Christoph Bussler
Malu Castellanos
Umesh Dayal
Sham Navathe (Eds.)

Business Intelligence for the Real-Time Enterprises

LNCS 4365

First International Workshop, BIRTE 2006
Seoul, Korea, September 2006
Revised Selected Papers

Springer

Christoph Bussler Malu Castellanos
Umesh Dayal Sham Navathe (Eds.)

Business Intelligence for the Real-Time Enterprises

First International Workshop, BIRTE 2006
Seoul, Korea, September 11, 2006
Revised Selected Papers

Springer

Volume Editors

Christoph Bussler
Cisco Systems Inc.
San Jose, CA 95134, USA
E-mail: chbussler@aol.com

Malu Castellanos
Hewlett-Packard
CA 94304, USA
E-mail: malu.castellanos@hp.com

Umesh Dayal
Hewlett-Packard
CA, 94304, USA
E-mail: umeshwar.dayal@hp.com

Sham Navathe
Georgia Institute of Technology
Atlanta, Georgia 30332, USA
E-mail: navathe@yahoo.com

Library of Congress Control Number: 2007931598

CR Subject Classification (1998): H.3.5, H.4.1, H.2.7, H.5.3, K.4.3, K.4.4, K.6, J.1

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

ISSN 0302-9743
ISBN-10 3-540-73949-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-73949-4 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12100705 06/3180 5 4 3 2 1 0

Lecture Notes in Computer Science 4365

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

Lecture Notes in Computer Science

For information about Vols. 1–4534

please contact your bookseller or Springer

Vol. 4660: S. Džeroski, J. Todoroski (Eds.), Computational Discovery of Scientific Knowledge. X, 327 pages. 2007. (Sublibrary LNAI).

Vol. 4651: F. Azevedo, P. Barahona, F. Fages, F. Rossi (Eds.), Recent Advances in Constraints. VIII, 185 pages. 2007. (Sublibrary LNAI).

Vol. 4647: R. Martin, M. Sabin, J. Winkler (Eds.), Mathematics of Surfaces XII. IX, 509 pages. 2007.

Vol. 4632: R. Alhajj, H. Gao, X. Li, J. Li, O.R. Zaïane (Eds.), Advanced Data Mining and Applications. XV, 634 pages. 2007. (Sublibrary LNAI).

Vol. 4617: V. Torra, Y. Narukawa, Y. Yoshida (Eds.), Modeling Decisions for Artificial Intelligence. XII, 502 pages. 2007. (Sublibrary LNAI).

Vol. 4616: A. Dress, Y. Xu, B. Zhu (Eds.), Combinatorial Optimization and Application. XI, 390 pages. 2007.

Vol. 4613: F.P. Preparata, Q. Fang (Eds.), Frontiers in Algorithmics. XI, 348 pages. 2007.

Vol. 4612: I. Miguel, W. Ruml (Eds.), Abstraction, Reformulation, and Approximation. XI, 418 pages. 2007. (Sublibrary LNAI).

Vol. 4611: J. Indulska, J. Ma, L.T. Yang, T. Ungerer, J. Cao (Eds.), Ubiquitous Intelligence and Computing. XXIII, 1257 pages. 2007.

Vol. 4610: B. Xiao, L.T. Yang, J. Ma, C. Muller-Schloer, Y. Hua (Eds.), Autonomic and Trusted Computing. XVIII, 571 pages. 2007.

Vol. 4609: E. Ernst (Ed.), ECOOP 2007 — Object-Oriented Programming. XIII, 625 pages. 2007.

Vol. 4608: H.W. Schmidt, I. Crnkovic, G.T. Heineman, J.A. Stafford (Eds.), Component-Based Software Engineering. XII, 283 pages. 2007.

Vol. 4607: L. Baresi, P. Fraternali, G.-J. Houben (Eds.), Web Engineering. XVI, 576 pages. 2007.

Vol. 4606: A. Pras, M. van Sinderen (Eds.), Dependable and Adaptable Networks and Services. XIV, 149 pages. 2007.

Vol. 4605: D. Papadias, D. Zhang, G. Kollios (Eds.), Advances in Spatial and Temporal Databases. X, 479 pages. 2007.

Vol. 4604: U. Priss, S. Polovina, R. Hill (Eds.), Conceptual Structures: Knowledge Architectures for Smart Applications. XII, 514 pages. 2007. (Sublibrary LNAI).

Vol. 4603: F. Pfenning (Ed.), Automated Deduction – CADE-21. XII, 522 pages. 2007. (Sublibrary LNAI).

Vol. 4602: S. Barker, G.-J. Ahn (Eds.), Data and Applications Security XXI. X, 291 pages. 2007.

Vol. 4600: H. Comon-Lundh, C. Kirchner, H. Kirchner (Eds.), Rewriting, Computation and Proof. XVI, 273 pages. 2007.

Vol. 4599: S. Vassiliadis, M. Berekovic, T.D. Hämäläinen (Eds.), Embedded Computer Systems: Architectures, Modeling, and Simulation. XVIII, 466 pages. 2007.

