Towards a Cost Model for Distributed and Replicated Data Stores

Heinz & Kurt Stockinger
CERN, European Organization for Nuclear Research Geneva, Switzerland
Inst. for Computer Science and Business Informatics University of Vienna, Austria
Heinz.Stockinger@cern.ch, Kurt.Stockinger@cern.ch

Erich Schikuta
Inst. for Computer Science and Business Informatics
University of Vienna, Austria
schiki@ifs.univie.ac.at

Ian Willers
CERN, European Organization for Nuclear Research,
Geneva, Switzerland
Ian.Willers@cern.ch

Abstract

Large, Petabyte-scale data stores need detailed design considerations about distributing and replicating particular parts of the data store in a cost-effective way. Technical issues need to be analysed and, based on these constraints, an optimisation problem can be formulated. In this paper we provide a novel cost model for building a world-wide distributed Petabyte data store which will be in place starting from 2005 at CERN and its collaborating, world-wide distributed institutes. We will elaborate on a framework for assessing potential system costs and influences which are essential for the design of the data store.

1 Introduction

With the growth of the Internet in the last couple of years and expanding technologies in database research, data warehousing, networking and data storage, large distributed data stores with data amounts in the range of Petabytes are emerging [16]. Not only the choice of the optimal data storage system (relational or object-oriented databases, flat files, etc.) is essential but also the placement of certain parts of the data store has a major influence on the response time of the entire system. Hence, an analysis of costs and influences of replicated data stores is required which helps to solve the data placement optimisation problem.

The response time for serving multiple user requests is a major performance factor of a data store. We require a high throughput rather than a high performance system which just serves a single user per time. High throughput requires several factors to be considered and optimised.

In the database community, replication research mostly deals with update synchronisation of synchronous or asynchronous replicas [1, 4, 6, 7, 15, 21]. Read or write transactions are normally rather small and hence a small transfer of data is done over the network. [20] proposes a performance model for telecommunication applications which have the previously stated features regarding transactions. The main assertion is that a fully replicated data store is suboptimal since the update propagation to each replica takes a long time and hence decreases the response time for write transactions. However, partially replicated data stores do not provide optimal read response times since only parts of the data items are replicated. The trade-off between optimising the response time for read and write transactions by considering highly consistent replicas has to be found.

In our paper we want to address large scientific data sources which are mostly read-only. Furthermore, transactions are very large and write data of about one Gigabyte within a single transaction. Consistency of data is not as high as in a real-time distributed database management systems. This imposes different requirements which have to be addressed. The contribution of this paper is to analyse influences and costs of very large, replicated data stores over the wide-area network which are not addressed in database research. These can be directly used in the design of such data stores. Much emphasis is put on the placement of data items and the influence on the response time of the data store. A detailed cost model and some tradeoffs are given.

The paper is organised as follows. Section 2 gives an overview of related work and points out the main differences of replication methods in current database research to our approach. In Section 3 we state the assumptions for our model and introduce the CERN environment. The location of the data store is discussed in Section 4. Section

5 elaborates on details for costs and influences which are summarised and discussed in Section 6. Conclusions are given in the final section.

2 Related Work

Related work can be found in performance modelling of distributed systems and distributed database systems. However, a fully integrated cost model which covers the distributed data store as well as distributed system features is not discussed. Data servers can be modelled as queueing networks [17] where single transactions are analysed and represented by their arrival rate at the data server. Large data sizes are not mentioned.

A very detailed analytical queueing model can be found in [20] which uses telecommunication applications for replication performance models. An important feature of this work is the identification of throughput, bottlenecks and the correct placement of replicas. Access patterns and workloads are modelled by arrival times of transactions. A detailed study on replicating some objects to some sites is presented. Update transactions happen rather frequently. Therefore, the degree of replication can have a negative influence on the response time of the system since there is an increasing update overhead as the amount of replicas is increased.

3 Assumptions for the Model

Since there are several possible ways to replicate data specific to a particular problem domain, we want to limit our discussion to scientific data intensive computing environments like the CERN's DataGrid [13, 9] or GriPhyN [14]. Both projects deal with the placement and replication of and access to large amounts of data. Hence, our model is driven by the High Energy Physics community. We will now give a short background of the data management problem.

