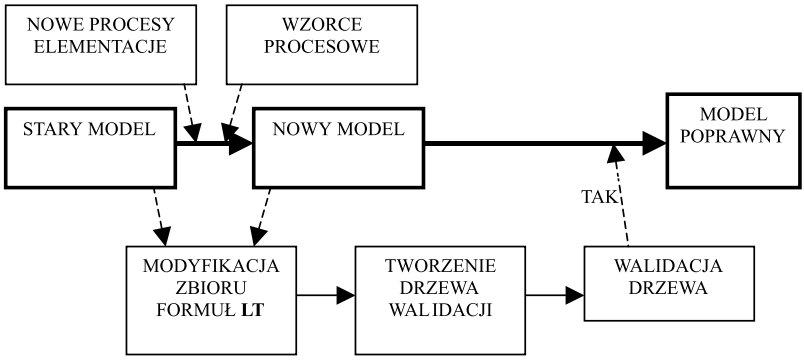
Możliwe będzie także łączenie zapisów obydwu języków skryptowych w jednym zasobie opisującym przyszły system. Schemat prezentujący procedurę modelowania systemu z użyciem skorelowanych skryto przedstawia rysunek 1.



Rys. 1. Skrypty SQL i BPSQL wykonywane w procesie kształtowania struktur danych oraz zachowania modelowanego systemu

4. Zapytania BPSQL

Ustabilizowanie zestawu zapytań języka BPSQL jest tylko pozornie zadaniem prostym. Naturalnie – samo określenie zestawu intuicyjnych operacji elementarnych składowych diagramu w notacji BPMN wygląda na oczywiste. Problem stanowi jednak fakt, że każda z operacji powodowana przez zapytanie BPSQL musi przeprowadzić model procesowy w inny, lecz ciągle poprawny jego wariant. Ogranicza to liczbę potencjalnych zapytań BPSQL jedynie do puli generującej poprawny i spójny model. W artykule wykazano, że taka pula jest możliwa do ustabilizowania i w zupełności wystarczająca – pokrywając wszystkie konieczne czynności edycyjne, które należy wykonać na modelu procesów wyrażonym poprzez BPMN. Techniki walidacji poprawności modelu zostały zaprezentowane na wcześniejszym etapie prac. Zbiory formuł logiki temporalnej, które podlegają walidacji, można wytwarzać każdorazowo po wykonaniu zapytania BPSQL. Można także modyfikować treść formuł w trakcie realizacji zapytania (rys. 2).



Rys. 2. Realizacja zapytania BPSQL na przykładzie zapytania rozbudowującego model

Pierwszym zapytaniem (rozpoczynającym konstrukcję sekwencji przepływów BPMN) jest utworzenie przepływu podstawowego składającego się jedynie z symbolu rozpoczęcia (*Start Event* zgodnie z notacją BPMN) i symbolu stopu połączonych ze sobą linią przepływu. Ewentualne inne symbole stopu (na przykład dla przebiegów alternatywnych procesu) będą dodawane do symboli rozgałęzienia lub rozwidlenia:

ADD BASICFLOW;

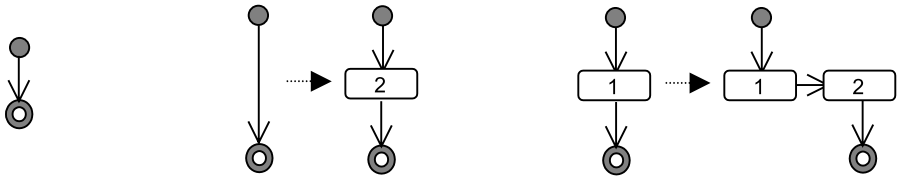
Dodanie elementarnego podprocesu (węzła) odbywa sie za pomocą instrukcji

ADD NODE [node_name] FOR [parent_node_name];

gdzie *parent_node_name* to identyfikator węzła, za którym zostanie umieszczony węzeł właśnie tworzony. Wariant:

ADD NODE [node_name];

posłuży do dodania węzła jako pierwszej operacji po symbolu startowym (rys. 3).



