Mobile E-Services and Their Challenges to Data Warehousing

Christian S. Jensen[1] and Torben Bach Pedersen[2]

[1] Department of Computer Science, Aalborg University, csj@cs.auc.dk, www.cs.auc.dk/~csj
[2] Department of Computer Science, Aalborg University, tbp@cs.auc.dk, www.cs.auc.dk/~tbp

Abstract. Continued advances in hardware technologies combine to create a new class of information services, termed mobile e-services, or simply m-services, which exploits the advances in, among others, wireless communications, positioning, and miniaturization. Because the users do not merely interact with the services from behind stationary desktop computers, but from a variety of increasingly unobtrusive information appliances while on the move, location information plays a fundamental role, and new types of services become of interest. Such services include tracking, way-finding, traffic management, safety-related services, and mixed-reality games, to name but a few.

Data warehousing has the potential for playing an essential part in m-services. However, for data warehousing to be successful in an m-service scenario, new challenges must be met by data warehousing technologies. Such challenges include support for non-standard dimension hierarchies and imprecision and varying precision in the data; transportation networks; continuous change; closed-loop usage; and dynamic services. This paper outlines a general m-services scenario and describes central challenges to be met by data warehousing in order for it to reach its full potential for usage in m-services.

1 Introduction

Continued advances in hardware technologies combine to create a new research area, termed mobile e-services (hereinafter "m-services"). In particular, the following seven trends are particularly important in enabling the area of m-services.

1. Continued miniaturization of electronics technologies. This applies across a wide range of technologies, including, e.g., processors and a variety of data storage technologies.
2. Continued advances in display devices.
3. Continued advances in wireless communications. Perhaps most importantly, the bandwidth of wireless communication continues to increase.
4. Continued advances in positioning technologies. Perhaps most prominently, GPS (global positioning system) is becoming increasingly accurate. In a few years, the accuracy is expect to reach 5 meters. GPS is free and globally accessible. Unlike radar, GPS in not a surveillance system because an object being positioned is responsible for, and in control of, measuring its own position, which enables the object itself to decide whether or not to reveal its position.
5. Improved power management of wireless computing devices.
6. Improved performance of general computing technologies. This includes faster processors and main memories and disks with larger capacities.
7. General improvement in the performance/price ratio of electronics hardware. For example, over the 18-month period from mid-1999 to the end of 2000, it is expected that there will be sold as much disk storage as has been sold so far in history until the start of this period.

The challenges faced in the area of m-services call for expertise that is currently accumulated in diverse areas of computer science and related areas, which have hitherto evolved quite separately, such as spatial, temporal, and spatio-temporal databases (including GIS); Internet technology; data warehousing; data mining; and mobile computing.

This paper puts focus on the relation between m-services and data warehousing. Data warehousing technology may play a prominent part in e-services. When moving objects use m-services, they generate so-called click-streams, which are accumulated in logs. Information gleamed from these streams is integrated with other user and service information in a data warehouse. This warehouse is then used for on-the-fly mass-customization of the services: the clicks generated from the use of a service are matched up against the warehouse.

The paper more specifically attempts to identify important, unmet challenges that data warehousing technology must meet in order to reach its full potential when deployed in an m-service context. It is hoped that more concrete incarnations of these challenges will spur new, productive research in an emerging, exciting, and increasingly important area.

The remainder of this paper first introduces m-services in more detail. In Section 3, a brief introduction to data warehousing is given, and a case study is introduced that is used for exemplification in the remainder of the paper. Then follows in Section 4 descriptions of central m-service challenges to data warehousing. These are organized into six subsections. Finally, Section 5 summarizes the paper.

2 M-Services

This section describes first the kinds of mobile objects that m-services concerns. This is followed by a description of the usage scenario for data warehousing technology in m-services and by examples of the novel services that may be offered.

2.1 Online, Position-Aware, Wireless Mobile Objects

The coming years will witness very large quantities of on-line (i.e., Internet-worked), position-aware, wireless objects capable of movement. Examples of such objects include the following.