Vol. 4598: G. Lin (Ed.), Computing and Combinatorics. XII, 570 pages. 2007.

Vol. 4597: P. Perner (Ed.), Advances in Data Mining. XI, 353 pages. 2007. (Sublibrary LNAI).

Vol. 4596: L. Arge, C. Cachin, T. Jurdziński, A. Tarlecki (Eds.), Automata, Languages and Programming. XVII, 953 pages. 2007.

Vol. 4595: D. Bošnački, S. Edelkamp (Eds.), Model Checking Software. X, 285 pages. 2007.

Vol. 4594: R. Bellazzi, A. Abu-Hanna, J. Hunter (Eds.), Artificial Intelligence in Medicine. XVI, 509 pages. 2007. (Sublibrary LNAI).

Vol. 4592: Z. Kedad, N. Lammari, E. Métais, F. Meziane, Y. Rezgui (Eds.), Natural Language Processing and Information Systems. XIV, 442 pages. 2007.

Vol. 4591: J. Davies, J. Gibbons (Eds.), Integrated Formal Methods. IX, 660 pages. 2007.

Vol. 4590: W. Damm, H. Hermanns (Eds.), Computer Aided Verification. XV, 562 pages. 2007.

Vol. 4589: J. Münch, P. Abrahamsson (Eds.), Product-Focused Software Process Improvement. XII, 414 pages. 2007.

Vol. 4588: T. Harju, J. Karhumäki, A. Lepistö (Eds.), Developments in Language Theory. XI, 423 pages. 2007.

Vol. 4587: R. Cooper, J. Kennedy (Eds.), Data Management. XIII, 259 pages. 2007.

Vol. 4586: J. Pieprzyk, H. Ghodosi, E. Dawson (Eds.), Information Security and Privacy. XIV, 476 pages. 2007.

Vol. 4585: M. Kryszkiewicz, J.F. Peters, H. Rybinski, A. Skowron (Eds.), Rough Sets and Intelligent Systems Paradigms. XIX, 836 pages. 2007. (Sublibrary LNAI).

Vol. 4584: N. Karssemeijer, B. Lelieveldt (Eds.), Information Processing in Medical Imaging. XX, 777 pages. 2007.

Vol. 4583: S.R. Della Rocca (Ed.), Typed Lambda Calculi and Applications. X, 397 pages. 2007.

Vol. 4582: J. Lopez, P. Samarati, J.L. Ferrer (Eds.), Public Key Infrastructure. XI, 375 pages. 2007.

Vol. 4581: A. Petrenko, M. Veanes, J. Tretmans, W. Grieskamp (Eds.), Testing of Software and Communicating Systems. XII, 379 pages. 2007.

Vol. 4580: B. Ma, K. Zhang (Eds.), Combinatorial Pattern Matching. XII, 366 pages. 2007.

Vol. 4579: B. M. Hämmerli, R. Sommer (Eds.), Detection of Intrusions and Malware, and Vulnerability Assessment. X, 251 pages. 2007.

Vol. 4578: F. Masulli, S. Mitra, G. Pasi (Eds.), Applications of Fuzzy Sets Theory. XVIII, 693 pages. 2007. (Sublibrary LNAI).

Vol. 4577: N. Sebe, Y. Liu, Y.-t. Zhuang (Eds.), Multimedia Content Analysis and Mining. XIII, 513 pages. 2007.

Vol. 4576: D. Leivant, R. de Queiroz (Eds.), Logic, Language, Information and Computation. X, 363 pages. 2007.

Vol. 4575: T. Takagi, T. Okamoto, E. Okamoto, T. Okamoto (Eds.), Pairing-Based Cryptography – Pairing 2007. XI, 408 pages. 2007.

Vol. 4574: J. Derrick, J. Vain (Eds.), Formal Techniques for Networked and Distributed Systems – FORTE 2007. XI, 375 pages. 2007.

Vol. 4573: M. Kauers, M. Kerber, R. Miner, W. Windsteiger (Eds.), Towards Mechanized Mathematical Assistants. XIII, 407 pages. 2007. (Sublibrary LNAI).

Vol. 4572: F. Stajano, C. Meadows, S. Capkun, T. Moore (Eds.), Security and Privacy in Ad-hoc and Sensor Networks. X, 247 pages. 2007.

Vol. 4571: P. Perner (Ed.), Machine Learning and Data Mining in Pattern Recognition. XIV, 913 pages. 2007. (Sublibrary LNAI).

Vol. 4570: H.G. Okuno, M. Ali (Eds.), New Trends in Applied Artificial Intelligence. XXI, 1194 pages. 2007. (Sublibrary LNAI).

Vol. 4569: A. Butz, B. Fisher, A. Krüger, P. Olivier, S. Owada (Eds.), Smart Graphics. IX, 237 pages. 2007.

Vol. 4566: M.J. Dainoff (Ed.), Ergonomics and Health Aspects of Work with Computers. XVIII, 390 pages. 2007.