3.1 The CERN Environment

In 2005 CERN, the European Organization for Nuclear Research, will have in place a new particle accelerator and detectors which will store information about particle collisions in huge amounts of data similar to [2]. A single detector like the Compact Muon Solenoid (CMS) will produce about 1 Petabyte of data per year during a lifespan of about 15 to 20 years. Once data are written centrally at CERN, they have to be distributed and replicated to world-wide distributed sites, called Regional Centres (RC), in order to enable distributed and fast user analysis for some 2000 researchers. The computing model of a typical experiment is shown in Figure 1.

One characteristic of these data is that they are read-only. In general, data are written by the experiment, stored at very high data rates (roughly 100 MB/s) and are normally not changed any more afterwards. This is true for about 90% of the total amount of data.

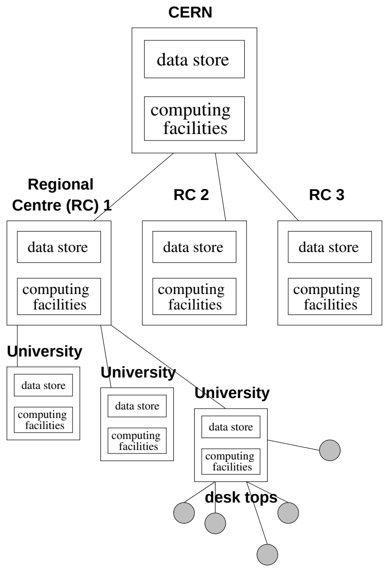

Figure 1. Example of the network of one experiment's computing model

These RCs are part of the distributed computing model and should complement the functionality of the CERN Centre. Regional Centres will be set up in different places around the globe.

3.2 CMS Specific Data Transfer Issues

In High Energy Physics the data production system is rather complicated and consists of several stages which result in different forms of data [12]. In general, we can distinguish between *raw data* generated by the detector, *reconstructed physics data* and *tag summary data*. The amount of raw data produced per year will be about 1 PB. The amount of reconstructed data will be around 500 TB per year. Since the experiment will run for about 100 days per year, roughly 5 TB of reconstructed data will be produced per day. The amount of reconstructed data to be transferred to a single RC is 200 TB per year.

[8] foresees a 622 Mbps ATM network link between CERN and a Regional Centre as the minimal requirement to satisfy the data transfer needs. This corresponds to a maximum transfer rate of 77.75 MB/s which most probably cannot be reached in reality. We assume an average bandwidth of 30 MB/s. This is a rather conservative assumption

whereas the generally accepted payload is 50%-60% [5], which corresponds roughly to a maximum of 2.6 TB to be transferred over the network link per day.

However, if we look at the conservative case of 30 MB/s, we see that the bandwidth is nearly totally used for data transfer to one Regional Centre, and does not leave much space for other network activities.

3.3 General Assumptions

For our cost model we have the following input parameters: Physicists are working at distributed locations in the world requesting access to the data store. We further assume a client-server model where a data store has one or more *data servers*. The time which elapses between sending the request and retrieving the required data locally is defined as the *response time* of the system. The location of the client application has an important impact on the response time because a LAN normally has a much higher bandwidth than a WAN. This results in a shorter response time which we consider as the major performance measurement for the data store.

The response time for read-only data is optimal when all data are available at each site. If the costs of storing replicated data and the response time still should be kept, certain queries and their impact on data access have to be considered.

4 Location of the Data Store

4.1 Centralised Data Store

Given a centralised data store with a single data server, all the incoming requests from application clients have to be served by the centralised data store. All users have to send their requests to the single data server regardless if the user runs the application client over a LAN or a WAN. The consequence is that the response time for WAN users will mostly be considerably higher than for LAN users. For a large number of concurrent users a single data server can easily become a bottleneck for a system with a centralised data store.

4.2 Distributed and Fully Replicated Data Store

Based on the fact that many physicists will do their analysis work at Regional Centres, we can identify a distribution of users at certain centres, an inherent distribution of application clients and hence also a distribution and replication of data. According to the principle of *data locality* we do not want that users access data via the WAN. Instead, data should be located such that users can access data in the same LAN.