– Consumers using Internet-enabled mobile-phone terminals (e.g., WAP or I-mode), possibly with enhanced displays, as well as diverse types of personal digital assistants (PDAs, e.g., PalmPilots).
  As examples, tourists may carry on-line and position-aware "cameras," "wrist watches" will evolve to become on-line and position-aware, as will "clothing" and luggage.
– Vehicles with computing equipment, including automobiles, public transportation vehicles, recreational vehicles, sea vessels, etc.
  Luxury cars already carry navigation equipment, and a wide range of such equipment is available to consumers for integration into older vehicles as well as new economy vehicles. The major car manufacturers are working intensely with, e.g., the major mobile phone providers to integrate Internet access into their future automobiles.
– Home appliances.
  Using local-area wireless communications technologies (e.g., Bluetooth) within homes and wireline technologies beyond the homes, home appliances will increasingly be on-line, at lower and lower cost.

Some technology observers predict that the Internet of the future will extend to billions of wireless Gizmos.

2.2 General M-Service Scenario

The following general scenario is considered.

Moving objects use e-services that involve location information. The objects disclose their positional information (position, speed, velocity, etc.) to the services, which in turn use this and other information to provide specific functionality.

The services maintain a log of the requests made to them, and use this for analyzing user interaction with the service. Specifically, in Internet contexts a sequence of requests is called a click-stream. The services accumulate data derived from the click-streams and integrate this with other customer data in data warehouses, which are very large repositories of integrated information used for data analysis (described further in the next section).

The data in the data warehouse is used for mass-customization of the services, so that each user receives a service customized to the user's specific situation, preferences, and needs. This involves the immediate generation of dynamic web-page content based on the warehouse and the user's current interaction with the service, e.g., the display of advertisements that are relevant to this particular user at the particular place and time.

In addition, the warehouse is used for delayed modification of the services provided, and for longer-term strategic decision making. Business intelligence techniques such as on-line analytical processing and data mining are used for these purposes.

The integration of location information into this scenario has received very little attention and offers a number of fundamental challenges. Common to these challenges is the task of extending techniques that work well for static data to support the kinds of dynamic, continuously evolving data that is found in m-services.

2.3 Example M-Services

The five examples of m-services described next characterize what may be thought of as standard services; they do not attempt to describe the diversity of m-services.

i *Traffic coordination and management.*
  Based on positional data of the subscribers to a service, the service may identify traffic jams and determine the currently fastest route between two positions, and it may give estimates and accurate error bounds for the total travel time. It also becomes possible to automatically charge fees for the use of infrastructure such as highways or bridges (termed road-pricing and metered services).

ii *Location-aware advertising in general.*
  Consumers may receive sales information for locations close to them when they indicate to the service that they are in "shopping-mode." Positional data is used together with an accumulated user profile to provide a better service, e.g., ads that are more relevant to the user.

iii *Integrated tourist services.*

This covers the advertising of the available options for various tourist services, including all relevant information about these services and options. Services may include *over-night accommodation* at camp grounds, hostels, and hotels; *transportation* via train, bus, taxi, or ferry; *cultural events*, including exhibitions, concerts, etc. For example, this latter kind of service may cover opening-hour information, availability information, travel directions, directions to empty parking, and ticketing. It is also possible to give guided tours to tourists, e.g., that carry on-line "cameras."

iv *Safety-related services.*

It is possible to monitor tourists traveling in dangerous terrain, and then react to emergencies (e.g., skiing or sailing accidents). It is possible to offer senile senior citizens more freedom of movement.

v *Position-varying information in industrial environments.*

One example of this is managing fleets of vehicles, e.g., taxis, to determine which vehicle should be assigned to a certain task, e.g., ten minutes from now. Another example is the management of free-moving, autonomous robots in industrial settings.

3 M-Service Data Warehouses

A brief introduction to data warehousing is followed by an example of a specific data warehouse, which is then used for illustration in the next section.

3.1 Introduction to Data Warehousing

A data warehouse is a large data repository that integrates data from several sources into structures expressly designed for *analytical* purposes. Data warehouses typically employ a *multidimensional model* for organizing data [15]. This type of model typically categorizes data as either business *facts* with associated *measures*, which are numerical in nature, or *dimensions*, which characterize the facts and are mostly textual.

For example, in a retail business, *products* are sold to *customers* at certain *times* in certain *amounts* at certain *prices*. A typical fact would be a *purchase*. Typical measures would be the *amount* and *price* of the purchase. Typical dimensions would be the *location* of the purchase, the type of *product* being purchased, and the *time* of the purchase.

