MobiSnap

Managing Database Snapshots on a Mobile Environment

Technical Annex

MobiSnapTeam (Algoritmi - CITI)

Summary

Recent technological breakthroughs such as increased battery capacity, low-consumption, small light components and widespread use of cellular communication devices, have resulted in a significant development of portable computer models. As a consequence, an heterogeneous array of mobile computers with varying dimensions, weight and computing power is currently available, leading to an ever increasing number of users.

Depending on their features, portable computers are mostly being used either as transportable workstations or as personal digital assistants (e.g. hosting calendars and to-do lists) occasionally providing some access to Internet resources (e.g. email). This happens because developing applications which take advantage of mobility and wireless communication requires balancing the availability and consistency of data.

This projects aims to provide methodologies and tools to develop database applications on mobile computers. This should be achieved by providing tools to extend current applications which depend on central databases.

In order to allow continued operation of mobile computers, which often get disconnected from the network, the MobiSnap framework will provide mechanisms to semi-automatically maintain weakly consistent partial copies of the central database. This requires policies to control which data are replicated and how concurrent updates of data are reintegrated into the central database. The programming environment provided to the client application will be a subset of SQL, suitably extended to explicitly deal with weak coherence. For instance, it should be possible to specify the expected behavior of delayed transactions.

On the central server, the MobiSnap framework will use a proxy for each mobile client, which will be responsible for mediating the access to the central database. The proxy will guarantee that the divergence of mobile copies from the central database is bounded within reasonable limits. In this context, reasonable limits imply the definition of divergence metrics for each application, which should be tightly related to the risk involved. To bound divergence, the server will also be able to use several media to notify clients, ranging from the short message service of cellular networks (SMS) to traditional local area networks.

In short, the goal is to build a framework to extend current database applications, which already exist in most organizations, to encompass a new class of clients residing in mobile computers. The key issue is the homogeneous support through the relational model and the SQL interface, which is the standard in most database systems. A systematic approach to the definition of replication, update and coherence policies will enable the development of tools that safely introduce a radical change into mobile database access. The freedom to select which policy is effectively used for each application will allow the framework to be useful in the context of various economic activities that can take advantage of the mobility of current computing devices.

Sumário

Nos últimos anos assistiu-se a uma rápida evolução dos computadores portáteis, resultante da conjungação de vários desenvolvimentos tecnológicos. Entre outros, pode citar-se o aumento de capacidade das baterias, o desenvolvimento de componentes especializados com reduzidas dimensões, peso e necessidades energéticas e a generalização das comunicações celulares. Esta evolução traduziu-se no aparecimento e subsequente popularidade de diversos de computadores portáteis com diferentes dimensões e desempenhos.

Dependendo das suas características, os computadores portáteis têm sido utilizados ou como versões transportáveis de estações de trabalho, ou no outro extremo, como suporte de pequenos utilitários pessoais (por ex., agenda pessoal) e programas de interacção simples com a Internet (por ex., correio electrónico). Isto acontece porque a criação de aplicações mais complexas que explorem a mobilidade e as especificidades dos sistemas de comunicação móveis, implica conciliar a disponibilidade com a coerência dos dados manipulados.

Neste contexto, com este projecto pretende-se sistematizar o desenvolvimento de aplicações de consulta e actualização de bases de dados relacionais a partir de computadores móveis. Para o efeito, pretende-se fornecer uma plataforma que permita o rápido desenvolvimento de extensões para computadores móveis de aplicações já existentes que interagem com bases de dados centrais.

De forma a permitir uma utilização contínua dos computadores móveis sujeitos à variabilidade da comunicação, a infra-estrutura MOBISNAP facultará mecanismos para manutenção nos computadores móveis, de forma semi-transparente, de cópias parciais fracamente coerentes dos dados existentes nos sistemas centrais. Deverá permitir-se a definição de políticas de selecção dos dados a replicar, assim como de actualização concorrente dos dados, replicados localmente ou não, quando propagados para os sistema central. O ambiente de programação fornecido pelo componente cliente conterá uma parte das funcionalidades do SQL, as quais serão adicionadas operações que permitam lidar explicitamente com a fraca coerência dos dados, por exemplo, construções que permitam a definição exacta da operacionalidade esperada aquando da execução duma transacção.