The availability of several data servers and multiple copies of data items result in a relatively small load for each single data server. Thus, there is a reduction of the bottleneck of a single, central data server, because of the distribution of requests to several data servers. In terms of user response time it is certainly better to provide multiple copies of data at several distributed sites in order to have data as close as possible to the user.

Based on our model assumption and the distribution of users, the distributed data store is preferable.

5 Costs and Influences of the Data Store

After this introductory model assumption and a short discussion on the location of the data server, let us now have a more detailed look at data locality and how this can influence the response time of a system. We will distinguish between *system influences* on the response time (static and dynamic ones) and *economically relevant costs* that appear in distributed systems and discuss additional sources of influence.

5.1 Static System Influences

Static system influences are supposed to be stable and do not change over time. In general, these influences are easy to measure compared to the dynamic system influences.

Bandwidth of the network link: The physical network connections are of major importance for a WAN. However, since we are referring here to static system influences, the current network load is not considered here but falls into the discussion of dynamic system influences.

Size of data to be transferred: The size of data to be transferred has an influence on the time it takes to transfer data from one point to another one. There is a direct proportional relationship between the transfer time and the data size. Based on the amount of data to be transferred and the time constraint when the data have to arrive, the network can be a bottleneck. When the data size increases, not only the storage capacity (which grows as the amount of replicas grows) is increased, but also the the network bandwidth utilisation grows since more data has to be transferred over the network when data items are replicated. Based on the data requirement per time it is important to determine if it is possible at all to transfer the required data amounts over the network link. An alternative is to transfer parts of the requested data by normal *"air mail"* via magnetic tape.

Physical distance between two endpoints: This is can be measured by the *round trip time (RTT)* of a single data packet.

Network protocol: The influence of the physical distance in the network cannot be measured directly. Although the RTT tells us how long it takes to send a single data packet over the network, the deployed network protocol (TCP, UDP, FTP, ...) influences the response time very much. If a protocol implements many acknowledgements, the RTT delays the overall time to send the requested data. Hence, the selection of a reliable protocol with many acknowledgements has to be compared to an unreliable protocol with less or no acknowledgements.

5.2 Dynamic System Influences

These are influences that change dynamically and require a constant monitoring system.

Current bandwidth - network load: In a multi-user environment there is a high factor of uncertainty for the currently available bandwidth for a given time. This is very much dependent on external influences and also on peak times (for instance users in different time zones) when the network is utilised heavily. Bandwidth reservation and the usage of dedicated lines can significantly contribute to reduce the factor of uncertainty for the throughput. However, this comes again at an additional cost and network bandwidth reservation can only be used for particular network technologies e.g. ATM networks.

Current "data server load" and disk speed: Not only a single data server can be a bottleneck that reduces the response time, but even the speed of the disk subsystem can have a negative impact on the performance of the system. We also have to take into account the number of jobs waiting to be served by the data server.

5.3 Application Dependent Influences

Access patterns: Replication can be regarded as a form of caching which reduces the amount of data traffic over the network. The *access patterns* of the users and hence the amount of data to be requested within a particular amount of time have to be considered. When we assume that we have n sites and therefore n times as much as data in the entire system (we assume a fully replicated system with a replica of each data item at each site), one data item has to be requested on average n times in order to profit in terms of network traffic reduction from a distributed and replicated system.

Data consistency: The *data consistency factor* is the time delay with which copies of data or updates arrive at a site and a fully equivalent copy is created. The lower the factor, the less up-to-date is a remote site. This factor, also known as *relaxed coherency* [10], has to be agreed on by the database administrators and the users, and is also very much dependent on the available network bandwidth. In other words, the user has to tell what amount of consistency is required and this has to be achieved by possible replication mechanisms. For instance, we can identify rather loose consistency where update synchronisation takes place every minute, hour or even day. If strict consistency is required, replica update and synchronisation is done in real time and immediately after changes have been made.

In addition, distributed data stores have to exchange communication messages for (meta) data updates. Even if all the data are read-only, they have to be introduced to each distributed data store at least once. Since multiple updates can occur at the same time in a distributed, multi-user system, global transactions have to ensure data consistency at multiple sites [11].