Each dimension is organized into a hierarchical structure of *levels*, which enables the aggregation of facts to the desired levels of granularity. For example, the Time dimension may have levels *Day*, *Month*, *Quarter*, and *Year*. A prototypical query applies an aggregate function, such as SUM, to the facts characterized by specific values from the dimensions, e.g., to obtain the totals of sales by Month, Product Group, and State.

3.2 M-Service Data Warehouse Case Study

We proceed to describe a small case study of an m-service data warehouse, which stores *requests* made to an m-service and makes them available for analyses. A diagram describing the schema of the warehouse is seen in Fig. 1.

Two of the five dimensions of this data warehouse concern time. The Date dimension captures the date the request was made. It has the *levels* Day, Week, Month, Quarter, and Year, as well as a $\top$ level that represents *all* dates. The hierarchy illustrates how the different levels *roll up* to one-another, e.g., days roll up to weeks and months, but weeks do not roll up into months. The Time Of Day dimension captures the time during the day the request was made. It has been separated from the Date dimension to allow for analyses regarding the time of day across several days.

The Content dimension of the warehouse captures the content of the service being requested, e.g., sports news or concert information. The precise type of content, e.g., Sports News is grouped into Content Types, e.g., News. The User dimension captures which user made the request, and additionally groups users into User Groups. The Location dimension captures the location from which a service was requested. Some user locations can be obtained very precisely, e.g., via GPS, while others can only be obtained at the level of the *cells* in the cellular transmission network.

The warehouse has three measures. The Number Of measure captures the number of requests made for a particular combination of dimension values. The Dwell Time records the number of seconds the user looked at the response to the request before making a new request. The Delivery Time measure captures the number of seconds used to serve the requests for the given combination of dimension values.

The warehouse is used for answering questions about the connection between the user, the user's location, the time of day, and the content of the service being requested. This can be used for providing customized services, e.g., custom content specific to the interest of the user, the user's location, and the date and time of day.

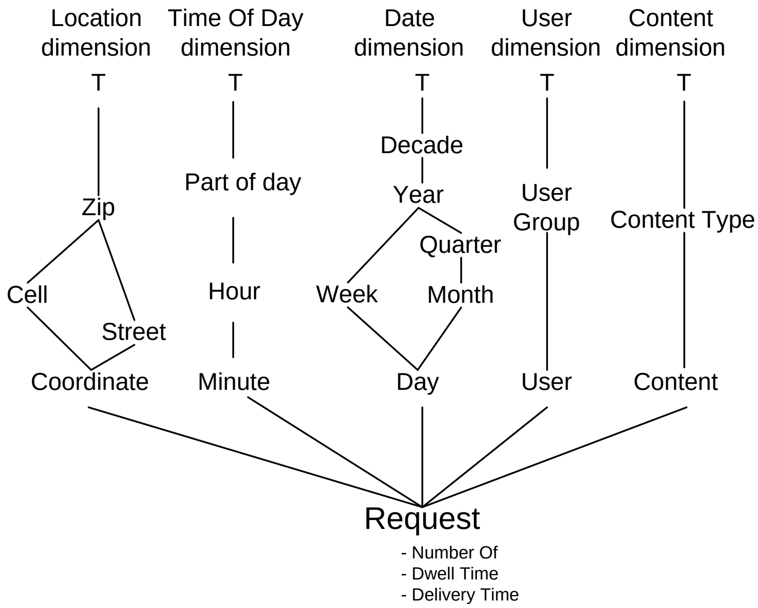

Fig. 1. Schema of M-Service Data Warehouse Case Study

4 M-Service Challenges to Data Warehousing Technologies

We proceed to describe a number of challenges, organized into six subsections, posed to data warehousing technology by m-services. As we shall see, location management in relation to moving users, which is a key ingredient in m-services, underlies many of the challenges to be described.