Vol. 4565: D.D. Schmorrow, L.M. Reeves (Eds.), Foundations of Augmented Cognition. XIX, 450 pages. 2007. (Sublibrary LNAI).

Vol. 4564: D. Schuler (Ed.), Online Communities and Social Computing. XVII, 520 pages. 2007.

Vol. 4563: R. Shumaker (Ed.), Virtual Reality. XXII, 762 pages. 2007.

Vol. 4562: D. Harris (Ed.), Engineering Psychology and Cognitive Ergonomics. XXIII, 879 pages. 2007. (Sublibrary LNAI).

Vol. 4561: V.G. Duffy (Ed.), Digital Human Modeling. XXIII, 1068 pages. 2007.

Vol. 4560: N. Aykin (Ed.), Usability and Internationalization, Part II. XVIII, 576 pages. 2007.

Vol. 4559: N. Aykin (Ed.), Usability and Internationalization, Part I. XVIII, 661 pages. 2007.

Vol. 4558: M.J. Smith, G. Salvendy (Eds.), Human Interface and the Management of Information, Part II. XXIII, 1162 pages. 2007.

Vol. 4557: M.J. Smith, G. Salvendy (Eds.), Human Interface and the Management of Information, Part I. XXII, 1030 pages. 2007.

Vol. 4556: C. Stephanidis (Ed.), Universal Access in Human-Computer Interaction, Part III. XXII, 1020 pages. 2007.

Vol. 4555: C. Stephanidis (Ed.), Universal Access in Human-Computer Interaction, Part II. XXII, 1066 pages. 2007.

Vol. 4554: C. Stephanidis (Ed.), Universal Acess in Human Computer Interaction, Part I. XXII, 1054 pages. 2007.

Vol. 4553: J.A. Jacko (Ed.), Human-Computer Interaction, Part IV. XXIV, 1225 pages. 2007.

Vol. 4552: J.A. Jacko (Ed.), Human-Computer Interaction, Part III. XXI, 1038 pages. 2007.

Vol. 4551: J.A. Jacko (Ed.), Human-Computer Interaction, Part II. XXIII, 1253 pages. 2007.

Vol. 4550: J.A. Jacko (Ed.), Human-Computer Interaction, Part I. XXIII, 1240 pages. 2007.

Vol. 4549: J. Aspnes, C. Scheideler, A. Arora, S. Madden (Eds.), Distributed Computing in Sensor Systems. XIII, 417 pages. 2007.

Vol. 4548: N. Olivetti (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. X, 245 pages. 2007. (Sublibrary LNAI).

Vol. 4547: C. Carlet, B. Sunar (Eds.), Arithmetic of Finite Fields. XI, 355 pages. 2007.

Vol. 4546: J. Kleijn, A. Yakovlev (Eds.), Petri Nets and Other Models of Concurrency – ICATPN 2007. XI, 515 pages. 2007.

Vol. 4545: H. Anai, K. Horimoto, T. Kutsia (Eds.), Algebraic Biology. XIII, 379 pages. 2007.

Vol. 4544: S. Cohen-Boulakia, V. Tannen (Eds.), Data Integration in the Life Sciences. XI, 282 pages. 2007. (Sublibrary LNBI).

Vol. 4543: A.K. Bandara, M. Burgess (Eds.), Inter-Domain Management. XII, 237 pages. 2007.

Vol. 4542: P. Sawyer, B. Paech, P. Heymans (Eds.), Requirements Engineering: Foundation for Software Quality. IX, 384 pages. 2007.

Vol. 4541: T. Okadome, T. Yamazaki, M. Makhtari (Eds.), Pervasive Computing for Quality of Life Enhancement. IX, 248 pages. 2007.

Vol. 4539: N.H. Bshouty, C. Gentile (Eds.), Learning Theory. XII, 634 pages. 2007. (Sublibrary LNAI).

Vol. 4538: F. Escolano, M. Vento (Eds.), Graph-Based Representations in Pattern Recognition. XII, 416 pages. 2007.

Vol. 4537: K.C.-C. Chang, W. Wang, L. Chen, C.A. Ellis, C.-H. Hsu, A.C. Tsoi, H. Wang (Eds.), Advances in Web and Network Technologies, and Information Management. XXIII, 707 pages. 2007.

Vol. 4536: G. Concas, E. Damiani, M. Scotto, G. Succi (Eds.), Agile Processes in Software Engineering and Extreme Programming. XV, 276 pages. 2007.

Preface

The 1st Workshop on Business Intelligence for the Real-Time Enterprise (BIRTE 2006) was held on September 11, 2006 in conjunction with the 32nd International Conference on Very Large Data Bases (VLDB 2006) in Seoul, Korea. The co-location with VLDB is very important as the topic of the workshop was centered on different aspects in the lifecycle of business intelligence on very large enterprise-wide operational real-time data sets.