No servidor central, a plataforma MOBISNAP será constituída por um representante de cada cliente móvel, responsável pela interacção entre estes e a base de dados relacional central. Este representante deve garantir que a divergência entre as cópias presentes nos computadores portáteis e os dados presentes no servidor se mantém dentro de limites considerados razoáveis. A definição de razoabilidade está associada à definição de métricas de divergência específicas para cada tipo de aplicação, as quais devem estar intimamente ligadas ao risco associado. Deverão ser explorados diversos mecanimos de comunicação, desde o serviço de mensagens curtas de redes celulares (SMS) a redes locais, para notificação dos clientes.

Em suma, procura-se uma plataforma que extenda os sistemas de gestão de dados pré-existentes em diversas organizações, por forma a enquadrar uma nova classe de clientes, residentes em computadores móveis. Para o efeito, suportam-se bases de dados relacionais e adopta-se o SQL como o interface destas com o representante do cliente, o que permite um tratamento homogéneo e padronizado de uma vasta gama de sistemas. A sistematização das políticas de gestão de actualizações e de controlo de divergência deverá fornecer as ferramentas necessárias para a introdução segura de evoluções radicais na organização das interações com as bases de dados centrais. O grau de liberdade associado à definição das políticas efectivas deverá permitir a adaptação de configurações específicas às várias actividades económicas e profissionais que pretendam tirar partido da mobilidade facultada pela emergência tecnológica que é actualmente patente nesta área.

1 Introduction

In recent years, advances in hardware and network technology enabled a rapid evolution of portable computers. Some of these advances can be seen in wireless and cellular communication, battery technology, and in the constant reduction of the dimensions, weight and energy requirements of many components [13]. This has led to a new generation of portable machines with different dimensions and capacity, ranging from small handheld devices with scarce storage and processing capacity to powerful laptop computers. Regardless of their size, most of these mobile computers are equipped with wireless connections, allowing access to static computers and even to other mobile devices. The widespread use of such computers creates a heterogeneous environment where mobile computing devices are ubiquitous.

Mobile computing paves the way to new classes of applications, exploiting not only the inherent portability and mobility of computer devices, but also the wide availability of communications. Information systems may be accessed from any place in order to query and modify data, opening new operational possibilities for a wide range of economic activities [14]. However, the distinct characteristics of mobile computing [22] demand novel solutions to old problems such as communication breakdowns. In other words mobile computing can not be regarded as a "scaled-down" version of the well-studied field of distributed system.

Currently, most applications available for a broad range of portable computers consist of small utilities such as calendars and fax programs, and Internet-based programs such as e-mail programs and WWW browsers. The development of new applications for specific problems is largely dependent on the access to information servers. Unfortunately, no infrastructure is widely available to help programmers to easily deal with mobility, including reduced, varying and even unavailable connectivity, and battery power limitations. As a consequence, programmers have to develop their own solutions from scratch over and over again.

Relational database systems are widely used to store and process company-wide data. As a result, MobiSnap aims at developing a middleware infrastructure that allows access to relational database systems from mobile computers with a clear semantics in presence of all operational scenarios (from high to unavailable connectivity). This platform should isolate programmers from the problems related to mobility and disconnection, allowing them to easily develop new applications to mobile environments, focusing only in application-specific problems. MobiSnap will be based on SQL, thereby also providing a close integration to legacy information system.

To support operation in the presence of disconnection, MobiSnap will combine two major techniques: caching and operation logging. The first, in combination with a weak consistent model of data access, enables users to query data in any situation (even when disconnected). The second allows users to modify data by logging their transactions and propagating them to servers later. We intend to investigate caching techniques based in user profiles and develop adequate solutions to our problem. Transaction submission in presence of disconnection is a subject that also requires further investigation.

The weak consistency of data may pose problems to some economic activities (but, unfortunately it is required to enable disconnected operation). Therefore, we intend to provide a mechanism to measure divergence between clients cache snapshots and data present in the central server. This mechanism should be highly related with risk associated with using each data item. The divergence metrics will be used, not only to users information, but also to improve data consistency. Different forms of synchronization and notification may be used to reduce inconsistency, depending on connection quality of service.