4.1 Support for Non-Standard Dimension Hierarchies

In traditional data warehouse environments, the dimension hierarchies must have a very regular structure; more precisely, they must be *balanced trees*. However, m-service warehouses often embody *irregular* dimensions. This occurs when the data is too complex to be captured using standard hierarchies. As an example, consider the instances of the Location dimension in Fig. 2. This hierarchy is irregular in two ways. First, the hierarchy is *non-strict* [19, 22] as cell 1001 is a

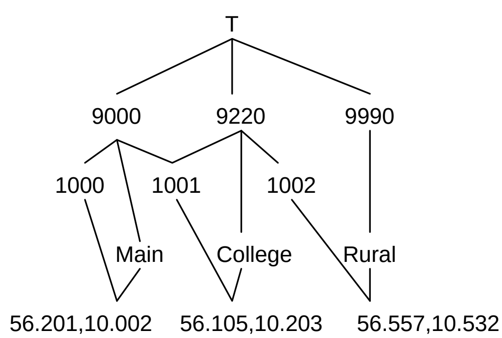

Fig. 2. Instances of the Location Dimension

child of more than one Zip value, namely the two Zip's 9000 and 9220. Second, the hierarchy is *non-onto* [22] as ZIP 9990 do not have any cells in it (this area is a small off-shore island not covered by the cellular network).

Irregular hierarchies pose two prominent problems. First, these properties cannot be captured using the standard multidimensional data models used in data warehouses currently. This means that m-service warehouse designers cannot specify the dimensions that they desire and that the problem of incorrect aggregation of data (double-counting or not counting certain values) [22] may occur. Second, current optimization techniques, which are essential to providing adequate query performance in very large data warehouses, require regular hierarchies.

It follows that there is a great need for both data modeling and implementation techniques that support irregular dimension hierarchies. Only little previous research has addressed the issue of modeling irregular hierarchies [19, 22] and implementing systems with such hierarchies efficiently [21]. It remains to be seen whether this research initial is effective in the m-service setting.

4.2 Support for Imprecision and Varying Precision

Imprecision is a fundamental aspect of central m-service data. This perhaps most prominently to the locations of users.

User locations are sampled according to some specific protocol. As examples, (i) the users may disclose their locations when they desire a service, (ii) they may supply their positions at regular intervals, or (iii) they may keep track of where the service thinks they are and then issue position samples to the service exactly when necessary in order for the service's record to be within a certain threshold of the actual position. In this latter example, the threshold is dependent on the specific service desired. As a result of the sampling, complete traces of the users' movements are unavailable; rather, the service only knows the locations of the users at discrete times.

Additionally, the samples themselves are imprecise [23]. The sample imprecision is dependent on the technology used and the circumstances under which a specific technology is used. For example, the cellular infrastructure itself, the positioning technologies offered by companies such as SnapTrack and Cambridge Positioning Technologies, and GPS and server-assisted GPS offer quite different precisions. And, for example, GPS technology is dependent on lines of sight to several satellites, which affects the robustness of the technology. In other words, the accuracy of the positioning is highly dependent on the user's location.

In the case study, the locations captured by the Location dimension are imprecise, and this imprecision must be taken into account in the data representation and must be handled intelligently during querying. If a precise (but incorrect) trace is maintained for each user, the users querying the warehouse may make suboptimal decisions based on wrong positional information. On the other hand, if an overly imprecise record of the positions is kept, the warehouse also offers suboptimal support for queries.

Another challenge related to imprecision is that the data in an m-service warehouse often has *varying precisions*. In the case study, some locations are obtained using GPS and can thus be obtained at the very precise level of Coordinate (within 10 meters). However, for the user without a GPS receiver, the location can only be obtained using the cellular network, where the most precise location is approximately that of a cell; this may lead to positioning that is imprecise in the range of hundreds of meters or the range of kilometers. It should be possible to represent data with such varying precisions, and queries have to give meaningful answers, even though the data has this highly varying level of precision.

There has been quite a lot of research in the management of imprecise data in general databases [6], as well as some initial research on the handling of imprecision and varying precision in multidimensional data models [20, 22]. Additional research is needed to support the complexities of m-service data warehousing, e.g., the handling of imprecision and varying precision in queries about complete *user traces* rather than single points.

4.3 Support for Movement Constraints, including Transportation Networks

The movements of the users of m-services are often subject to various constraints. These constraints fall into two broad categories.

Blocking objects Objects may block the movement of users. As examples, the movements of skiers are blocked by terrain without snow and the movements of sea vessels are blocked by land.

Networks The movement of objects may be constrained by networks. Such networks encompass transportation networks, including road and rail networks, and the infrastructures of buildings consisting of rooms connected by walkways and stairways.