Rys. 3. Reprezentacja BPMN działania zapytań (od lewej):
ADD BASIC FLOW, ADD NODE „1”, ADD NODE „2” FOR „1”

Dodawanie rozdzielenia przepływu w wariancie alternatywy (rozgałęzienia) wymaga wcześniejszego zaplanowania głównego przepływu alternatywnego i odgórnego podjęcia decyzji o sposobie zakończenia tego przepływu. W notacji BPMN jest równoważne z utworzeniem symbolu *Gateway*. Możliwości to zakończenie symbolem stopu lub połączenie go z innym już istniejącym węzłem w celu kontynuacji przetwarzania główną ścieżką:

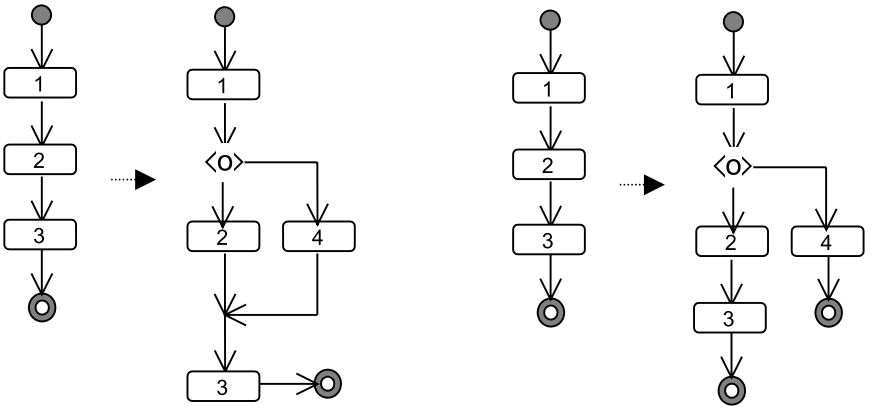
ADD ALTERNATIVE STOP [node_name] FOR [parent_node_name] WHERE [node_constraint];

Zapytanie utworzy symbol stopu i doda nowy symbol decyzji wstawiając go bezpośrednio za węzłem *parent_node_name*. Ponadto utworzy nowy węzeł prowadząc do niego linię przepływu pod warunkiem *node_constrain*. Poprzedni przepływ (także wychodzący z symbolu decyzji) uwarunkuje jako ‘else’. Przepływ z nowoutworzonego węzła *node_name* będzie prowadził do symbolu stop.

W drugim wariancie tworzenia przebiegu alternatywnego posłużyć się można następującym zapytaniem:

ADD ALTERNATIVE [node_name] FOR [parent_node_name] WHERE [node_constraint] CONTINUE [continuation_node_name];

Utworzy ono nowy symbol decyzji wstawiając go bezpośrednio za węzłem *parent_node_name*, utworzy nowy węzeł prowadząc do niego linię przepływu pod warunkiem *node_constraint,* a poprzedni przepływ (także wychodzący z symbolu decyzji) uwarunkuje jako 'else'. Ponadto poprowadzi linię przepływu sterowania od nowo utworzonego węzła *node_name* do *continuation_node_name* (rys. 4).



Rys. 4. Reprezentacja BPMN działania zapytań (od lewej): ADD ALTERNATIVE, ADD ALTERNATIVE STOP

Podobne jak w przypadku symbolu alternatywy zachowanie musi być wymuszane wywołaniem instrukcji dodającej symbol rozwidlenia wprowadzający zrównoleglenie przepływu sterowania. W notacji BPMN jest równoważne z utworzeniem podprocesu (*BPMN Subprocess*). Takie przetwarzanie także może zostać zakończone w jednym z dwóch wariantów: zatrzymaniem wątku (symbol stopu) albo synchronizacją wykonania z innym przepływem. Do utworzenia pierwszego rozwiązania adekwatna będzie instrukcja:

ADD FORK STOP [node_name] FOR [parent_node_name];

