Armada: an evolving database system
Groffen, F.E.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (http://dare.uva.nl)
1

Introduction

1.1 Background

The era of huge monolithic main-frames is over. Instead, server rooms are now populated with clusters of machines that serve the workload. No longer a single system, but instead a group of systems is responsible for a particular task. Clusters typically have a bigger combined capacity that can easily be expanded by adding new systems. Having multiple systems being responsible, failure of an individual system does not stop the system as a whole, but merely leads to a graceful degradation.

Not surprisingly, there is a trend to deploy for almost every task where a computer is involved, a cluster setup involving multiple machines. However, not every task can as easily be deployed on a cluster as others. Typically, informative web-sites such as those of search engines, can be served off any server that has a replica of the original server’s contents. Periodical updates to the contents are easy, as none of the replicas change the data, only a designated “master” performs updates. These replication clusters are well suited for read-only tasks, however, as soon as an application generates a heavy, interactive write workload, simple replication techniques are no longer sufficient. Consider a database used by a web store to record what each customer ordered. While e.g. the catalogues and pictures on the web sites themselves may be easily replicated, actual customer orders cannot since those are typically frequently updated. To use a cluster for the shopping data, either the replication techniques need to ensure that all replicas in the cluster have the same order data at the same time, or fragmentation has to be used to split the data up over the cluster, where each system in the cluster is responsible for its own part of the data. These two techniques, in particular for storing and retrieving of data, are deployed in database technology to benefit from a cluster setup.

Distributed Database Management Systems (DBMSs) cover the area of de-
ploying databases on a cluster. They store structured data. Replication and fragmentation are well researched topics and various forms of either copying or splitting the database contents exist. However, the nature of a DBMS is to be well in control. Certain properties that equip a distributed DBMS with robustness, reliability and correctness (referred to as ACID properties) prevent the individual systems from operating on their own. Instead, they have to serve the system that coordinates the process of retaining the ACID properties in the cluster. This PhD thesis explores the setting in which a distributed DBMS seeks to loosen the ACID properties to the degree where in contrast to traditional clusters, the distributed DBMS consists of sovereign and autonomous database systems. The autonomy of the systems allows them to solve problems on their own, using local information to influence the solution.

In a search for structured access to the data, database technology has been developed over the years. Not only the data volumes grow, but also the use of databases for storing that data. Stimulated by freely available database servers of all kinds, applications are (re-)designed to use databases as their storage back-ends. For some applications not only the data, but also the state of the application is stored in a database to benefit from their reliability and persistence, like in Hibernate [39]. Databases are more and more often used, with an increasing amount of data involved.

In Werner Vogel's keynote at VLDB’07 [72], database scalability is identified as an indispensable ingredient required by applications of world wide sizes. The matter in which a database can be extended to scale to a new level, defines its usefulness to the application in terms of adapting to its data needs. This process, where a database grows along with the data requirements of the application, is coined as incremental scalability. The database evolves over time, requiring a flexible administration of data whereabouts to support reconfigurations. In the light of this property, autonomy is an enabler for flexible reconfigurations. No longer a change in the system needs to be agreed upon by the distributed DBMS, but just the local system that is involved takes care of the operation, whenever appropriate.

Such freedom resulting from autonomy needs a strategy to become effective in a cluster. The generic model proposed in this thesis is aimed at administering data spread over multiple autonomous systems. The administration includes generic functions that define how the data is spread. These functions are independent, and hence their implementation is decided upon by the system involved. With specific implementations of those functions, existing distributed database systems can be emulated, thereby it demonstrates the model's generic
nature. Being a model to define how original, central, data is spread around, we acknowledge that it need not to be specific to relational databases only. However, we approach the model from a relational database point of view.

In this thesis we study the distribution of databases with evolutionary behaviour as focus. To achieve the flexibility of this evolution process, we research a database system comprising autonomous systems. With this study we make steps into directions for further research and development of decentralised, evolving and autonomous database systems.

The ability to adapt to the change in data requirements is considered important and an enabling technology that is necessary to deal with the future. The abilities to flexibly and sufficiently adapt to data requirements are missing in many database products today.

1.2 Research Questions

The focus of this thesis is an exploration towards autonomy, decentralisation and evolution of database systems. The general research question addressed, stresses the continuous aspect of evolution in a dynamic database:

How to support a continuously evolving database management system consisting of autonomous sites and a decentralised catalog?

The research question defines decentralisation and autonomy as important directions of the exploration. To get a better grip on how to answer this question, we refine the general research question into four more specific questions. The first question addresses data distribution with the autonomy and derived independence of sites that are part of a larger database management system:

1. In what way can we distribute data in a dynamically evolving system using site local decisions and avoid global site control?

The aforementioned site autonomy has effects on how those sites can be used by clients. In particular the question on the level of client participation is formulated by the next question:

2. What is the role of application clients in an autonomous, distributed database management system?

Tightly related to the autonomy dimension is the evolution component. The next question expresses the focus on the continuity of that evolution and how to make it an integral part:
3. How can incremental scalability become a natural component in an evolving system?

Finally, in a search for how our ideas affect implementations of distributed database systems, we investigate the effects of our exploration on existing machinery in the last question:

4. To what extent are existing common techniques to manage a catalog sufficient to support autonomy, decentralisation and evolution?

1.3 Approach and Outline

To answer the research questions, we start with the introduction of a model in Chapter 3 that forms the foundation for all the subsequent chapters of this thesis. The formal model describes a way to administer the evolution process of a database. The model itself has emerged in an evolutionary process of many iterations, with each iteration being a refinement based on new insights gained during the process. Eventually, a generic, minimal form of the model was found, which is described in Chapter 3. The assumption of the use of autonomous local decisions made by the participating systems is embedded in the model. With this assumption, the model — as heart of this very thesis — answers the first research question by describing how data distribution can be done using autonomous participants. The model also answers to a large extent the third research question, because its design includes continuous refinement, empowered by the autonomy of sites to locally reconfigure and refine.

The model achieves incremental scalability by operations applied to the data. These operations can contain an arbitrary function working on the data. In Chapter 4 we examine these functions, their properties and their effect when used in the model. The operations applied within the model allow for continuity during the evolution of the database, and hence form the rest of the answer to the third research question.

The autonomous nature of the model directly affects its application clients. Chapter 5 describes an autonomous query strategy that matches the model, thereby answering the second research question. The autonomy used by the strategy makes it unconventional and yet largely unexplored territory. In Chapter 6 we explore the effects of the model combined with different variants of the application client strategy to gain some insight on the viability of both when put in practice.
To answer the fourth question, Chapter 7 presents a simulation study, in the de-facto query language SQL. This study includes a little experimentation on the overhead of this implementation on some SQL database systems. It is shown that standard DBMSs are far from being able to support the proposed model. Therefore in Chapter 8 we describe the last contribution of this thesis, an architecture that supports the model and strategy that are part of our exploration.

Before we start with previously mentioned chapters, in the next chapter we first give an overview of work done in the field. It positions this thesis in the broad area of database research. The overview is followed by the main chapters of this thesis, which in turn are followed by a conclusion and outlook on future work. An appendix on developed technology aimed at a realisation of the ideas presented in this thesis is included for background information purposes.