In today's competitive and highly dynamic environment, analyzing data to understand how the business is performing, to predict outcomes and trends, and to improve the effectiveness of business processes underlying business operations has become critical. The traditional approach to reporting is not longer adequate; users now demand easy-to-use intelligent platforms and applications capable of analyzing real-time business data to provide insight and actionable information at the right time. The end goal is to improve the enterprise performance by better and timelier decision making, enabled by the availability of up-to-date, high-quality information.

As a response, the notion of "real-time enterprise" has emerged and is beginning to be recognized in the industry. Gartner defines it as "using up-to-date information, getting rid of delays, and using speed for competitive advantage is what the real-time enterprise is all about... Indeed, the goal of the real-time enterprise is to act on events as they happen. "

Although there has been progress in this direction and many companies are introducing products towards making this vision reality, there is still a long way to go. In particular, the whole lifecycle of business intelligence requires new techniques and methodologies capable of dealing with the new requirements imposed by the real-time enterprise. From the capturing of real-time business performance data to the injection of actionable information back into business processes, all the stages of the business intelligence (BI) cycle call for new algorithms and paradigms as the basis of new functionalities including dynamic integration of real-time data feeds from operational sources, evolution of ETL transformations and analytical models, and dynamic generation of adaptive real-time dashboards, just to name a few.

The goal of the BIRTE 2006 workshop was to provide a forum for the discussion of five major aspects of business intelligence for the real-time enterprise: Models and Concepts for Real-Time Enterprise Business Intelligence, Architectures for Real-Time Enterprise Business Intelligence, Uses Cases of Real-Time Enterprise Business Intelligence, Applications of Real-Time Enterprise Business Intelligence and Technologies for the Real-Time Enterprise Business Intelligence.

The workshop started with the keynote "Practical Considerations for Real-Time Business Intelligence" by Donovan Schneider. It continued with several sessions addressing various aspects of real-time data analysis. The first session "Streaming Data" concentrated on data streams as one mechanism for obtaining real-time enterprise data. The second session "Data Loading and Data Warehouse Architectures" addressed data loading and data warehouse architectures that both are a basis for the actual analysis task. The third session "Integration and Data Acquisition" focused on

heterogeneous data sources as well as mechanisms for obtaining real-time data. The final session "Business Processes and Contracts" extended the analysis aspect from data to processes. The workshop closed with the interesting panel ""How Real Can Real-Time Business Intelligence Be?" moderated by Malu Castellanos, and Chi-Ming Chen, Mike Franklin, Minos Garofalakis, Wolfgang Lehner, Stuart Madnick and Krithi Ramamrithan as speakers.

The field of business intelligence for the real-time enterprise is fairly new, albeit increasingly important. This first workshop on the topic was meant to be a starting point of a series of several workshops covering various aspects in more detail over time. As academic research and industrial application experience more in-depth insights and use of this technology, an interesting research field opens up as well as an exciting area for practitioners. We encourage researchers and those in industry to continue their exciting work, and we encourage newcomers to enter this challenging and increasingly important field as there is still a lot of exciting work to be done.

We wish to express special thanks to the Program Committee members for providing their technical expertise in reviewing the submitted papers and preparing an interesting program. We are particularly grateful to the keynote speaker, Donovan Schneider, for delighting us with his very interesting keynote. Special recognition goes to the panelists for their enthusiastic participation in presenting their perspectives. To the authors of the accepted papers we express our appreciation for sharing their work and experiences in this workshop. Finally, we would like to extend many thanks to the VLDB 2006 Workshop Co-Chairs, Sang-goo Lee and Ming-Chien Shan, for their support in making this workshop possible.

September 2006

Christoph Bussler
Malu Castellanos
Umesh Dayal
Sham Navathe

Organization

Organizing Committee

General Chair

Umeshwar Dayal, Hewlett-Packard, USA

Program Committee Chairs

Christoph Bussler, Cisco Systems, Inc., USA
Malu Castellanos, Hewlett-Packard, USA
Sham Navathe, Georgia Institute of Technology, USA

Program Committee

Christof Bornhoevd, SAP Labs, USA
Mike Franklin, UC Berkeley, USA
Venkatesh Ganti, Microsoft, USA
Dimitrios Georgakopoulos, Telcordia Technologies, USA
Ramesh Jain, UC Irvine, USA
Meichun Hsu, HP Labs, China
Kamal Karlapalem, IIIT Hyderabad, India
Rajesh Parekh, Yahoo, USA
Torben B. Pedersen, Aalborg University, Denmark
Ee Peng, Nanyang Technological University, Singapore
Krithi Ramamritham, IIT Bombay, India
W.M.P. Van der Alst, Eindhoven University of Technology, The Netherlands
Panos Vassiliadis, University of Ioannina, Greece
Kazi Zaman, Siebel Systems Inc., USA

Publication Chair

Kamalakar Karlapalem, IIIT Hyderabad, India

Reviewers

Bin Zhang

Table of Contents

Practical Considerations for Real-Time Business Intelligence

Donovan A. Schneider

Yahoo! Inc.
701 First Avenue, Sunnyvale, CA 94089
dschneider@yahoo-inc.com

Abstract. The area of real-time business intelligence is ill defined in industry. In this extended abstract we highlight the practical requirements through the use of examples across several domains.