In recent years, our research groups have been investigating problems related to mobile computing environments and to large-scale collaborative applications. Experience has been gained addressing problems posed by wireless communications, concurrent updates and caching. The combination of both of our previous experiences will help in the development of the current project. We expect to validate our ideas with a real applicational scenario that has already served as the basis to produce our design. We believe the same approach to be useful in many other situations.

The rest of the proposal is organized as follows: in section 2 we introduce some related work; in section 3 and 4 we present the system overview and programming interface; section 5 and 6 further detail server and client components of the architecture; in section 7 we introduce an applicational scenario for our platform; in section 8 we make a brief summary of project's main goals and challenges; in section 9 we conclude by presenting the projects workplan.

2 Related Work

The MobiSnap project addresses the use of weak consistent data replication (caching) for database manipulation under a mobile environment. As such, two areas of related work converge in this project: weak consistency data management protocols that deal with mobility, such as those introduced by Coda[23], Ficus[11], Notes [17], Bayou[8] and Rover[16]; and asynchronous update protocols for distributed database operation with relaxed concurrency control[19], such as *quasi copies* [3, 4] and *epsilon serializability*[21]. This two corner areas are complemented by specific research on the design of mobile computing applications, that delimit important aspects such as the identification of the asymmetric relation between mobile computers and hosting computers [13], the existence of lower dependability on mobile storage [22], the need to introduce location awareness [25, 1] as well as the need to efficiently manage heterogeneous connectivity[2].

2.1 Weak Consistent Mobile Systems

Several systems have been developed to manage data in large-scale environments including mobile computers. Notably, some mobile database systems based on transactions [10] use a well-understood model of concurrency control. However, traditional transactions based on simple read and write operators define a too restrictive model of concurrency control for mobile applications. The lack of operational semantics leads to frequent and undesirable abortions. Novel system designs are required to permit looser concurrency control in order to cope with the divergence that results from disconnected updates. Some of the more relevant system designs are described below.

Lotus Notes [17] is a replicated document database. Documents have a record-like structure composed by typed fields defined in forms. Notes architecture is composed by a group of servers that replicate databases (sets of documents) using epidemic techniques and by clients that cache documents. Notes propagates fields values, handling concurrent updates by creation of multiple versions of data that must be manually merged.

Coda[18] is a replicated file system with support for disconnected clients. It also supports low bandwidth networks and intermittent communication. While disconnected, clients log all updates to the file system, which are replayed on reconnection. System executes automatic update conflict resolution for directories. Application-specific programs can be provided for automatic resolution of file updates conflicts. However, lacking of update semantics - files are modified as untyped byte streams - makes updates merging rather difficult and sometimes impossible.

Ficus [12] is a replicated file system that uses similar conflict resolution policies, but uses an epidemic scheme to propagate updates among servers. Although the above-mentioned systems provide an update merging mechanism, it presents severe shortcomings due to the state propagation strategy. Next we present two systems that use an update propagation strategy: Rover and Bayou.

Rover [15] combines relocatable dynamic objects (RDO) and queued remote procedure calls (QRPC) to provide information access for mobile clients. Each RDO has a home server and may be imported by clients. While imported, updates are logged and performed locally. When the RDO is exported, logged updates are applied to the replica at the home server. Resolution of detected conflicts is achieved at servers by calling type-specific methods. RDOs are also used to export computations to servers. QRPC are used to execute all communications between clients and servers, allowing non-blocking RPCs even while disconnected. Although allowing great flexibility, this system imposes operations to code specific preconditions. We believe that generic linguistic constructions should be provided to ease system development.

Bayou [24] is a replicated database system to support data-sharing among mobile users. The Bayou architecture is composed by a group of servers that replicate databases using epidemic techniques and by clients access these replicas. Bayou updates (writes) include information to allow generic automatic conflict detection and resolution through dependency checks and merge procedures. Bayou data presents two values: tentative and committed. A primary replica scheme is used to fasten update commitment. One of Bayou main strengths is its update definition. However it is also one of its weaknesses because it allows only simple updates (not full transactions). This is due to the techniques used to deal with server replication.

