STUDIA INFORMATICA 2011
Volume 32 Number 2A (96)

Michał WIDERA
Streams Lab

INTEGRACJA STRUMIENIOWEGO I RELACYJNEGO SYSTEMU ZARZĄDZANIA DANYMI

Streszczenie. W takcie prac nad strumieniowym systemem zarządzania danymi, przeznaczonym dla potrzeb przetwarzania sygnałów, pojawiła się potrzeba złączenia przetwarzanych danych z danymi przechowywanymi w systemie relacyjnym. W artykule opisano przyjęte rozwiązanie oraz opracowane metody formułowania i realizacji zapytań opartych na zbiorze relacji i strumieni.

Słowa kluczowe: strumieniowe bazy danych, języki zapytań, konstrukcja techniczna systemów baz danych, przetwarzanie danych

INTEGRATION OF STREAM AND RELATIONAL DATA MANAGEMENT SYSTEM

Summary. Working on data stream management system for digital signal processing purposes a need for stream and relational data connection was appear. This paper present adopted solution and developed methods of creating queries on streams and relations.

Keywords: data stream management systems, query languages, database construction, data processing

1. Wstęp

Sposobem komunikacji pomiędzy systemem zarządzania danymi a użytkownikiem jest wybrana forma języka zapytań. W przypadku systemów relacyjnych, zagadnienie zostało szczegółowo przebadane i opisane w literaturze. Nadal jednak interesujące problemy badawcze, związane z językami zapytań, pojawiają się w przypadku systemów opartych na alternatywnych modelach danych.

Analizując początkowe prace badawcze, związane ze strumieniowymi systemami zarządzania danymi w aspekcie rozwoju języków zapytań, można odkryć dwa równoległe kierunki badań. Naukowcy związani z projektem rozwijanym na uczelni Stanford [1,2] skupili się na rozszerzeniu istniejącego języka zapytań, natomiast grupa związana z uczelnią MIT skupiła się na rozwiązaniu graficznym [3]. Oba kierunki badań dały w efekcie początek wielu rozwiązań komercyjnych i badawczych.

Komercyjne firmy oferując strumieniowe systemy zarządzania danymi posługują się akronimem CEP, oznaczającym system przeznaczony do złożonego przetwarzania zdarzeń (ang. *Complex Event Processing*). Na uwagę zasługują następujące systemy – Oracle CEP, Sybase CEP oraz StreamBase. Rozwiązanie firmy Oracle bazuje na wynikach badań opracowanych na uczeni Stanford, rozwiązanie StreamBase stanowi komercjalizację wyników badań opracowanych na uczelni MIT, natomiast rozwiązanie Sybase CEP stanowi efekt syntezy wyników obu ścieżek badawczych. W wyniku prac wymienionych firm i związanych z nimi zespołów badawczych powstały unikalne rozwiązania, które jednak zupełnie inaczej interpretują i przetwarzają napływające dane.

Obecnie w literaturze naukowej zauważa się potrzebę sformalizowania i uszeregowania oferowanej semantyki. Pierwszą próbę przedstawiono w pracy badawczej, opracowanej przez zespoły związane z uczelniami Stanford i MIT [4]. Jednak już w następnych latach wskazano na ograniczenia formalne zaprezentowanego modelu klasyfikacji, przedstawiając uzupełnienie w postaci bardziej ogólnego modelu [5]. Zaprezentowany model miał na celu ujednolicenie klasyfikacji systemów przetwarzania strumieni danych, a w szczególności uwzględnienie funkcjonalności osiągniętej w produkcie firmy Sybase.

W pracach dotyczących tej tematyki powszechnie jest używany akronim SPE, określający typ mechanizmu przetwarzania strumieni danych (ang. *Stream Processing Engine*). Na podstawie przedstawionego modelu powinna istnieć możliwość identyfikacji SPE, zastosowanego w danym CEP. Co więcej, użycie danego SPE powinno dać możliwość odtworzenia odpowiedzi systemu CEP w innym produkcie. Poszukiwania tego typu rozwiązań stanowią obecnie jedną z gałęzi prowadzonych badań w zakresie strumieniowych systemów zarządzania danymi [5].

1.1. Rozszerzony model strumienia danych

Zaprezentowane wyniki poszukiwań ogólnego modelu klasyfikacji strumieniowych systemów zarządzania danymi dały w efekcie rozszerzony model strumienia danych. W ramach modelu przedstawiono 6 szczegółowych definicji pojęć, wymaganych do klasyfikacji zachowania SPE [5].