1 Introduction

Real-time Business Intelligence (BI) is an ambiguous area. To be practical in industry, real-time BI must satisfy two requirements:

1. Time is money. It costs money to reduce latency. The decisions to be made on the reduced latency data must justify the investment.
2. Actionable Data. Effective decision making requires rich contextual data.

1.1 Examples

We discuss several examples across different domains to highlight the practical real-time business intelligence requirements. Fraud detection is the canonical example of real-time BI. It involves detecting anomalies, for example, in credit card usage. Detection must be done quickly in order to prevent further fraudulent use. Fraud detection can be thought of as a form of alerting. The time in which to make a decision may be seconds or minutes. However, a surprising amount of context is required to prevent excessive numbers of false positives or false negatives. For example, different alerting applications may require knowledge of days of weeks (e.g., weekday vs. weekend), holidays, geographical location, past behavior and trends, to name a few. Without sufficiently rich context the decision making will be of limited usefulness.

Another illustrative example is real-time marketing. When a customer calls into a call center a decision may be made to pitch a premium service (up-sell) or a related product (cross-sell). The decision of which marketing message to present must be made in seconds (or sub-seconds). Again, in order for the marketing to be effective ample information must be available, including the reason for the current call, previous offers and their acceptance, behavior of similar users, history of the customer, etc. Often this rich contextual data is built offline in the form of models in order to meet the ultra low latency requirements.

C. Bussler et al. (Eds.): BIRTE 2006, LNCS 4365, pp. 1 – 3, 2007.

Many other forms of traditional business intelligence exist. These require very rich contextual data including role-specific views (e.g., an executive, district manager and a sales representative see different views of the same data) and task specific views (sometimes referred to as guided navigation). Most of these applications do not require decisions to be made in seconds, but rather in minutes. Thus, these types of applications are classified as near real-time, not real-time. When humans are making the decisions a person needs time to analyze the information and make a decision. Even when programs/systems are making the decisions enough data must be available for a useful decision to be made. Examples include incident tracking, inventory management, and sales analytics.

Several interesting examples of BI exist in the domain of web analytics. A common example is a recommendation service where other products (movies, books, etc.) are recommended based on what similar people liked or bought. Behavioral targeting is somewhat similar to this; it involves showing specific advertisements or personalized content to a user based on sophisticated models that may include demographic data (age, gender, income level, etc.), geographic data, and past and present user behavior. The interesting aspect of these examples is that although the decision of the recommendation or advertisement to present must be made in real-time (seconds), the context is often built off-line as part of a sophisticated modeling process. Refinements to the model may be made in real-time.

An interesting marketing area for the web is search engine marketing (SEM). SEM involves bidding on search terms from the search engine vendors (e.g., Yahoo!, Google, and MSN) to lead users to a particular web page, and then analyzing the click-thru and conversion rates of the users. Decisions to buy more or less of a search term must be made quickly. However, latencies of many minutes are common because enough user traffic must be analyzed before an effective decision can be made.

Another category of web analytics is popularity. Although it is possible to update the list of most popular search terms or downloads in real-time, this is often not done because the decisions to be made do not justify the investment and/or because of concerns of abuse (spam, pornography, abuse, etc.).

Experimentation is another common web application. An experiment may involve an A/B test to evaluate whether a new page design is superior. The metrics may involve measuring an increase of time spent on the page, click-thru rate on advertisements, or moving a user to a desired end state such as upgrading to a premium service. In order for an accurate decision to be made, though, enough users must see both versions. Thus, decisions cannot typically be made for at least 15 minutes.

Some common themes can be seen from these examples. First, rich contextual data is needed to make effective decisions. Second, relatively few applications justify true real-time decision making. This is because the cost of providing the data in context does not justify the decision making, or simply because more time is needed to acquire the context to make an informed decision. In many cases, the requirement is for near real-time business intelligence which is measured in minutes, not seconds.

2 Challenges

There are many technical challenges in providing practical real-time or near real-time business intelligence. Because the decisions to be made have been determined to be

valuable and must be made with low latency, the underlying systems must be highly available. The rich contextual data often implies a high degree of data integration, access to detailed data, and access to aggregated/trending data. Sophisticated modeling may often be employed in order to classify behavior into similar segments. As the requirements get closer to real-time (seconds), the applications must tolerate some amount of data incompleteness or inaccuracy, as it is often not feasible (financially or technically) to provide 100% of the data within such strict time requirements.

3 Architectures

Several architectures exist for providing business intelligence. The most common solution for real-time business intelligence is to build a custom system. Commercial off the shelf platforms are not typically suited to the ultra low latency access, high availability and integration with detailed and aggregated data.