Both kinds of movement constraints may be used in the positioning of the users. Knowing such constraints may enable a service to better estimate the position of a user, which may lead to more accurate position information or a lower sampling rate.

Transportation networks[1] are particularly interesting. When a user's movement is constrained to such a network, the movement is effectively constrained to a space with a lower dimensionality [17]. For example, if a user is moving on a road and is interested in advertisements of sales, the problem of finding the locations of sales nearest to the user may in a

[1] Note that a transportation network is different from the mathematical notion of a (directed) graph. In a transportation network the (geographical) location of the nodes is significant, while the notion of position of the nodes in a graph is non-existent.

sense be reduced to a one-dimensional problem, even if the user is really moving in what is perceived as a two-dimensional world. There is potential for exploiting transportation networks in m-services.

In the case study, we would maintain a representation of the infrastructure, in which movement occurs, separately from the data warehouse proper. And we would attempt to determine the changing transportation modes of the users. For example, a user may be walking, using public transportation, or driving a car. The use may indicate the mode to the service, or the service may determine the mode based on the user's location, speed, or general movement pattern.

4.4 Support for Continuous Change in Indexing and Precomputation

Two classes of implementation techniques are essential to ensuring adequate performance of querying, namely indexing and precomputation (the terms caching and materialized views have also been used in this context). We consider the challenges posed to these by the continuous change of user locations.

Indexing. Indexing is a fundamental technique in data management, as it makes it possible to locate desired data items in very large databases efficiently, typically in time logarithmically proportional to the total number of data items. Because data access is increasingly becoming a bottleneck, indexing techniques are becoming increasingly important. (The improvement in the rate of transfer of data between disk and main memory is slow in comparison to the growth in disk capacities and processor speeds.)

Traditional indexing techniques work only for static data, meaning that the indices have to be explicitly updated when changes occur to the data. For large, continuous datasets, e.g., those capturing the positions of moving users, the constant updating of the indices would require very large computing resources, rendering the use of indices either impractical or totally impossible. Or, alternatively, the large volumes of updates and the mechanisms that regulate the concurrent use of the indices would combine to block the querying of these structures, also rendering them useless.

It is a fundamental challenge how to obtain, if at all possible, the well-known and widely relied upon benefits of indexing when the data being indexed change continuously. The indexing of continuous data represents a fundamental challenge in m-services and beyond.

Precomputation. Precomputation is an essential technique in data warehousing, where it is used to give fast answers to queries that involve very large amounts of data. An example application of precomputation would be to compute and store the total uses of a service by county and month from all uses of the service, which are initially registered in the service's click stream and are subsequently captured in the data warehouse.

This enables fast answers to queries that ask for the number of uses of the service, e.g., by month alone, by county alone, or by quarter and county in combination. The answers may be derived from the precomputed results alone; access to the bulks of data in the data warehouse is not needed.

However, precomputation has traditionally assumed static data, meaning that precomputation has problems that are to some extent similar to those of traditional indexing techniques. A fundamental problem is then how to apply precomputation when continuous change occurs in the data. Solutions to this problem will have wide applicability.

Accommodating Continuity. Two general approaches may be taken towards accommodating continuity. Techniques may be applied that (i) create less updates, or (ii) the existing techniques may be enhanced to support rapid, non-bulk update.

One example of the former are the representation of the movement of a user by a position and a velocity vector instead of simply by a position. With the velocity vector available, the position needs only be updated when this vector changes, which generally leads to less updates. Another example was given in Section 4.3, were a transportation network was utilized to reduce the need for updates.

Many techniques may play a part in supporting rapid updates. For example, buffer techniques [1] may be applied. Briefly, these remedy the inefficiencies of transferring blocks with little data between main memory and disk by buffering updates.

Some research has been done on data cubes for *dynamic environments* [7–9]. However, this research deals only with rather simple queries, which do not support m-services adequately. Also, no support for continuously changing data is provided.

Another interesting research direction is *on-line aggregation* [11, 13, 24, 25], where the results of aggregate queries are approximated using sampling techniques and gradually refined, meaning that no pre-computed aggregates have to be updated when the data changes. The applicability of these techniques for the complex queries needed in m-services is unclear, and the time it takes to offer sufficiently precise answers may not be short enough to support m-service requirements.