Definicja 1 (Wymiar czasu). Wymiar czasu T jest dyskretnym, liniowo uporządkowanym, przeliczalnym i nieskończonym zbiorem elementów $t \in T$. Wymiar czasu T jest ograniczony z dołu.

Definicja 2 (Strumień). Strumień S jest przeliczalnym i nieskończonym zbiorem elementów $s \in S$. Każdy element strumienia S zawiera krotkę v zgodną ze schematem, czas aplikacji $t_{app} \in T$. Czas systemowy $t_{sys} \in T$ oraz identyfikator pakietu bid $\in N$.

Do uzupełnienia pełnego modelu, opisującego SPE użyto też kilku dodatkowych definicji. Zdefiniowano pojęcie Pakietu (ang. *Batch*), rozumianego jako skończony podzbiór elementów danego strumienia S, gdzie dla każdego elementu $b \in B$, t_{app} jest takie samo. Dodatkowo, każdy element b zawiera unikalny bid $\in N$. Ściśle określono również pojęcia Okna (ang. *Windows*), Okna w czasie (ang. *Time-based Window*) oraz Przesunięcia i Rozmiaru okna (ang. *Window Size and Slide*).

Na podstawie tak przedstawionego modelu strumienia danych, w pracy [5] opisano poszczególne różnice semantyczne pomiędzy komercyjnymi systemami CEP. Należy jednak podkreślić, że autorzy w jednym z ostatnich zdań swojej publikacji przyznają, że przedstawiony model może nie mieć zastosowania w przypadku identyfikacji alternatywnych systemów SPE.

Kwestia efektywnego i spójnego sposobu przetwarzania danych zarządzanych przez system relacyjny wraz z danymi napływającymi w strumieniach danych stanowi problem badawczy. Z uwagi na fakt, że w każdym CEP zastosowano inny SPE, problem złączenia danych zawartych w zbiorze strumieni danych oraz w zbiorze relacji musi zostać rozwiązany za każdym razem w inny sposób.

1.2. Rozwiązanie zastosowane w projekcie STREAM/Oracle

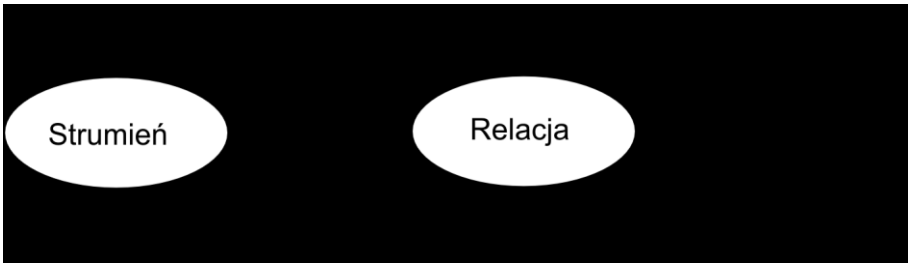
W ramach prac badawczych nad systemem STREAM opracowano język zapytań CQL (ang. *Continious Query Language*) [1]. W początkowej fazie badań przyjęto, że proste, ciągłe zapytania mogą zostać przedstawione za pomocą zwykłego języka SQL. W założeniach zastąpiono odwołania do relacji, odwołaniami do strumieni danych. Jednak w trakcie dalszych badań bardzo szybko skorygowano to błędne założenie. Ciągłe zapytania stawały się coraz bardziej skomplikowane, zawierały dodatkowe operacje agregacji, podzapytania, konstrukcje oparte na oknach danych oraz trudno definiowalne złączenia. Zastosowanie zwykłej algebry relacji i języka SQL stawało się coraz bardziej problematyczne [1].

Dlatego, w celu formalnego przedstawienia operacji na strumieniach i relacjach przedstawiono następujące definicje strumieni i relacji:

Definicja 3 (Strumień). Strumieniem S nazywamy zbiór par <s,t>, gdzie s stanowi krotkę, a t znacznik czasu, określający czas przybycia krotki s do strumienia S.

Definicja 4 (Relacja). Relacją R nazywamy zbiór krotek. Zbiór krotek jest oznaczony w danym czasie t. Przez R(t) rozumiemy instancję relacji R w czasie t.