Experience with these systems gave us a better understanding of requirements and possible solutions for a mobile data management system. We expect to develop a solution based on the weak consistency model of data access and update propagation strategy that allows applications programmers to easily express the operational semantics of each action. Being these systems either based on file system extensions, or tied to new development frameworks, we believe that it is now crucial to offer a system that allows orthogonal extensions of installed applications.

2.2 Relaxed Consistency Database Systems

Previous work on divergence bounding in database systems is taken as the basis for our proposed work on the definition of divergence metrics for the server and client view of the replication system.

A recent characterization [19] distinguishes two classes of consistency requirements, *data oriented* and *operational approaches*. Data oriented methods [3] classify divergence as bounded to the replica contents and are able to place restrictions on the allowed divergence of the state. As a practical example, the Portuguese ATM (Automated Teller Machine) system allows disconnected withdraws up to a given bound, that is expressed in monetary units.

In contrast, operational approaches bound the permitted amount of inconsistency to units of execution like individual operations or transactions. This can be expressed by a extended notion of transaction, as in *epsilon transactions* [21], that allows the application programmer to specify the allowed inconsistency for each operation.

Existent work on the characterization of semantic properties of operations [9, 7] also provides valuable leads to the provision of an interface that allows the programmer to rule different divergence metrics for given classes of operations. In MobiSnap we intend to develop divergence metrics that may be related with data, and to its inherent value to mobile users activity. This notion will be highly related with the risk associated with outdated data usage, but measured according to activity risk and not in abortion rates. Moreover, rather than using these metrics to place restrictions on users operations we expect to use them as users information and to improve data consistency.

3 System Overview

MobiSnap is a middleware infrastructure to allow access to relational database servers from mobile computers with variable connectivity (ranging from high-speed connection to complete disconnection). Our targeted environment consists in a group of mobile users that are expected to have access to information stored in the database server in any situation. For instance, a group of sellers that are visiting their prospect clients should have access to information about the products (description, prices, stock) they are selling to inform their clients. This information is usually stored in the supplier central server. They should also be able to enter their customers orders, which are (later) transmitted to the central server.

Mobile users require access to subsets of all information that exists in the central server. To this end, each mobile client caches the relevant information based on a caching algorithm. In our environment we expect that the pre-fetching policy will be based mainly on the combination of users profiles and data clustering [5] - a computers seller will cache only all information related to computers.

Caching and update propagation algorithms will be based on the notion of database snapshot. Each database snapshot is composed by several table snapshots, which are a subset of data stored in a database table. Table snapshots include a subset of rows and columns of a given table, and may be defined as the subset of table rows that match a certain condition. For instance, a given seller would only need a snapshot related with some (not all) information about the products and customers she/he is going to visit. The definition of snapshots boundaries is highly related with the pre-fetching and caching policies.

As strict consistency of information is not required in most situations, mobile users may continue their work in disconnected mode using cached data (snapshots). For instance, sellers do not need to have up-to-date information about products to continue their work. However, as in some circumstances, decisions will be taken based on the weakly consistent data, we intend to provide some mechanisms to evaluate transaction validity (we will detail this in the programming interface section).

Transactions issued by mobile users will be logged and later propagated and reintegrated in the central database. These transactions may be applied to cached data or not, depending on users preferences. This mechanism also allows submission of transactions executing over data not present in local cache (as applications know tables structure transactions may be defined). Thus, complete operation is allowed in presence of any connectivity conditions.

Although weak consistency of snapshots is acceptable for most operations, strict consistency would be preferable. Therefore, the system should be responsible to guarantee that data divergence is kept below a given threshold. To this end, near to the database servers, it exists a proxy of each mobile client that is responsible to track all changes made to the snapshot cached by each mobile computer. This proxy is responsible to evaluate data divergence in order to minimize risk associate with users activity.

Whenever data divergence exceeds a user-defined threshold, users are notified to perform data synchronization or at least to be warned that data divergence has reached some threshold. The combination of synchronous communication sessions with asynchronous notification mechanisms, as those presented by GSM, will enable a wide support for different connectivity degrees. For example, it is usual that in border areas, asynchronous SMS notifications are available although the quality of service is insufficient to establish synchronous connections.

The importance of data consistency for mobile users activity is diverse for different data items, and for the same data item, it may be diverse for different ranges of values. For instance, sellers should have precise information about products price. However, regarding available stock they only require exact values when supplies are running short. In general, the importance of each data item consistency is highly related with the risk associated with missed opportunities and invalid transactions due to outdated data.