Degree of replication: Whereas in a fully replicated system each data item is replicated to 100%, in a partially replicated system certain data items can occur only once while others occur multiple times. The degree of replication represents the percentage of how often a data item is replicated. The exact placement of a data item and the amount of replicas for each data item has to be optimised and highly depends on the access patterns and the data consistency factor.

Quantity and quality of data: Quantity and quality of data can represent different costs for a data location cost model. Even if a large quantity of data is frequently requested, it does not necessarily mean that these data always have to be replicated, e.g. there may be a smaller set of data that in relation is not as frequently requested as others but has a higher quality and thus higher priority for the final result of the query and/or the computation. For example, data that contain the Higgs boson appear not very frequently (low quantity) but are very important for physics analysis (high quality) [23]. In a partially replicated system the choice of which data to be replicated has to be based on these quantity and quality constraints. A higher quality/quantity of data corresponds to a higher possibility of replicating data to several sites.

[3] points out that the main problems of generic replication solutions is that knowledge about the data is required. In order to have a correct quantity/quality tradeoff, the user input and/or some hints are required for deciding about the creation and allocation of replicas. Let us define the choice of having a certain amount of a particular set of data items the *replica selection* process.

Replica placement: The I/O requirements of users should be served as fast as possible. If most of the time of an application is spent on I/O, the requirement to replicate the requested data increases. In contrast, the need for replication decreases when very much time of the applications is spent on CPU and only small amounts of data are requested. *Application monitoring* which keeps track of user access patterns and the time spent on computation has an important influence on the optimal degree and placement of replicas.

An advanced, partially replicated system can automatically and dynamically create and delete replicas based on access patterns in order to have better response times (by creating replicas) and reduced storage space (by deleting not very frequently requested replicas). A more detailed discussion on automatic replication adapted to access patterns of users can be found in [24].

Data compression: The *compression time factor* has a negative influence on the data consistency factor since additional computing power and thus time delay will be required but this can be evened out to a large extent by the *compression factor* which results in less network bandwidth utilisation. The tradeoff between these two factors has to be found out. By an increase of computing power over time which is predicted by Moore's law [19], the compression factor becomes more and more important.

5.4 Economic Costs of a Distributed System

The advantage gained by data locality comes at the costs of other factors and constraints that have to be analysed. In particular, multiple copies of data result in more resources and more sites to be maintained. We define a *site* as a number of computing resources (CPUs) and data storage devices connected via a network link to another site. In the CERN environment a site is either CERN or a Regional Centre. The following additional costs and constraints occur in a model with multiple sites:

Infrastructure costs: These costs cover storage devices like disks and tapes, CPU and network resources that have to be put in place. Additionally, user support and maintenance of each single site, and regular replacements of devices have to be taken into account. Maintenance also covers costs for hardware and software personnel. One might claim that a distributed system in general requires network connections to remote users. We can clearly see from our application that only the distribution of users causes already additional network costs.

Policies and political influences: Finally, political considerations which constrain technically possible solutions have to be added to the technical ones mentioned above. A political constraint can be that a site only wants to be updated every 5 hours. This has to be expressed by the data consistency factor. Another example is that a site wants to get only a particular set of files and partially replicated data stores are the consequence.

6 The Cost Model

The target function of a cost model for a distributed and replicated data store is the response time of a user's request, which is the focus of the minimisation process.

We listed different influencing factors for our cost model in the preceding sections. However, we have to express these factors within a mathematical notation for further analysis.

Basically, we have to distinguish two different approaches for the resolution of a user's request, *data migration* and *application migration*.

Data migration. Hereby the data moves across the network and the application executes locally. The response time of a user's request $Time_{data\_mig}$ is the sum of 3 terms, the time for accessing local data of size D_{local}, the time for accessing remote data of size D_{remote}, plus the time $Time_{dist}$ to cover the physical distance to a remote location (which is zero in the special case if everything is accessed only locally). Each term is dependent on the replication factor, the pattern of the system and the access pattern of the user request. We assume for simplicity that the access pattern is equally distributed as the replication pattern, which allows us to express the replication simply by a replication factor rep as a fraction (from 0 to 1). Given the bandwidth for the LAN by BW_{LAN} and for the WAN by BW_{WAN} we define the following general formula

$$Time_{data\_mig} = \frac{rep \cdot D_{local}}{f_{LAN} \cdot BW_{LAN}} + \frac{f_{D_{remote}} \cdot (1-rep) \cdot D_{remote}}{f_{WAN} \cdot BW_{WAN}} + Time_{dist}$$

In this formula three influencing factors, f_{LAN}, f_{WAN} and $f_{D_{remote}}$, describe the static and dynamic system situation.