O ile definicja 3. nie budzi zastrzeżeń i została przyjęta w dalszych pracach badawczych, o tyle przedstawiona w pracy definicja 4, opisująca relację R, znacząco różni się od znanej i stosowanej definicji z algebry relacji. Było to konieczne w celu uproszczenia zapisu wielu zależności, związanych z mechanizmem transakcyjnym, a co za tym idzie z integracją modelu relacyjnego ze strumieniowym.



Rys. 1. Typy danych i semantyka wprowadzonych operatorów (Projekt STREAM)
Fig. 1. Data types and operators semantic (STREAM Project)

Na podstawie operatorów przedstawionych na rys. 1 ukazano wiele formalnych operacji na relacjach i strumieniach danych. Brakujący zbiór operatorów przekształcających strumień danych w inny strumień danych jest realizowany poprzez złożenie funkcji operatorów strumień-w-relację oraz relacja-w-strumień. Przyjęto, że operatory relacja-w-relację pokrywają się z operatorami algebry relacji.

1.2.1. Operator strumień-w-relację

Przykład zastosowania operatorów można przedstawić na podstawie zapytania mającego za zadanie przedstawienie identyfikatorów pojazdów pojawiających się w ramach pewnej relacji w czasie ostatnich 20 sekund.

```
Select Distinct identyfikator_pojazdu
From WybranyStrumien [Range 20 Seconds]
```

W ramach planu realizacji tego zapytania twórcy języka deklarują zastosowanie złożenia operatorów strumień-w-relację, opartego na ruchomym oknie danych [2], poprzedzonego operatorami typu relacja-w-relację, realizujących typową operację projekcji oraz eliminacji powtórzeń. W wyniku realizacji planu tego zapytania tworzona jest relacja. Można przyjąć [4], że wspominane operatory strumień-w-relację zostały oparte na ruchomym oknie danych.

1.2.2. Operator relacja-w-strumień

Poszczególne SPE różnią się m.in. w sposobie działania operatorów typu relacja-w-strumień [4,5]. W przypadku systemu STREAM, w ramach tej klasy operatorów zdefiniowano następujące grupy operatorów: Istream, Dstream, Rstream. Zastosowanie operatora Istream (ang. *Insert stream*) na relacji R tworzy strumień z elementów, które do danej relacji

zostały dołączone w zadanym okresie. Analogicznie operator Dstream (ang. *Delete stream*) tworzy strumień z elementów, które zostały usunięte w danym okresie. Dopełnieniem zbioru przedstawionych operatorów jest Rstream (ang. *Relation stream*), określający strumień elementów znajdujących się w relacji w danej chwili t. Zastosowanie praktyczne w planach realizacji zapytań mają operatory Istream i Dstream, aczkolwiek oba te operatory mogą zostać wyrażone za pomocą operatora Rstream, wraz z operatorami zdefiniowanymi na ruchomym oknie danych.

Przykład zastosowania w planie realizacji zapytania operatora Istream przedstawia się następująco:

```
Select Istream(*)
From  WybranyStrumien [Range Unbounded]
Where szybkość > 50
```

Opracowany język CQL oparto na rozszerzeniu standardu SQL:1999. Język CQL doczekał się implementacji w rozwiązaniu firmy Oracle. Adaptacje przedstawionych założeń można znaleźć w publikacjach firmowych [6]. Praktyczny przykład zapytania, zapisany w języku stworzonym na potrzeby produktu Oracle CEP, przedstawia się następująco:

```
CREATE QUERY Q AS SELECT identyfikator_pojazdu
FROM  WybranyStrumien [RANGE 20 SECONDS]
```

Definicje odpowiednich operatorów Istream oraz Dstream pojawiły się również w okresie późniejszym, w pracach związanych z systemem StreamBase oraz językiem StreamSQL.

1.3. Rozwiązanie zastosowane w projekcie StreamBase

Projekt StreamBase jest projektem powstałym w wyniku komercjalizacji badań nad systemami Borealis oraz Aurora. System został opracowany m.in. na uczelni MIT. W początkowej fazie badań nad CEP rozważano jedynie graficzny sposób zapisu zapytań. Zapytania składano z predefiniowanych operatorów tworząc plan realizacji zapytania. Z tego powodu, w początkowej fazie rozwoju było bardzo trudno porównać systemy rozwijane w ośrodkach Stanford i MIT. Pierwsze prace badawcze, mające na celu porównanie obu systemów, polegały na stworzeniu testu – Liniowego Testu Drogowego (ang. *Linear Road Benchmark*) [7]. Test miał na celu wykazanie użyteczności strumieniowego systemu zarządzania danymi. W trakcie dalszych prac nad systemem StreamBase przedstawiono język StreamSQL, w ramach którego wprowadzono implementację operatorów znanych z języka CQL. Przykład ciągłego zapytania wyrażony w języku StreamSQL przedstawia się następująco [8]:

```
SELECT * FROM StrumienWejsciowy
  WHERE danePole >= 3 && danePole <= 6
  INTO StrumienWyjsciowy;
```