In MOBISNAP, we intend to develop an appropriate mechanism to express data items importance. The associated metrics will be used to determine when users should be notified about data updates and how urgent each notification is. Moreover, these metrics will also be used to determine when clients should propagate mobile users updates to servers, in order to increase the probability of transactions success.

One of the aims of the project will be to study divergence metrics. The divergence metrics may be based in database values and/or in the number of writes performed. Support from the database system will be required in the latter case, while the former could be implemented without any support (although it might be more efficient with database cooperation). Ideally, we envision a snapshot divergence evaluation trigger mechanism that could be integrated in the database system without any performance impact. Several options exist to implement it, and further research is required.

In MOBISNAP, all components will be implemented without any interference with the central database server. Thus, legacy programs will continue to run without any required modification.

4 Programming Interface

Usually, applications interact with relational database servers through SQL statements. However, providing full SQL compliance in small and powerless computers such as small personal digital assistants is not an easy task. To overcome these difficulties we are planing to provide a programming interface that exports snapshots (cached data) as simple bidimensional arrays with some simple queries (e.g., searching rows with a given column value). In this case, updates could be performed either through (extended) standard SQL transactions or using the reduced interface.

Since snapshots (cached "tables") may correspond to the result of join operations in server data, we believe that the above restriction (not allowing SQL complex queries) will represent no problem for applications development. Nevertheless, in powerful laptops we intend to offer a full SQL compliant interface. We also intend to investigate the possibility to provide SQL for less powerfull machines. Both solutions represent alternative interfaces for application development. Interaction between clients and (proxy) servers will be defined after further investigation. Access to multiple database servers in a federated-like way could be an interesting feature for mobile users (e.g., sellers may have information from multiple suppliers in the same "tables" - snapshots). We intend to further study the problems posed by that situation.

Due to weakly consistent data, some transactions submitted by mobile users may be invalid. In our sellers example, if some buying decision is issued based on a wrong price or stock we expect to be able to invalidate that decision. However, transaction validation using concurrency control schemes based on detection of read/write and write/write conflicts is often inadequate. In the previous example, a buying decision may be valid with different stock values (since enough available stock exists). To express such situations we intend to provide a pre-condition definition mechanism to be associated with transaction definition.

Usually, in database systems invalid transactions lead to abortion. However, as in mobile systems, the result of a submitted transaction will be evaluated later and without direct contact with the submitter, it may be interesting to provide some procedure to be executed in case of failure (e.g., this procedure may be used to notify the mobile user or to propose a different transactions). Due to the same reason, it may also be interesting to provide a procedure to be executed when transactions succeed.

Some previous research systems [8] have already addressed some of these problems. However, their solutions have limitations due to requirements we do not need to address (namely, server replication). In this project we intend to devise mechanisms that allow mobile users to precisely express the expected operational behavior. Another issue that we intend to study is the definition of updates (whether additive operators should be provided instead of usual write operations). In an ongoing research effort, the members of one of the teams involved in this project are investigating the same problems in a different environment – a large-scale collaborative setting [20]. We expect that experience gained in both settings could be used to produce better specific solutions for each specific environment.

5 Server Component

Mobile systems are naturally asymmetric when a Support Station to Mobile Host relation is established. In MobiSnap the server component takes the role of a support station and as such should strive to releave the client side from all tasks that can be delegated. This component will take the dual role of acting as the interface to the legacy database, as well as supporting the mobile client.

Under its interface role, the server will be ready to provide the translation between the relational database, accessed via SQL, and the table snapshots abstractions that are used on the mobile client. Once defined, these snapshots are derived from the database by issuing a set of appropriate queries that are associated to each snapshot object. A snapshot is defined as the set of objects that express a view of the database at a particular time.

Client update operations are done directly on the database. Translation from the log of client operation into the appropriate standard SQL statements will be done if required. Translated clients operations are applied by the server as soon as they are uploaded. Once these updates are executed, the client (if still accessible) can be updated with a recently computed snapshot. It is also on the server component that re-integration policies should be tailored to rule update operations when semantic knowledge is available.