Commodity solutions for near real-time business intelligence typically involve enterprise data warehouses; these can be virtual or physical. The warehouse environment provides the detailed and aggregated data as well as high availability. The primary challenges are to load the data into the warehouse with low latency and to query it with low latency.

Several startups have emerged to build platforms to support low latency decision making on high volumes of streaming data. The primary challenges are to build a platform that is cost effective for applications that do not require ultra low-latency and to integrate with alternative data sources to provide the rich context necessary for decision making.

4 Summary

Most applications do not require real-time business intelligence which we define as making decisions in seconds or sub-seconds. Rather, given the difficulty and high cost of providing real-time BI, many of these applications can tolerate, or even require, less strict latency requirements.

What Can Hierarchies Do for Data Streams?

Xuepeng Yin and Torben Bach Pedersen

Aalborg University

Abstract. Much effort has been put into building data streams management systems for querying data streams. However, the query languages have mostly been SQL-based and aimed for low-level analysis of base data; therefore, there has been little work on supporting OLAP-like queries that provide real-time multi-dimensional and summarized views of stream data. In this paper, we introduce a multi-dimensional stream query language and its formal semantics. Our approach turns low-level data streams into informative high-level aggregates and enables multi-dimensional and granular OLAP queries against data streams, which supports the requirements of today's real time enterprises much better. A comparison with the STREAM CQL language shows that our approach is more flexible and powerful for high-level OLAP queries, as well as far more compact and concise.

Classification: Real-time OLAP, Streaming data, Real-time decision support.
Submission Category: Regular paper.

1 Introduction

Pervasive Computing is the newest wave within the IT world. Examples are temperature and noise sensors that can measure whether the environment behave as expected, and report irregularities. The data produced by these devices are termed *data streams*. Due to the different characteristics of data streams (e.g., continuous, unbounded, fast, etc.) from those of traditional, static data, it will most often be infeasible to handle the total data stream from a large number of devices using traditional data management technologies, and new techniques must therefore be introduced.

Recent studies have been focusing on building Data Stream Management Systems (DSMS) similar to the traditional DBMS's. However, queries in these systems have to a large extent been based on SQL and targeted for low-level data, and therefore are not suitable in performing OLAP-like operations to provide multi-dimensional and multi-granular summaries of data streams. As the notion of real-time enterprise is more and more recognized in the industry, analyzing data in a timely fashion for effective decision making in today's competitive and highly dynamic environment has become critical. Examples of such technologies are real-time OLAP, real-time Business Activity Monitoring (BAM), and streaming data. The solution presented in this paper is to build a multi-dimensional stream query language with built-in support for hierarchies, enabling the OLAP functionalities such as slice, roll-up and drill-down queries for powerful and timely analysis on data streams, which supports the requirements of today's real time enterprises much better.

Specifically, we present the following novel issues: 1) a new cube algebra that enables multi-dimensional and multi-granular queries against static OLAP cubes. That is,

C. Bussler et al. (Eds.): BIRTE 2006, LNCS 4365, pp. 4–19, 2007.

high-level and low-level facts representing summaries and details can be presented together in a query result and different levels of selection criteria can also be applied. 2) conversion operators that transfer a continuous data stream into conventional cubes and also the other way around. 3) stream operators that perform OLAP operations on data streams, e.g., aggregates, roll-ups and drill-downs, with all the powers of the cube operators on static data. 4) comparisons with the Stanford STREAM language for roll-up and drill-down queries. We believe we are the first to present a multi-dimensional stream query language capable of performing typical OLAP operations against data streams, and the concrete query semantics for the above operators. The comparisons with the STREAM CQL query language suggest that our approach is much more compact and concise, and more effective in multi-dimensional and multi-granular analysis.

There has been a substantial amount of work on the general topic of OLAP [1]. Relevant work includes OLAP data modeling and querying [2,3,4,5]. However, all this work builds their solutions for static data, e.g., stored relational data. A more related topic is data integration of OLAP databases with dynamic XML data [6]. However, the system proposed is targeted for B2B business data on the web, which has far smaller data volumes and update frequencies in comparison with data streams. Recent interests in building data stream management system has generated a number of projects, including Aurora [7], Gigascope [8], NiagaraCQ [9], STREAM [10], and TelegraphCQ [11]. The query languages used by these systems generally have SQL-like syntax and the operators are analogous to operators in the relational algebra. Gigascope [12] supports shared fine-granularity aggregation queries to compute multiple coarser aggregation queries with different grouping attributes, which is a maintenance optimization rather than an OLAP extension. Therefore, OLAP-like queries involving hierarchical structures upon the basic stream schema have not yet been supported by current DSMS's.

The following descriptions of our approach is based on a sensor network, where sensor motes are deployed in a building to measure temperature every thirty seconds, producing a data stream with the schema SensorStream(Temperature, Id, Timestamp), capturing the current temperature reading, a unique identifier of the sensor, and the time of measurement. Also, we define the measure Temperature which is characterized by the dimensions Location(All-Floor-Room-Id), and Time(All-Day-Hour-Minute-Second), where the bottom levels are the attributes from the stream schema. A regular OLAP database, SensorCube, contains all the stream data produced on June 15, 2005.