Podstawowa różnica pomiędzy rozwiązaniem zastosowanym przez firmę Oracle a rozwiązaniem zastosowanym w systemie StreamBase polega na tym, że w pierwszym przypadku używane są operacje oparte na przepływie czasu, a w drugim na przepływie danych [4].

1.4. Sybase CEP

Większość dużych producentów oprogramowania (m.in. Microsoft, IBM) podejmuje próby stworzenia oprogramowania w ramach swoich prac lub w kooperacji z ośrodkami badawczymi (np. Oracle). W międzyczasie, w literaturze naukowej pojawiło się bardzo wiele odnośników do materiałów zamieszczonych przez firmę Coral8. Po połączeniu firm Aleri oraz Coral8 i zaprezentowaniu swojego CEP, firma Sybase przejęła to rozwiązanie udoskonalając je pod swoją marką. Niestety wraz z tym zdarzeniem zawarte w literaturze naukowej odnośniki utraciły ważność. Zaprezentowane niedawno przez Sybase oficjalne rozwiązanie [9] stanowi bardzo ciekawą odpowiedź na rozwiązanie zaprezentowane przez Oracle i StreamBase. W przedstawionym rozwiązaniu połączenie DBMS z SEP wynika z naturalnego rozwoju systemu. Zaprezentowane rozwiązanie zapewnia graficzny i tekstowy interfejs użytkownika, jest przemyślane semantycznie, a zaprezentowane na stronach internetowych przykłady użycia pokrywają obszar większości typowych zastosowań systemów CEP, opisanych w literaturze naukowej. Formalny mechanizm służący do porównań i integracji powyższych systemów CEP przedstawiono w pracy [5].

Jako rozwiązanie problemu połączenia danych relacyjnych i strumieniowych można przedstawić przykład wzorowany na materiałach szkoleniowych Sybase. W ramach przykładowego zadania, którego celem jest okresowe wypełnianie relacji danymi pojawiającymi się w wyniku zachodzących zdarzeń po stronie systemu strumieniowego zaproponowano następującą formę zapytania:

```
Execute Statement Database „TickDatabase”
[[
  insert into TabelaRelacyjna(ts, Wartosc)
  values (?TS, ?Wartosc);
]]
Select NOW() as TS
       Wartosc as Wartosc
From StrumienDanych;
```

Analizując inne przykłady, umieszczone na stronie producenta, natrafimy m.in. na ciekawy przykład implementacji FSM w języku zapytań.

1.5. Otwarte problemy badawcze

Nadal pozostaje bardzo wiele otwartych problemów badawczych. W krótkim przeglądzie aktualnego stanu wiedzy przytoczono tylko kilka podstawowych systemów. Na uwagę zasłu-

gują jednak jeszcze systemy: TelegraphCQ, który postał jako rozszerzenie systemu PostgreSQL [10] oraz jego przykład komercjalizacji – system Truviso [11,12]. W przypadku systemu DejaVu [13] rozszerzono system MySQL o możliwość przetwarzania strumieni danych za pomocą systemu rozszerzeń (ang. Plugin).

Jednym z otwartych problemów badawczych jest integracja kilku SPE w jednym działającym systemie. Problemem jest brak jednoznacznej semantyki opracowanych modeli strumieniowych systemów zarządzania danymi. Brakuje jednoznacznej definicji tak podstawowych pojęć, jak strumień czy okno. Jednak wyniki ostatnio opublikowanych prac badawczych [4,5], związanych z przedstawionymi we wstępie systemami, dają nadzieję na rozwiązanie tego problemu, przynajmniej w zakresie tych systemów zarządzania danymi.

Kolejnym otwartym problemem badawczym jest rozwiązanie problemów optymalizacyjnych w przypadku integracji SPE-SPE oraz SPE-DBMS. Przez DBMS rozumiemy (w tym przypadku) zarówno relacyjny system zarządzania danymi, jak i sam mechanizm przetwarzania relacji. Okazuje się, że w pracach badawczych poszukiwania skupiły się głównie na stworzeniu rozwiązań autonomicznych. Na dalszy plan zeszły problemy związane z optymalizacją procesów w przypadku integracji systemów.