For each mobile client, the server will additionally keep a copy of the last snapshot that was sent to the client. By periodically calculating the current snapshot and comparing it to the one present in the mobile computer, the server can determine the amount of divergence that is introduced by ongoing updates to the database. This is particularly important, as divergence can be calculated without intercepting ongoing updates, thus allowing a seaming-less integration on existing systems.

The specification of adequate divergence metrics and triggers should result in a prolific research topic. Once established, these triggers can force asynchronous notifications to the client, and instruct him to establish communication and refresh its snapshot. In fact, communication should be established from the client side, since it is the client who is subject to changes on the available connectivity.

6 Client Component

MobiSnap clients will cache data to be used by mobile users while disconnected. Caching techniques have been extensively studied and used in distributed systems to overcome performance weaknesses. However, using caching to solve disconnection problems presents different requirements (a cache miss poses an unsolvable problem while disconnected). Thus, different approaches are required and have been developed in previous research projects [5, 23, 6]. These approaches are based mainly on data clustering (set of related data that are jointly cached), users profiles (users have explicitly or implicitly - through inherent properties - associated data) and statistical analysis. We intend to combine these techniques to develop appropriate caching mechanisms in face of our targeted environment special requirements, and to provide support to define data clusters and users profiles.

MobiSnap clients will log transactions performed while disconnected in persistent storage. These transactions will be later propagated to the servers, as possible. Logged transaction may be immediately applied to cached data, depending on users preferences. The use of an operational log has already been adopted in previous systems [23] (sometimes disguised as asynchronous remote procedure calls [16]) and represents a widely used technique. The major challenge associated with it is the development of an adequate mechanisms to express operational semantics (see programming interface section for more details).

Clients communicate with servers to synchronize their states, updating clients caches and propagating transactions performed while disconnected. Log re-integration should be done as soon as possible to decrease the probability that data divergence leads to transaction invalidation. However, to minimize communicational costs it should be delayed as much as possible. These two conflicting goals will be addressed in MobiSnap using divergence metrics (similar to those used to notify clients of cache changes). Different communicational modes will also be used to propagate mobile transactions to servers (SMS asynchronous notifications will be combined with synchronous connections). Clients must also manage communications with proxy server when notifications about snapshot updates are received.

7 Applicational Scenarios

The MobiSnap project trades strict data consistency for mobility and availability. As a result, all database applications that can tolerate optimistic replica control will benefit from MobiSnap. It is envisaged that the client side of current SQL-based applications will migrate into increasingly portable devices such as palmtops and PDAs, deferring data synchronization with the server to periods where communication is available or requested by the server in order to update the remote caches.

As mentioned above a typical scenario is provided by a group of people whose job is to contact previous and prospective customers for the purpose of selling some merchandise. Traditionally this involves collecting the orders and updating the central database, either in a batch at the end of the day or using a mobile phone. MobiSnap will allow (cheaper and widely available) disconnected operation using local snapshots, yet providing a means of contacting the central server to ensure data consistency, for example to take advantage of lower prices.

Other potential applications would be shop-floor control where information must be passed between the server and hand-held devices, or the stock market where investors may require timely information about stock-exchange values. In general, applications following the publisher-subscriber model will clearly benefit from the mobility support provided by MobiSnap.

8 Synthesis

Mobile computing is still a recent area of research. Nevertheless, over the past years, it has become clear that the techniques used in traditional distributed systems do not scale well to mobile and disconnection-aware applications.

MobiSnap has a two-folded goal. First, its technological component should fill the gap in current mobile frameworks which instead of proposing new development scenarios, aim at the extension of existent applications for the inclusion of mobile capabilities. MobiSnap addresses this objective by basing its data interface on a widespread database industry standard.

Secondly, scientific contributions are sought on a delimited area that encompasses the definition of suitable divergence metrics and associated actions, both on server and client sides. Previous research in those topics was done is the database area and does not consider the influences of different quality of service of connections, as well as the very different ratio of disconnections that characterizes mobility. The definition of adequate mechanisms to express snapshots boundaries and the expected operational semantics in presence of mobility and disconnection will also be investigated.

9 Workplan

9.1 Tasks

The development of MobiSnap project comprises several tasks, described below. For each one, we present its objective, needed human resources, people and teams in charge and time frame to be executed.