4.5 Support for Closed-Loop Usage

Data warehouses have traditionally had *human users*, such as data analysts. In an m-service setting, the most important class of "users" are the *m-service systems* themselves. These use the information in the warehouse, e.g., information about a user's hotel preferences, for customizing their interaction with the human users of the service. Thus, data flows from the operational systems to the warehouse, and information flows from the warehouse back to the operational systems, without human intervention.

This usage paradigm has been termed "closed-loop" data warehouse usage. Key challenges set forth by closed-loop usage are presented next.

Faster Response Time. It is a non-trivial challenge to support different response-time requirements for different "user" classes. A human user may be content with a 10–20 second response time for queries, as it often takes on the order of minutes to to analyze the result of a query. In this case, 10–20 seconds do not add significantly to the total time spent. However, if the "user" is an m-service that must deliver a response to an end user within a second, a 10–20 second response time is clearly unacceptable.

A system is needed that can guarantee a fast, sub-second response time when needed, e.g., when the user is an on-line m-service, or compute the result "on the side" with relatively low priority if a longer response time is permissible, e.g., if the user is a human or a long-running batch process.

An interesting new possibility in the m-service setting is to use *default answers*, meaning that the "real" result of a query may not need to be computed if it is not available in fast storage. For example, a query may ask for a custom greeting for a particular user based on the user's previous behavior. If it will take too long to compute this, it is still useful to return a default greeting, or one of a small set of default greetings, based on, e.g., the user's age and gender, or based on an earlier location of the user.

The solution could be similar to the "hot response cache," proposed by Kimball [16], that pre-computes and stores results to oft-asked queries in a fast cache. However, a solution is desired that is more advanced than both this and the traditional data warehouse caching techniques [10, 12], in at least three respects.

First, it should be self-tuning, adjusting itself to the queries asked without human intervention. Second, it should support guaranteed response times and integrate this with the handling of default results. Third, it should support more advanced queries, e.g., complex computations of user preferences, rather than the simple aggregate queries supported by current warehouse caching systems.

A little previous research has dealt with the issue of self-tuning caching [5, 18], but current solutions offer neither guaranteed response times with default values nor support for the advanced, complex queries needed in m-services.

Continuous Operation. Traditionally, data warehouses have operated in a *two-mode* fashion [15]. In *query mode*, the warehouse is read-only and cannot be updated. In *update mode*, the warehouse is write-only and is not available for querying. The warehouse enters update mode during off-hours, e.g., during the night or the weekend, where a large number of updates are processed in one step, a so-called *bulk load*.

However, m-services (and e-services in general) operate in a 24-by-7 environment where the service must *always* be available. To properly support the closed-loop usage described above, the warehouse must always operate in the same way—*single-mode* operation must be supported, where the warehouse can be updated while simultaneously being queried.

Note that this does not represent a return to a traditional DBMS solution, as the warehouse still only has one process updating the data. Thus, there is no need for the full complexity of traditional concurrency control. Rather, techniques that takes the special characteristics of the m-service warehouse (such as many large reads, at most one (large) update at a time) into account while preserving the high performance associated with the traditional bulk-loading techniques must be developed.

4.6 Support For Dynamic Services

We expect the m-service context of data warehouse use to be considerably more *dynamic* than the contexts in which data warehouses have traditionally been deployed. M-services must evolve very quickly to meet new, changing, or unanticipated user needs. This leads to several challenges to data warehousing technology.

First, the content of some warehouse dimensions can evolve very quickly. For example, the dimension capturing user interest must evolve in *near real-time* to support the rapidly changing interests of the users. When a major event suddenly occurs, a substantial amount of users are likely to show great interest in it. Such *rapidly changing dimensions* pose a significant challenge to data warehouse technology, which assumes that dimensions are quite stable ("slowly changing," to use Kimball's term [15]), where the rate of change is measured by the week or month.

Second, the *schema* of a data warehouse must be updated often in order to support new kinds of services that cannot be handled with the existing design. This must occur quickly and seamlessly, preferably without taking the warehouse off-line, so that continuous operation is supported. Further, the schema must be changed in such a way that all existing queries, reports, and applications can still be run and give the same answers. Thus, support for *backwards-compatible schema evolution or versioning* is in high demand. With traditional data warehouse technology, schema evolution is a cumbersome and error-prone process, as is schema versioning.