The rest of the paper is organized as follows. Section 2 describes a query algebra and a multi-dimensional query language over a static cube model. Then, Section 3 introduces the stream model and the stream query language. Section 4 compares our language with the STREAM CQL with respect to OLAP-like analysis. Section 5 describes the current implementation. Finally Section 6 concludes the paper.

2 Querying Cubes

This section introduces the terms used in the following descriptions of cube operations. More formal definitions about the data model and the operators can be found in [13].

The Cube Model. A *dimension* D_i has a hierarchy of *levels* $L_{i1}, \ldots, L_{ik_i}$. A level is a set of *dimension values*. There exists a partial order, denoted $\sqsubset_i$ such that for two levels

in a dimension, L_{il} and L_{ik}, we say $L_{il} \sqsubset_i L_{ik}$ holds if and only if the values of the higher level L_{ik} contain the values of the lower level L_{il}. For example, let D_i be a time dimension, $Day \sqsubset_i Year$ because years contain days. Similarly, a partial order also exists between dimension values. We say that $e_1 \sqsubset_{D_i} e_2$ if e_2 can be said to contain e_1. For example, the year 2004 has the date, Feb. 29th 2004, is denoted 2004-02-29 $\sqsubset_{D_i}$2004. We also define $\sqsubseteq$ to denote a dimension value contained or equal to another. We use $e_i \in D_i$ to represent an arbitrary value e_i in dimension D_i.

A *measure* M_j is a set of numeric values that are being analyzed, e.g., sales, quantity, etc. A fact contains measure and dimension values, i.e., a tuple with the schema $(M_1, \ldots, M_m, D_1, \ldots, D_n)$ where M_j is a measure and D_i is a dimension. A fact is $r = (v_1, \ldots, v_m, e_1, \ldots, e_n)$, where v_i is a measure value characterized by dimension values $e_1, \ldots, e_n$. Also, a fact can have *any granularity in any dimension*, i.e., $(e_1, \ldots, e_n) \in D_1 \times \ldots \times D_n$. A *fact table* R is a set of facts with the schema $(M_1, \ldots, M_m, D_1, \ldots, D_n)$, such that in each fact, the measure values $v_1, \ldots, v_m$ are characterized by the values from the same set of dimensions $D_1, \ldots, D_n$ at any granular. For example, a fact table could be { (28.0, floor#1, 2005-06-15 08), (29.0, room#11, 2005-06-15 08), (27.0, room#12, 2005-06-15 08)}, where, there exist facts for the hourly temperatures of floors as well as rooms. A *cube* is a three tuple $C = (N, D, R)$ consisting of the name of the cube N, a non-empty set of dimensions $D = \{D_1, \ldots, D_n\}$ and a fact table R.

Querying Cubes. The cube generalized projection operator (Π_{cube}) turns the facts in a cube into higher level facts and aggregates the measures correspondingly. We also allow the result facts to have any granularity in any dimension to enable the roll-up and drill-down effects on certain dimensions in the query results, meaning that there might be multiple combinations of grouping values where the values from the same dimension in different combinations may be from different levels. When compared with the CUBE and ROLLUP operators [3], the cube generalized projection operator is more flexible and powerful in terms of OLAP-like queries. Specifically, the Π_{cube} operator can roll-up the input cube to any combination of levels without having to enumerate a full set of *super-aggregates* (as the CUBE and ROLLUP operators always do) and to specify the subset using conditions on the GROUPING() functions [3]. Moreover, the operator allows roll-up to or drill-down on a specific dimension value to, e.g., monitor anomalies on certain locations, which is not possible for the CUBE and ROLLUP operators. Thus, the cube generalized projection operator serves better for the purpose of our approach.

To ensure correct aggregation and also to be deterministic, we always use the *lowest-level* facts in each group, where in each tuple, every dimension value is from the bottom level. For example, to compute the hourly average temperature of a floor, we use the tuples directly from the sensors with the timestamps at the second level. We say such tuples have the lowest *level-combination* which is (Id, Second). However, sometimes, the tuples with such a level-combination may not be available, e.g., the base tuples are rolled up to higher levels, then the lowest level-combination now contains the lowest available levels in the dimensions of the current tuples. Currently, we assume that there always exists a lowest level-combination (in either sense above) in a group.