9.1.1 Task 1 - Design

Goal Clearly define MobiSnap architecture, including client and server components and interaction protocols.

Timing 1 to 6 th month

Human-resources 4.4 man*month - Algoritmi 50% CITI 50%

Coordinator FM

9.1.2 Task 2 - Programming issues

Goal Define snapshot representation and interaction interface. Define mobile transactions representation.

Timing 1 to 12 th month

Human-resources 7 men*month - Algoritmi 30% CITI 70%

Coordinator NMP

9.1.3 Task 3 - Metrics definition

Goal Define a framework to express data importance and to estimate data divergence.

Timing 3 to 15 th month

Human-resources 7 men * month - Algoritmi 70% CITI 30%

Coordinator JALM

9.1.4 Task 4 - Client component

Goal Implement the client component as defined by tasks 1 to 3.

Timing 6 to 18 th month

Human-resources 15 men * month - CITI 100%

Coordinator JALM

9.1.5 Task 5 - Server component

Goal Implement the server component as defined by tasks 1 to 3.

Timing 6 to 18 th month

Human-resources 15 men*month - Algoritmi 100%

Coordinator FM

9.1.6 Task 6 - Overall integration

Goal Integrate client and server components in an integrated and functional platform.

Timing 15 to 24 th month

Human-resources 4 men*month - Algoritmi 50% CITI 50%

Coordinator JALM

9.1.7 Task 7 - Application development

Goal Implement at least one application to test MobiSnap design and prototype implementation.

Timing 12 to 24 th month

Human-resources 6 men*month - Algoritmi 80% CITI 20%

Coordinator CBM

9.1.8 Task 8 - Training, documentation and coordination

Objective Project management. Produce documentation, publication material and two MSc thesis.

Timing 1 to 24 th month

Human-resources 10 men*month - Algoritmi 50% CITI 50%

Coordinator FM

9.2 Deliverables and Dissemination

Early prototypes and stable versions of the proposed system will be disseminated in the WWW sites of the team. This method allows a fast and wide delivery of the product to other research teams and industry users, and has been successfully used in the past for the same purpose.

Each task will produce work reports that evolve together with the task and end up in a final version when the task is concluded. These reports will enable early dissemination of results among peers, and will base the production of the research papers and publications that are scheduled for the dissemination of the project results.

As the project also plans to integrate Master and PhD students, as well as final year students, it is expected to give an extended contribution to the formation of young scientists.

References

[1] Arup Acharya, T. Imielinski, and B. R. Badrinath. Dataman project: Towards a mosaic-like location dependant information service for mobile clients. Technical Report TR-320, Rutgers University, 1994. http://paul.rutgers.edu/~acharya/publications.html.

[2] Tahir Ahamad, Mike Clary, Owen Densmore, Steve Gadol, Arthur M.Keller, and Robert Pang. The diana approach to mobile computing. Technical report, Stanford University, Sun Microsystems, 1993. http://www.cs.rutgers.edu/~badri/journal/contents11.html.

[3] Rafael Alonso, Daniel Barbara, and Hector Garcia-Molina. Data caching issues in an information retrielval system. *ACM TOCS*, pages 359–384, September 1990.

[4] Rafael Alonso and Henry F. Korth. Database system issues in nomadic computing. In *International Conference on Management of Data, ACM SIGMOD RECORD*, volume 22, pages 388–392, May 1993.

[5] B. Badrinath and S. Phatak. Database server organization for handling mobile clients. Technical Report dcs-tr-324, Computer Science Department, Rutgers University.

[6] Daniel Barbara-Milla and Tomaz Imielinski. Sleepers and workaholics: Caching strategies in mobile environment. *ACM SIGMOD RECORD*, 23(2), May 1994. http://www.cs.rutgers.edu/~badri/journal/contents11.html.

[7] S. Ceri, M. Houtsma, A. Keler, and P. Samarati. Independent updates and incremental agreement in replicated databases. Technical Report CS-TR-92-1446, Stanford University, 1992.

[8] Alan Demers, Karin Petersen, Mike Spreitzer, Douglas Terry, Marvin Theimer, and Brent Welch. The bayou architecture: Support for data sharing among mobile users. In *IEEE Workshop on Mobile Systems and Aplications*, Computer Science Laboratory, Xerox Palo Alto Research Center, December 1994.