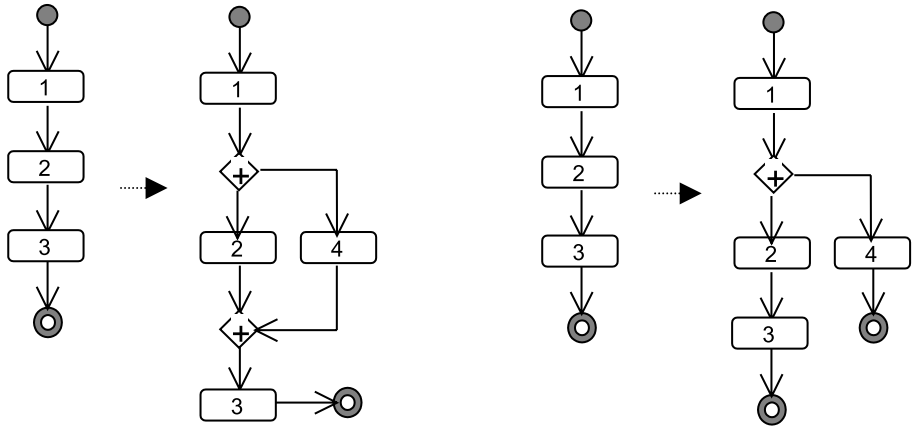
W tym przypadku istniejąca linia przepływu prowadząca od węzła *parent_node_name* do następnego po nim symbolu zostanie zastąpiona przez symbol rozwidlenia (*fork*). Na drodze pomiędzy symbolem rozwidlenia a nowym symbolem stopu umieszczony zostanie nowy węzeł *node_name*. Linia przepływu będzie także prowadzić od symbolu rozwidlenia do symbolu uprzednio następnego po *parent_node_name* (rys. 5).

Wariant zapytania umożliwiający wprowadzenie późniejszej synchronizacji lub zakończenia równoległego przepływu będzie wyglądał następująco:

ADD FORK [node_name] FOR [parent_node_name] JOIN [join_node_name];

W tym przypadku do nowo utworzonego symbolu *join* (synchronizacji) będzie prowadził przepływ wychodzący uprzednio z węzła *join_node_name*. Proces tworzenia symbolu *fork* będzie analogiczny jak w poprzedniej sytuacji – z tą różnicą, że na linii

przepływu prowadzącej od symbolu *fork* do *join* zostanie utworzony nowy węzeł o *node_name* (rys. 5). Przepływ uprzednio wychodzący do symbolu następnego po *join_node_name* będzie nadal prowadził do tego symbolu, jednak także z nowo utworzonego *node_name* (kontynuacja sterowania po synchronizacji).



Rys. 5. Reprezentacja BPMN działania zapytań (od lewej): ADD FORK, ADD FORK STOP

Warto zauważyć, że obydwa warianty instrukcji dodawania wzorców projektowych umożliwiają tworzenie także odpowiedników pętli programowych.

Wykluczone jest tworzenie linii przepływu, które wychodzą jednocześnie z tego samego węzła. Nie jest także możliwe prowadzenie przepływu do symbolu *fork* i *alternative* jednocześnie z tego samego węzła. Próby tworzenia takich konstrukcji przy użyciu zapytań ADD FORK i ADD ALTERNATIVE muszą być wykrywane.

Czynności edycyjne dla już istniejących struktur mogą polegać głównie na zlecaniu usunięcia komponentów wadliwych i zastępowaniu ich nowymi. Należy pamiętać, że zgodnie z postawionymi założeniami symbole węzłów są jedynie referencjami do procesów, więc brak rozbudowanej funkcjonalności w zakresie modyfikacji istniejących już węzłów nie będzie kłopotliwy (funkcja taka odpowiada usunięciu i utworzeniu nowego, skorygowanego egzemplarza węzła). Usuwanie może dotyczyć jedynie węzłów, gdyż eliminację pozostałych komponentów modelu można prowadzić automatycznie. Redukcja przepływu rozdzielonego do ostatniego węzła i następnie usunięcie tego węzła powodują, że przepływ ten traci rację bytu (jest pusty). Ulega wówczas likwidacji. Warto zaznaczyć, że likwidacja będzie mogła dotyczyć obydwu wariantów przepływu (rozgałęzienia i rozwidlenia). Nie będzie także uzależniona od istnienia symbolu stopu na końcu przepływu (zamiast połączenia z innym).