The bandwidth of the LAN is influenced by the server load $sl(t)$, a time t dependent value between 0 and 1 (0 = no influencing server load, 1 = server fully loaded) and the network protocol np (0 = no network protocol, only data transfer; 1 = only protocol, no data transfer). This leads to

$$f_{LAN} = (1 - sl(t)) \cdot (1 - np)$$

The bandwidth of the WAN is influenced by the network protocol too and, instead of the server load, by the network load $nl(t)$ (0 = no additional network load; 1 = fully loaded network, no free bandwidth). Thus

$$f_{WAN} = (1 - nl(t)) \cdot (1 - np)$$

The data size transferred from a remote size can be influenced by a data compression factor $dc(D)$ dependent on the data D, where 0 denotes no compression possible; 1 (theoretically) full compression, no data to transfer necessary. We can therefore define

$$f_{D_{remote}} = (1 - dc(D_{remote}))$$

The other named issues influencing the behaviour of the data store can be interpreted as restrictions, e.g. the relation between disk (bandwidth of the disk system BW_{disk}) and network characteristics (BW_{LAN}, BW_{WAN}) expressed by $BW_{disk} > BW_{WAN} > BW_{LAN}$, are beyond the decision of this model, e.g. policies and cost limits.

Application migration. Another approach for the request resolution is to migrate the application to a remote data store, execute the application there and send the result data to the requesting client. The time for this way of execution summaries four factors, the time for sending the application D_{app}, the time for accessing the requested data locally at the remote data store, the time for sending the result D_{result} to the client and finally the time to cover the distance between client and server. This leads to the formula

$$Time_{app\_mig} = \frac{D_{app} \cdot f_{D_{remote}}}{f_{WAN} \cdot BW_{WAN}} + \frac{D_{local}}{f_{LAN} \cdot BW_{LAN}} + \frac{f_{D_{remote}} \cdot D_{result}}{f_{WAN} \cdot BW_{WAN}} + Time_{dist}$$

The influencing factors are the same as in the formula for $Time_{data\_mig}$. The replication factor is not applicable due to accessing the data fully at the remote site.

Target function of the cost model. Summing up the target function for minimisation of the proposed cost model is the minimum of both possible approaches, which can be expressed finally by

$$Time_{request} = Minimum(Time_{data\_mig}, Time_{app\_mig}).$$

Discussion of tradeoffs concerning CMS: Assumptions and information gained through our internal studies, Monarc [18] and RD45 [22], revealed that the raw data of High Energy Physics (HEP) is in general very difficult to compress. What is more, the LAN links are expected to getting cheaper faster than the CPU gets faster. Both factors are strongly in favour of investigating more effort into higher LAN network connections rather than into research of efficient compression algorithms. However, as soon as the data are transferred over the WAN, data compression becomes a more important issue since, for instance, meta data for global access and transaction control are assumed to be much more easy compressible. In addition, the WAN connections are still far behind the capacities of CPUs and disk subsystems.

A further improvement of the throughput can be gained by the inherent relaxed coherency. In other words, the consistency of the replicas in the distributed system is not such an important issue in HEP and, for instance, updates of newly created physics algorithms do not need to be propagated in real-time. Thus, an increased replication factor yields higher data availability in case of a network failure and a higher read-performance due to local data placement.

7 Conclusion

The cost model presented in this paper is applicable for design considerations for a replicated, distributed data store. Advantages and disadvantages of a fully replicated data store are highlighted. On comparing two extreme cases of data location, a detailed discussion on partially replicated systems is given. The actual optimisation of the data location problem needs a very detailed understanding of application requirements, knowledge about the data as well as hardware and software resources. Since we base our discussions on scientific, read-only data, we use a different approach than in the database research community and state possible tradeoffs, connections and influences which occur in the design of a data store.