Finally, the data sources that feed an m-service warehouse are often highly distributed, which is necessary in order to provide fast, local service. However, the distributed setting means that data are not always available to the central warehouse. The flow of data into the m-service warehouse will often be delayed, and some data will only make it into the warehouse with a delay. Thus, queries will often be run on incomplete data, which will in some cases lead to erroneous decisions that will later have to be reversed. This renders it very important to know exactly what data was available in the warehouse at each past point in time. This in turn calls for so-called *transaction-time* support. Transaction time is not supported by current data warehouse technology, and only very little research has been done on how to support it efficiently [2].

5 Summary

Mobile e-services is rapidly emerging as a prominent area of deployment of information technology. It is argued that data warehousing technologies may play a prominent role in m-services, but that these technologies must meet new and tough challenges in order for them to reach their full potential in the area of m-services.

The paper briefly describes the advances in the underlying hardware technologies that combine to enable m-services. It then exemplifies the moving objects that m-services concern, describes the general scenario in which the services may exploit data warehouse technologies, and gives what may be considered typical examples of specific m-services. Following an introduction to data warehousing and a case study, the remainder of the paper is devoted to the discussion of challenges to data warehousing. Table 1 summarizes these.

challenge	described in section
Support for irregular dimension hierarchies—non-strict hierarchies	4.1
Support for irregular dimension hierarchies—non-onto hierarchies	4.1
Support for imprecision of data	4.2
Support for varying precision of data	4.2
Support for movement constraints, including transportation networks	4.3
Support for continuous change in indexing and precomputation—update reduction	4.4
Support for continuous change in indexing and precomputation—faster updates	4.4
Support for closed-loop usage—defaults and faster response time	4.5
Support for closed-loop usage—continuous operation	4.5
Support for dynamic services—rapidly changing dimensions	4.6
Support for dynamic services—backwards-compatible schema evolution and versioning	4.6
Support for dynamic services—transaction time	4.6

Table 1. Summary of M-Service Challenges to Data Warehousing

Several of the challenges concern the support for more advanced types of data. Location information, which is central to m-services, serves well to exemplify these. Other requirements stem from the not only frequent, but continuous movement that the users of m-services are capable of. Next, the paradigm of using a warehouse in an off-line fashion falls short. In m-services (and other areas as well!), it is desirable that data be fed into the warehouse without any delay and that the warehouse is available for the m-service systems to query without human interaction at all times. This calls for faster response time and continuous operation. Finally, the dynamic nature of m-services are expected to pose new challenges.

It is hoped that these challenges will spur researchers to formulate more concrete challenges, to which they can subsequently develop new solutions. It should also be emphasized that the challenges brought forward here reflect the backgrounds of the authors; it is likely that other authors will emphasize other, perhaps equally valid, challenges.

Acknowledgments

The authors would like to thank Christian Hage of Euman.com and employees of the company Mindpass.com for sharing their insights into location-based and e-services with us.

This work was supported in part by the Wireless Information Management network, funded by the Nordic Academy for Advanced Study through grant 000389, by the Danish Technical Research Council through grant 9700780, and by a grant from the Nykredit Corporation.