Zapytanie usuwające węzeł będzie miało najbardziej z możliwych prozaiczną postać:

DELETE [node_name];

gdzie likwidacja węzła *node_name* może w konsekwencji pociągnąć wyżej opisane akcje. Zapytania modyfikujące już istniejące procesy sprowadzają się do określania wymiany procesu elementarnego na inny:

UPDATE [node_name] SET [new_node_name]

Wszystkie przedstawione wyżej zapytania operują wyłącznie na przestrzeni nazw węzłów będących procesami elementarnymi. Nowe przepływy powstają na skutek sekwencyjnego dodawania węzłów. Likwidacja przepływów także występuje w konsekwencji usuwania wytypowanych węzłów (likwidacja ostatniego węzła na trasie danego przepływu skutkuje likwidacją całego przepływu).

Redukcja liczby zapytań BPSQL do tak wąskiego zbioru możliwości ma kluczowe znaczenie przy prowadzeniu konwersji BPMN do zbiorów formuł logi temporalnej – będącej przedmiotem wcześniejszych prac. Wszystkie przypadki modyfikacji linii przepływu sterowania oraz węzłów będą wiązały się w modyfikacjami jedynie pojedynczych formuł zbioru. Formuły nie muszą być rozbijane, gdyż w większości przypadków modyfikacja polega na substytucji termu lub dodania do formuły wyrażenia łączonego z wcześniejszą jej treścią spójnikami logicznymi. Ma to kluczowe znaczenie dla prowadzenia procesu walidacji poprawności modelu BPMN po wykonaniu zapytania BPSQL.

5. Podsumowanie

Języki realizujące zapytania użytkownika w trybie interaktywnym są niewątpliwie podstawą większości interfejsów komunikacyjnych mających na celu dostarczenie pomostu do łatwej rozbudowy edytowanych treści. W przypadku zastosowań wobec modelowania przepływu sterowania wymagają wariantów silnie zmodyfikowanych jednak wyrażają jednocześnie podejście innowacyjne. W połączeniu w klasycznie stosowanymi językami modyfikującymi strukturę danych systemu tworzą hybrydę metod projektowych umożliwiającą jednoczesną modyfikację zarówno danych, jak i zachowania systemu wpływającego później na treść owych danych. Wyrażenie czynności edycyjnych w postaci skryptu otwiera drogę do daleko idących automatyzacji procesów modelowania systemów biznesowych, umożliwiając projektowanie narzędzi. Chodzi tu głównie o interpretery skryptów, które bazując w przypadku SQL i BPSQL na tej samej logice realizacji zapisów skryptu, są w stanie współbieżnie rozwijać struktury danych oraz procedury przetwarzające te dane. Natychmiastowa walidacja poprawności diagramu BPMN po przeprowadzeniu go do postaci zestawu formuł logiki temporalnej umożliwia otrzymanie rozwiązania analogicznego do walidatora więzów integralności występujących pomiędzy tabelami bazy danych.

Wykorzystanie walidatora (podobnie jak podczas realizacji zapytań SQL) będzie możliwe każdorazowo podczas realizacji zapytań BPSQL, gwarantując poprawność diagramu przez cały czas jego rozbudowy.

Literatura

[1] Momotko M., Subieta K., *Process Query Language – A Way to Make Workflow Process more Flexible*. Advences in Databases and Information Systems, Springer, 2004, 313–321.
[2] Business Process Management Initiative: Business Process Query Language Web Page: http:// www. bpmi.org/bpql.esp.
[3] IBM Developer Works: Business Process Execution Language for Web Services, 2002.
[4] Subieta K., *Theory and Construction of Object-Oriented Query Languages*. Polish-Japanese Institute of Information & Technology, 2004.
[5] Workflow Management Coalition: Workflow Process Definition Language – XML process definition language, WfMC-TC-1025, 2002.
[6] Booch G., Rumbaugh J., Jacobson I., *The Unified Modeling Language User Guide*. Addison-Wesley, 1998.