W obszarze badań pominięto problemy związane z definicją transakcji. Wyniki tego typu badań są warunkiem precyzyjnego określenia modelu integracji SPE-SPE oraz SPE-DBMS. Bardzo szczątkowo poruszono – związane z tym zagadnieniem – problemy dotyczące algorytmów tolerowania błędów oraz predykcji odpowiedzi. Niektóre z analizowanych przypadków dotyczyły scenariuszy odtwarzania odpowiedzi oraz minimalizowania tego czasu dla systemu.

2. Strumieniowy system do przetwarzania sygnałów

W ramach prowadzonych od kilku lat prac badawczych [14,15] tworzony jest strumieniowy system zarządzania danymi, przeznaczony do przetwarzania sygnałów i realizacji algorytmów DSP w systemie zarządzania danymi. Potrzeby twórców systemów monitorujących przeznaczonych do przetwarzania sygnałów są różne w stosunku do oferowanej funkcjonalności komercyjnych systemów CEP. W początkowym etapie prac nie rozważano połączenia SPE-DBMS jako istotnego problemu badawczego. Jednak na dalszym etapie pracy pojawiła się taka potrzeba.

W ramach konstrukcji istniejącego systemu stworzono dedykowaną algebrę i autonomiczny parser języka zapytań, umożliwiający zapis typowych filtrów sygnałowych. Stworzony język zapytań przypomina język SQL, jednak nie jest z nim zgodny. Stworzone rozwiązanie wymagało zaprezentowania własnego sposobu połączenia systemu DBMS z opracowanym SPE.

W pracach prowadzonych w ramach projektu STREAM przyjęto, że operacje strumień-w-strumień będą realizowane przez złożenie operacji strumień-w-relację i relacja-w-strumień. Tego typu założenie niestety nie może zostać przyjęte w opracowanym systemie. W uproszczeniu można stwierdzić, że taki stan wynika z faktu, iż opracowana algebra opiera się na twierdzeniach teorii liczb, a algebra relacji jest oparta na teorii zbiorów.

Przedstawione w pierwszym rozdziale różne definicje strumieni danych oraz wnioski zawarte w pracach badawczych [5] wskazują na istnienie nierozwiązanych problemów badawczych. Od początku prac nad tworzonym systemem [14] stosowana jest odmienna od przyjętej w literaturze definicja strumienia danych. Przyjęty model strumienia oznaczono jako model różnicowy.

Definicja 5. (Strumień). Strumień S jest parą uporządkowaną (s_n,Δ_n), której pierwszym elementem jest ciąg krotek $s \in S$, a drugim ciąg wartości wyznaczających ciąg różnic czasu pomiędzy kolejnymi krotkami. Identyfikator kolejnych krotek $n \in N$.

W ramach przedstawionego rozwiązania nie występuje problem równoczesnych zdarzeń. Nawet jeśli w wyniku zastosowania operatorów stworzonej algebry powstają w strumieniu krotki o tym samym czasie zdarzenia, to poprzez zastosowanie kolejnych operatorów jesteśmy w stanie rozróżnić, które zdarzenie powstało, w którym strumieniu.

2.1. Model połączenia SPE-DBMS w opracowanym rozwiązaniu

Jak wspomniano powyżej, przyjęty model złączenia relacji i strumieni danych w przypadku tworzonego rozwiązania musi zostać gruntownie zmodyfikowany. W trakcie tworzenia fizycznego rozwiązania systemy strumieniowy i relacyjny zostały odseparowane od siebie warstwą interfejsu. Systemy relacyjny i strumieniowy w proponowanym rozwiązaniu dysponują autonomicznymi parserami zapytań.

Rys. 2. Typy danych i semantyka wprowadzonych operatorów w tworzonym systemie
Fig. 2. Data types and available operators in created system

Na rys. 2 widać, że operatory typu strumień-w-strumień istnieją w modelu jawnie. Operatory te są realizowane w systemie strumieniowym. Operatory relacja-w-relację realizowane są w systemie relacyjnym. Połączenie i synchronizację zapewniają operatory typu strumień-w-relację, znajdują się one pod kontrolą obu tych systemów.