References

1. L. Arge. The Buffer Tree: A New Technique for Optimal I/O-Algorithms (Extended Abstract). In *Proceedings of the Fourth International Workshop on Algorithms and Data Structures*, pp. 334–345, 1995.
2. R. Bliujūtė, S. Šaltenis, G. Slivinskas, and C. S. Jensen. Systematic Change Management in Dimensional Data Warehousing. In *Proceedings of the Third International Baltic Workshop on DB and IS*, pp. 27–41, 1998.
3. J. Clifford, C. Dyreson, T. Isakowitz, C. S. Jensen, and R. T. Snodgrass. On the Semantics of "Now" in Databases. *ACM Transactions on Database Systems*, 22(2):171–214, June 1997.
4. H. D. Chon, D. Agrawal, and A. El Abbadi. Storage and Retrieval of Moving Objects. In *Second International Conference on Mobile Data Management*, pp. 173–184, 2001.
5. P. Deshpande, K. Ramasamy, A. Shukla, and J. F. Naughton. Caching Multidimensional Queries Using Chunks. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data*, pp. 259–270, 1998.
6. C. E. Dyreson. A Bibliography on Uncertainty Management in Information Systems. In A. Motro and P. Smets (eds.), *Uncertainty Management in Information Systems—From Needs to Solutions*, Kluwer Academic Publishers, pp. 413–458, 1997.
7. S. Geffner, D. Agrawal, A. E. Abbadi, and T. R. Smith. Relative Prefix Sums: An Efficient Approach for Querying Dynamic OLAP Data Cubes. In *Proceedings of the Fifteenth International Conference on Data Engineering.*, pp. 328–335, 1999.
8. S. Geffner, M. Riedewald, D. Agrawal, and A. E. Abbadi. Data Cubes in Dynamic Environments. *IEEE Data Engineering Bulletin*, 22(4):31–40, 1999.
9. S. Geffner, D. Agrawal, A. E. Abbadi. The Dynamic Data Cube. In *Proceedings of the Sixth International Conference on Extending Database Technology*, pp. 237–253, 2000.
10. H. Gupta. Selection of Views to Materialize in a Data Warehouse. In *Proceedings of the Sixth International Conference on Database Theory*, pp. 98–112, 1997.
11. P. J. Haas and J. M. Hellerstein. Ripple Joins for Online Aggregation. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data*, pp. 287–298, 1999.
12. V. Harinarayan, A. Rajaraman, and J.D. Ullman. Implementing Data Cubes Efficiently. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data*, pp. 205–216, 1996.
13. J. M. Hellerstein, P. J. Haas, and H. Wang. Online Aggregation. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data*, pp. 171–182, 1997.
14. X. Hong, T. J. Kwon, M. Gerla, D. Lihui Gu, and G. Pei. A Mobility Framework for Ad Hoc Wireless Networks. In *Second International Conference on Mobile Data Management*, pp. 185–196, 2001.
15. R. Kimball. *The Data Warehouse Toolkit*. Wiley, 1996.
16. R. Kimball and R. Merz. *The Data Webhouse Toolkit*. Wiley, 2000.
17. G. Kollios, D. Gunopulos, and V. J. Tsotras. On Indexing Mobile Objects In *Proceedings of the PODS Conference*, pp. 261–272, 1999.
18. Y. Kotidis and N. Roussopoulos. DynaMat: A Dynamic View Management System for Data Warehouses. In *Proceedings of the ACM SIGMOD International Conference on the Management of Data*, pp. 371–382, 1999.
19. T. B. Pedersen and C. S. Jensen. Multidimensional Data Modeling for Complex Data. In *Proceedings of the Fifteenth International Conference on Data Engineering*, pp. 336–345, 1999.
20. T. B. Pedersen, C. S. Jensen, and C. E. Dyreson. Supporting Imprecision in Multidimensional Databases Using Granularities. In *Proceedings of the Eleventh International Conference on Scientific and Statistical Database Management*, pp. 90–101, 1999.
21. T. B. Pedersen, C. S. Jensen, and C. E. Dyreson. Extending Practical Pre-Aggregation in On-Line Analytical Processing. In *Proceedings of the Twentyfifth International Conference on Very Large Databases*, pp. 663–674, 1999.
22. T. B. Pedersen, C. S. Jensen, and C. E. Dyreson. A Foundation for Capturing and Querying Complex Multidimensional Data. In *Information Systems*, special issue on data warehousing, 42 pages, to appear, 2001.
23. D. Pfoser and C. S. Jensen. Capturing the Uncertainty of Moving-Object Representations. In *Proceedings of the Sixth International Symposion on Spatial Databases*, pp. 111–132, 1999.
24. M. Riedewald, D. Agrawal, and A. E. Abbadi. pCube: Update-Efficient Online Aggregation with Progressive Feedback and Error Bounds. In *Proceedings of the Twelfth International Conference on Scientific and Statistical Database Management*, pp. 95–108, 2000.
25. M. Riedewald, D. Agrawal, and A. E. Abbadi. Flexible Data Cubes for Online Aggregation. In *Proceedings of the Eighth International Conference on Database Theory*, pp. 159–173, 2001.
26. Z. Song and N. Roussopoulos. Hashing Moving Objects. In *Second International Conference on Mobile Data Management*, pp. 161–172, 2001.