Acknowledgement

We want to thank Koen Holtman for his valuable discussions.

References

[1] D. Agrawal, A. El Abbadi, R. Steinke. Epidemic Algorithms in Replicated Databases (Extended Abstract). PODS pp. 161-172, 1997.

[2] J. Behnke, A. Lake. EOSDIS: Archive and Distribution Systems in the Year 2000. Eigth Goddard Conference on Mass Storage Systems and Technologies (7th IEEE Symposium on Mass Storage Systems), March 2000.

[3] Y. Breitbarth, H. Korth. Data Replication Gaining Popularity, IEEE Concurrency, June 1999.

[4] Y. Breitbart, H. Korth. Replication and Consistency: Being Lazy Helps Sometimes, In Proc. 16 ACM Sigact/Sigmod Symposium on the Principles of Database Systems, Tucson 1997.

[5] J. Bunn, personal e-mail discussion with CMS colleague, March 2000.

[6] S. Ceri, M.A.W. Houtsma, A.M. Keller, P. Samarati. Independent Updates and Incremental Agreement in Replicated Databases, Kluwer 1999.

[7] P. Chundi, D. J. Rosenkrantz, and S. S. Ravipi. Deferred Update Protocols and Data placement in Distributed Databases, In Proceedings of the International Conf. on Data Engineering, Feb. 1996.

[8] CMS Computing Technical Proposal, CERN/LHCC 96-45, December 1996.

[9] The CERN DataGrid Project: http://www.cern.ch/grid/

[10] R. Gallersdoerfer, M. Nicola. Improving the Performance in Replicated Databases through Relaxed Coherency. VLDB Conference, 1995.

[11] J. Gray, P. Helland, P. E. O'Neil, D. Shasha. The Dangers of Replication and a Solution. SIGMOD Conference pp. 173-182, 1996.

[12] K. Holtman, Prototyping of CMS Storage Management, Ph.D. thesis (proefontwerp), Eindhoven University of Technology, May 2000.

[13] W. Hoschek, J. Jaen-Martinez, A. Samar, H. Stockinger, K. Stockinger, Data Management in an International Data Grid Project, to appear in 1st IEEE, ACM International Workshop on Grid Computing (Grid'2000), Bangalore, India, Dec. 2000.

[14] The GriPhyN Project, http://griphyn.org

[15] B. Kemme, G. Alonso. A Suite of Database Replication Protocols based on Group Communication Primitives. In Proc. of the Int. Conf. on Distributed Computing Systems, Amsterdam, May 1998.

[16] A. Lake, J. Crawford, R. Simanowith, B. Koenig. Fault Tolerant Design in the Earth Observing System Archive. Eigth Goddard Conference on Mass Storage Systems and Technologies (7th IEEE Symposium on Mass Storage Systems), March 2000.

[17] D. Menasce, V. Almeida, L. Dowdy. Capacity Planning and Performance Modelling. Prentice Hall, New Jersey 1994.

[18] The Monarc Project: Models of Networked Analysis at Regional Centres for LHC Experiments, http://monarc.web.cern.ch/MONARC

[19] L. Roberts, Beyond Moore's Law: Internet Growth Trends, IEEE Computer, January 2000.

[20] M. Nicola, M. Jarke. Increasing the Expressiveness of Analytical Performance Models for Replicated Databases. International Conference on Database Theory, ICDT'99, Jerusalem, January 1999.

[21] E. Pacitti, E. Simon. Update Propagation Strategies to Improve Freshness in Lazy Master Replicated Databases. VLDB Journal 8(3-4): 305-318, 2000.

[22] RD45: A Persistent Object Manager for HEP, http://wwwinfo.cern.ch/asd/rd45/index.html

[23] L. Rurua. Possibilities for h0 $\rightarrow$ b-bbar with CMS at LHC, 5th International Conference on B-Physics "Beauty 97" Hadron Machines, Los Angeles, USA, October 1997.

[24] O. Wolfson, S. Jajodia, Y. Huang, "An Adaptive Data Replication Algorithm", ACM Transactions on Database Systems (TODS), Vol. 22(4), pp. 255-314, June 1997.