Jak widać w modelu zrezygnowano z operatorów relacja-w-strumień. Decyzja ta została przemyślana i podjęta w wyniku analizy potrzeb projektowych, występujących w typowych systemach monitorujących sygnały. Zdarzenia w systemie relacyjnym zachodzą asynchro-

nicznie. Transakcje realizowane w takim systemie podlegają niedeterministycznemu reżimowi czasowemu. System monitorujący, „pragnący" przedstawić wyniki i aktualny stan monitorowanych obiektów może bez problemu pobrać dane z systemu strumieniowego bądź z systemu relacyjnego. Okazuje się, że dostęp do danych relacyjnych w systemie strumieniowym jest dla takiego rozwiązania zbędny. Jeśli już musi zajść jakaś forma integracji danych w tym kierunku – może ona zajść na poziomie aplikacji i nie istnieje potrzeba tworzenia kolejnej warstwy abstrakcji lub definiowania interfejsu, koniecznego do stworzenia tego typu funkcjonalności. Zaproponowana metoda rozwiązania tego typu problemu nabiera znaczenia, w przypadku zastosowania systemu wbudowanego, w którym ograniczone zasoby stanowią jeden z czynników projektowych.

Na uwagę zasługuje również – poruszony w pracach badawczych [16] – aspekt definicji transakcji opartych na strumieniach danych. W trakcie prac na rozwijanym systemem, problem ten poruszono już w 2004 roku i przedstawiono [14] wstępne wyniki badań, w ramach pracy związanej z tworzeniem systemu medycznego. Należy podkreślić, że przedstawione wyniki od kilku już lat stanowią podstawę tworzonego systemu umożliwiając połączenie SPE-DBMS.

2.2. Techniczne aspekty opracowanego połączenia

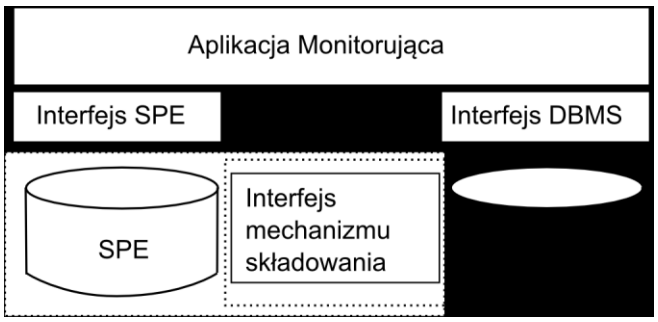
Podobnie jak w przypadku systemu DejaVu – przyjęto rozwiązanie polegające na integracji tworzonego systemu z MySQL za pomocą mechanizmu wtyczek. Mechanizm włączania do systemu dodatkowych źródeł danych został opisany w literaturze technicznej [17]. Połączenie polega na stworzeniu kodu przedstawiającego w przestrzeni tabel systemu MySQL zbioru tabel. Struktura oznaczona w systemie jako Handletron jest połączeniem implementacji wzorca Singleton wraz z modułem obsługi tabel (ang. *Handler*). Większość funkcji struktury handletron dotyczy obsługi transakcji.

Możliwość rozszerzania systemu MySQL o własne mechanizmy składowania w postaci interfejsu abstrakcyjnego pojawiła się podczas przejścia z wersji 3.22 na 3.23. Ta funkcjonalność pełniła kluczową rolę w czasie integracji mechanizmu składowania InnoDB – dającego zbiór niezawodnych funkcji transakcyjnych. Możliwość obsługi tabel poprzez własny mechanizm składowania pozwala uzyskać funkcjonalność obsługi tabel dla dowolnego formatu zapisu, w którym zdefiniowano metody odczytu i zapisu rekordów. W ramach wspomnianej literatury technicznej opisano sposoby integracji mechanizmów składowania dla wersji 4.1 i 5.1.

W przypadku opracowanego systemu przygotowano mechanizm udostępniania bieżących fragmentów strumieni danych, uzyskanych na podstawie implementacji metody ruchomych okien danych. Jako, że dane udostępniane były tylko w jednym kierunku, implementując zbiór operatorów strumień-w-relację należało jedynie zapewnić spójność prezentowanych danych przez SPE. Realizacja dowolnej operacji odczytu w systemie relacyjnym wymagała

uchwycenia i przytrzymania okna danych na czas realizacji transakcji. Jako, że operacja przytrzymania okna – w tym wypadku już z danymi archiwalnymi – odbywa się w ramach wątku struktury Handletron, autonomiczny system SPE może bez zakłóceń dalej realizować proces przetwarzania strumieni danych.