[9] A. Downing, I. Greenberg, and J. Peha. OSCAR: A system for weak-consistency replication. In *IEEE Workshop on the Management of Replicated Data*, 1990.

[10] Robert Gruber, Frans Kaashoek, Barbara Liskov, and Liuba Shrira. Disconnected operation in the thor object-oriented database system. In *IEEE Workshop on Mobile Systems and Aplications*, Laboratory for Computer Science, Massachusetts Institute of Technology, December 1994.

[11] Richard G. Guy, John S. Heidemann, Wai Mak, Thomas W. Page, Gerald J. Popek, and Dieter Rothmeier. Implementation of the ficus replicated file system. In *USENIX Conference Proceedings*, pages 63–71. USENIX, June 1990.

[12] J. S. Heidemann, T. W. Page, R. G. Guy, and G. J. Popek. Primarily disconnected operation: Experience with ficus. In *2nd Workshop on the Management of Replicated Data*, pages 9–12, 1992.

[13] Thomas Imielinski and H. Korth. *Introduction to Mobile Computing.* Kluwer Academic Publisher, 1996.

[14] Tomasz Imielinski and B. R. Badrinath. Mobile wireless computing, solutions and challenges in data management. February 1993.

[15] Anthony D. Joseph, Alan F. deLespinasse, Joshua A. Tauber, David K. Gifford, and M. Frans Kaashoek. Rover: A toolkit for mobile information access. In *Fifteenth Symposium on Operating Systems Principles*, December 1995. http://www.pdos.lcs.mit.edu/rover/.

[16] Anthony D. Joseph, Joshua A. Tauber, and M. Frans Kaashoek. Building reliable mobile-aware applications using the rover toolkit. *ACM Wireless Networks*, 1996. http://www.pdos.lcs.mit.edu/rover/.

[17] L. Kawell, S. Beckhardt, T. Halvorsen, R. Ozzie, and I. Greif. Replicated document management in a group communication system. In *Proceedings of the 2nd ACM Conference on CSCW*. ACM, 1988.

[18] J. J. Kistler and M. Satyanarayanan. Disconnected operation in the Coda file system. In *Thirteenth ACM Symposium on Operating Systems Principles*, volume 25, pages 213–225, Asilomar Conference Center, Pacific Grove, US, 1991. http://www.cs.cmu.edu/afs/cs/project/coda/Web/docs-coda.html.

[19] Richard Lenz. Weak consistent data replication with "virtual primary copies". In Max Muhlhauser, editor, *Special Issues in Object-Oriented Programming*, pages 400–408. Dpunkt Verlag, 1996.

[20] N. Preguiça, J. Legatheaux Martins, H. Domingos, and J. Simão. System support for large scale collaborative applications. Technical Report TR-FCT-DI-98-001, Departamento de Informática, Universidade Nova de Lisboa, 1998.

[21] Calton Pu and A. Leff. Epsilon-serializability. Technical report, Department of Computer Science, Columbia University, New York, 1991.

[22] M. Satyanarayanan. Fundamental challenges in mobile computing. In *PODC'96*, pages 1–7. ACM, 1996.

[23] M. Satyanarayanan, J. J. Kistler, L. B. Mummert, M. R. Ebling, P. Kumar, and Q. Lu. Experiences with disconnected operation in a mobile environment. In *USENIX Symposium on Mobile and Location Independant Computing*, pages 11–28, Cambridge, Massachusetts, US, August 1993. http://http://www.cs.cmu.edu/afs/cs/project/coda/Web/docs-coda.html.

[24] Douglas Terry, Marvin Theimer, Karin Petersen, Alan Demers, Mike Spreitzer, and Carl H. Hauser. Managing update conflict in bayou, a weakly connected replicated storage system. In *15th ACM Symposium on Operating Systems Principles*, Copper Mountain Resort, Colorado, US, December 1995. http://www.parc.xerox.com/csl/projects/bayou/.

[25] Geoffrey M. Voelker and Brian N. Bershad. Mobisaic, an information system for a mobile wireless computing environment & engineering. In *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, US, December 1994. http://snapple.cs.washington.edu/mobile/mcsa94.html.