Definition 1 (Cube Generalized Projection). *Suppose that $C = (N, D, R)$ is the input cube, the generalized projection operator is defined as:* $\Pi_{cube[\{e_{i_1 1}, \ldots, e_{i_1 n_1}\}, \ldots,}$

$_{\{e_{i_k1},\ldots,e_{i_kn_k}\}]<f_{j_1}(M_{j_1}),\ldots,f_{j_l}(M_{j_l})>}(C) = (N, D, R')$, *where* N *and* D *are the same as in* C, R' *is the new fact table,* $\{e_{i_h1}, \ldots, e_{i_hn_h}\}$ *is a set of dimension values from dimension* D_{i_h} *and* $f_{j_1}, \ldots, f_{j_l}$ *are the given aggregate functions for the specified measures* $\{M_{j_1}, \ldots, M_{j_l}\}$. *Similar to the relational aggregate operator, a combination of the dimension values (i.e. grouping values) from each of the given sets constitutes a group of fact tuples over which the measures are aggregated. A group is denoted as* $g_{(e_{i_1j_1},\ldots,e_{i_kj_k})}$ *where* $(e_{i_1j_1}, \ldots, e_{i_kj_k}) \in \{e_{i_11}, \ldots, e_{i_1n_1}\} \times \ldots \times \{e_{i_k1}, \ldots, e_{i_kn_k}\}$. *A group* $g_{(e_{i_1j_1},\ldots,e_{i_kj_k})}$ *is the set of tuples such that the values from the dimensions* $D_{i_1}, \ldots, D_{i_k}$ *in the tuple are contained in the values* $e_{i_1j_1}, \ldots, e_{i_kj_k}$ *from the same dimensions, i.e.,* $g_{(e_{i_1j_1},\ldots,e_{i_kj_k})} = \{(v_1, \ldots, v_m, e_1, \ldots, e_n) \in F | \exists e_{i_1}, \ldots, e_{i_k} \in \{e_1, \ldots, e_n\}(e_{i_1} \sqsubseteq_{D_{i_1}} e_{i_1j_1} \wedge \ldots \wedge e_{i_k} \sqsubseteq_{D_{i_k}} e_{i_kj_k})\}$.

Each group produces one fact tuple consisting of the measures calculated over the tuples in the group. A fact is a lowest-level fact, if for any dimension value e_{i_h} *in such a tuple, no descendants of* e_{i_h} *exists in any other fact of the group, and the group of such tuples is* g_{lowest}, *i.e., for a group* $g_{(e_{i_1j_1},\ldots,e_{i_kj_k})}$, $g_{lowest} = \{(v_1, \ldots, v_m, e_1, \ldots, e_n) \in g_{(e_{i_1j_1},\ldots,e_{i_kj_k})} | \nexists (v'_1, \ldots, v'_m, e'_1, \ldots, e'_n) \in g_{(e_{i_1j_1},\ldots,e_{i_kj_k})}, e'_{i_h} \in \{e'_1, \ldots, e'_n\}, e_{i_h} \in \{e_1, \ldots, e_n\}(e'_{i_h} \sqsubset e_{i_h})\}$. *The fact tuple produced over the group is* $r = (v'_{j_1}, \ldots, v'_{j_l}, e_{i_1j_1}, \ldots, e_{i_kj_k})$, *where* $v'_{j_q} = f_{M_{j_q}}(\{v_{j_q} | (v_1, \ldots, v_{j_q}, \ldots, v_m, e_1, \ldots, e_n) \in g_{lowest}\})$ *and the input to the aggregate function is a multiset. We use* $g_{(e_{i_1j_1},\ldots,e_{i_kj_k})} \mapsto r$ *to denote the relation between the group and the result tuple. The result fact table is* $R' = \{r | g \in G \wedge g \mapsto r\}$, *where* G *is the set of all the non-empty groups, i.e.,* $G = \{g_{(e_{i_1j_1},\ldots,e_{i_kj_k})} | (e_{i_1j_1}, \ldots, e_{i_kj_k}) \in \{e_{i_11}, \ldots, e_{i_in_1}\} \times \ldots \times \{e_{i_k1}, \ldots, e_{i_kn_k}\} \wedge g_{(e_{i_1j_1},\ldots,e_{i_kj_k})} \neq \emptyset\}$.

Temperature	Location	Time
28.0	s#1	2005-06-15 08:00:00
28.0	s#2	2005-06-15 08:00:00
27.0	s#3	2005-06-15 08:00:00
27.0	s#4	2005-06-15 08:00:00
28.2	s#1	2005-06-15 08:00:30
28.2	s#2	2005-06-15 08:00:30
27.2	s#3	2005-06-15 08:00:30
27.2	s#4	2005-06-15 08:00:30

(a) The fact table before selection

Temperature	Location	Time
27.6	floor#1	2005-06-15 08:00
28.1	room#11	2005-06-15 08:00
27.1	room#12	2005-06-15 08:00

(b) The fact table after the operation

Fig. 1. The fact tables before and after the cube generalized projection

Example 1. Let the table in Figure 1(a) be the current fact table of SensorCube. The cube generalized projection $\Pi_{cube[\{\text{floor\#1, room\#11, room\#12}\}, \{\text{2005-06-15 08:00}\}]}(\text{SensorCube})$ computes the average temperature per minute for floor#1, room#11, and room#12. The fact table after the operation is shown in Figure 1(b). The average floor temperature is computed over all the tuples in Figure 1(a), which are directly from the sensors and all at the lowest-level. Similarly, the average temperature of room#11 is computed over all the