Problemem pozostaje dobór danych bieżących i usuwanie danych archiwalnych. W przypadku pesymistycznego scenariusza można sobie wyobrazić sytuację, w której system SPE będzie generował na tyle szybko kolejne okna danych, że wątek struktury Handletron wyczerpie zasoby pamięci systemu. Dlatego w trakcie definiowania sposobu komunikacji SPE-DBMS konieczne jest również zdefiniowanie sposobu archiwizacji i cyrkulacji danych przechowywanych w zrzutach okien czasowych.



Rys. 3. Architektura systemu monitorującego
Fig. 3. Monitoring system structure

W praktyce aplikacja zapewniająca wizualizację procesu monitorowania realizuje dostęp do danych poprzez interfejsy baz danych. Na rys. 3 przedstawiono schematyczną architekturę takiego systemu. Dostęp do danych relacyjnych jest możliwy poprzez zastosowanie biblioteki SOCI. W kodzie aplikacji monitorującej zapisanej w języku C++ można znaleźć bezpośrednie odwołania do tabel za pomocą osadzonego języka SQL. Dodatkowo, jeśli występuje konieczność złączenia w zapytaniu danych zawartych w tabelach z danymi relacyjnymi, to poprzez interfejs mechanizmu składowania można otrzymać odpowiedź z systemu strumieniowego. System DBMS w takiej architekturze nie musi zasilać danymi systemu SPE. Zmiany w zbiorze relacji nie stanowią analizowanego strumienia danych w systemie monitorującym. Ciężar procesu monitorowania spada na system SPE.

Aplikacja monitorująca umieszcza w systemie SPE ciągłe zapytania i pobiera dane z systemu SPE w trybie ciągłym za pomocą Interfejsu SPE. W wyniku tego, że komunikacja z tym systemem nie odbywa się na zasadzie pytanie-odpowiedź, a raczej na zasadzie pytanie-odpowiedź, odpowiedź itd…, to zarejestrowane zapytanie tworzy obszar aktualizowany po stronie interfejsu na bieżąco, wraz z napływem danych. W tym celu został stworzony alternatywny w stosunku do SOCI interfejs komunikacji z SPE.

3. Wnioski i podsumowanie

Prowadzone prace mające na celu przedstawienie sposobu połączenia danych w systemach relacyjnych i strumieniowych są obecnie intensywnie rozwijanym obszarem badawczym. Poszukiwane są sposoby integracji różnych systemów strumieniowych pomiędzy sobą, jak również integracji systemów strumieniowych i relacyjnych [5].

W ramach prowadzonych prac badawczych, w artykule przedstawiono rozwijany mechanizm wymiany danych pomiędzy opracowanym systemem strumieniowym i relacyjną bazą danych. Różnice, które powstały już na początku prac badawczych dają w efekcie odmienne wyniki i inne problemy badawcze w stosunku do głównego nurtu badań. Specyfika potrzeb stawianych przed strumieniowym systemem zarządzania danymi, przeznaczonym do przetwarzania sygnałów uniemożliwia zastosowanie rozwiązań ogólnych. Usunięcie operatorów relacja-w-strumień i wprowadzenie operatorów strumień-w-strumień stanowi istotną różnicę w stosunku do obecnie ustalonych kierunków badań. Wprowadzona modyfikacja modelu stanowi alternatywny kierunek badań i stwarza nowe możliwości w zakresie poszukiwań rozwiązania dedykowanego.

Już kilka lat temu w publikacjach związanych z pracą nad tym systemem wskazano na konieczność zdefiniowania modelu transakcji na strumieniach danych. W pracy [16] przedstawiono propozycję rozwiązania tego problemu. Jako ciekawostkę można również podać fakt, że w pracach [14,15] opublikowanych w latach 2005-2006 przedstawiono – w ramach opracowanej algebry – operatory realizujące funkcję operacji SPREAD [4], której celem jest zmiana postaci strumienia danych. Wprowadzenie tego operatora stanowi jeden z podstawowych elementów pomocnych w procesie integracji strumieniowych i relacyjnych systemów baz danych.

Analizując obecnie publikowane prace badawcze można stwierdzić, że kierunek badań mający na celu przedstawienie formalnej semantyki łączenia systemów baz danych, oraz przedstawienie rozwiązania unifikującego metody dostępu do danych i przetwarzania strumieni danych jest jednym z aktualnie rozwijanych kierunków badań na świecie. Mając na uwadze ten fakt wskazane jest prowadzenie dalszych badań mających na celu przedstawienie rozwiązania dedykowanego, jednak zgodnego z istniejącymi trendami poszukiwań.

BIBLIOGRAFIA

1. Arasu A., Babu S., Widom J.: The CQL Continious Query Language: Semantic Foundations and Query Execution. VLDB Journal, Vol. 15, No. 2, 2006, s. 121÷142.
2. Babcock B., Babu S., Datar M., Motwani R., Widom J.: Models and Issues in Data Stream Systems. PODS, 2002.

3. Abadi D., Carney D., Cetintemel U., Cherniack M., Convey C., Lee S., Stonebraker M., Tabul N., Zdonik S.: Aurora: A New Model and Architecture for Data Stream Management. VLDB Journal (12)2: 2003, s. 120÷139.
4. Jain N., Mishr S., Srinivasan A., Gherke J., Widom J., Balakrishan H., Cetintemel U., Chernick M., Tibbetts R., Zdonik S.: Towards a streaming SQL standard. VLDB Journal (1)2, 2008, s. 1379÷1390.
5. Botan I., Derakhshan R., Dindar N., Haas L., Miller R.J., Tatbul N.: SECRET: A Model for Analysis of the Execution Semantics of Stream Processing Systems. VLDB Journal, (3)1-2, 2010, s. 232÷243.
6. Complex Event Processing in Real World. Oracle White Paper, Sep 2007.
7. Arasu A., Cherniack M., Galvez E. F., Maier D., Maskey A., Ryvkina E., Stonebraker M., Tibbetts R.: Linear Road: A Stream Data Management Benchmark. VLDB Journal, 2004, s. 480÷491.
8. StreamSQL Guide. StreamBase White Paper, 2010.
9. Analyze and Act on Fast Moving Data: An Introduction to Complex Event Processing. Sybase White Paper, 2011.
10. Chandrasekaran S., Cooper O., Deshpande A., Franklin M. J., Hellerstein J. M., Hong W., Krishnamurthy S., Madden S. R., Raman V., Reiss F. M., Shah A.: TelegraphCQ: Continuous Dataflow Processing. Sigmod, 2003, s. 668÷668.
11. Continious Analytics Technical Whitepaper. Truviso White Paper, 2010.
12. Franklin M. J., Krishnamurthy S., Conway N., Li A., Russakovsky A., Thobre N.: Continuous Analytics: Rethinking Query Processing in a Network-Effect World. CIDR, 2009.
13. Dindar N., Guc B., Lau P., Ozal A., Soner M., Tatbul N.: DejaVu: Declarative Pattern Matching over Live and Archived Streams of Events. ACM SIGMOD'09, 2009.
14. Widera M., Wrobel J., Widera A., Matonia A.: A Method of the ensuring data integrity in the data stream management system. JMIT Vol. 8, 2004, s. 141÷148.
15. Widera M.: Deterministic method of data sequence processing. Annales UMCS Sectio AI Informatica, (4), 2006, s. 314÷331.
16. Tatbul N.: Streaming Data Integration: Challenges and Opportunities. ICDE 2010.
17. Pachev S.: MySQL Mechanizmy wewnętrzne bazy danych. Helion, O'Reilly 2009.

Recenzenci: Dr inż. Henryk Josiński
Dr hab. inż. Zygmunt Mazur, prof. Pol. Wrocławskiej

Wpłynęło do Redakcji 31 stycznia 2011 r.

Abstract

This paper presents the way as well as raises the issue of necessity for data integration in SPE-DBMS form. The paper provides an overview of the related work done in this area of research. There are many academic and commercial stream processing engines (SPEs) today, each of them, possessing its own execution semantics (Fig. 1). This may lead to differences in query results.

This paper presents an overview of some of the solutions to this problem. It also proposes the author's solution to this issue, as well as a certain data model along with the corresponding operators (Fig. 2). At the end of the paper, several technical aspects of SPE-DBMS problem solution (Fig. 3) are presented.

Streaming data integration issue is a broad research area, presenting researches with numerous challenges and opportunities, as the available on the issue papers have only dealt with the subject in a very limited way.

Adres

Michał WIDERA: Streams Lab, ul. Reymonta 56/1, 41-800 Zabrze, Polska,
michal.widera@streams.pl.