무선환경에서의 이동객체 관리를 위한 통제기법: 공항 안전통제 업무를 중심으로

A Transaction Scheme for Managing Mobile Units in Wireless Environments: A Case of Airport Safety Control

김 용 재 (Yong Jae Kim)	건국대학교 경영정보학과 교수
안 현 섭 (Hyunsup Ahn)	KAIST 경영공학과
이 석 준 (Seogjun Lee)	건국대학교 경영정보학과 교수, 교신저자
현 두 운 (Doowoon Hyun)	아시아나 IDT 영업지원팀 차장
최 용 구 (Yonggoo Choi)	동서울 대학교 부교수
이 석 원 (Sangwon Lee)	KAIST 경영공학과

요 약

이 연구는 한국에 위치한 어느 공항의 프로젝트를 수행하는 가운데 도출된 결과다. IT 솔루션들을 놓고 검토한 결과, 레거시 정보시스템을 흡수하며 동시에 안전 수준을 향상시키는 시스템을 구축하기 위해서, PCS에 기반을 둔 모바일 데이터베이스 도입을 고려하면서 동반되는 물리적인 제약과 기술적인 문제들을 검토하고, 모바일 환경에서의 트랜잭션을 처리해 주는 두 가지 선택에 대하여 비교한다. 현재까지 알려진 문헌연구와 시뮬레이션 스터디를 통해서 이들 기술을 비교한 다음, 트랜잭션 프로세싱 코디네이터 마이그레이션에 대한 방법을 제시하며, 기술이 가진 한계를 현실적 환경에서 검토해 본다.

키워드 : 공항안전시스템, 모바일 데이터베이스 시스템, 레거시 시스템, 트랜잭션 코디네이터, 시뮬레이션

Ⅰ. Introduction

A major airport in Korea is considering enhancing safety level in its airside that includes runaways, taxiways, ramps, aprons and tank farms.[1] The airside is traversed by aircraft and non-aircraft objects. In order to avert potential accidents or close calls, the airside should be protected by highest safety measures. For example, one may not park or stand within 3 meters of any aircraft unless directly involved in

1) The name of this airport remains concealed by the request of administrating officers. Three authors are grateful for the generous support from Next-Generation Intelligent Airport Systems Development Research under the Ministry of Land Transport and Maritime Affairs of Korea.

the servicing of an aircraft or within 15 meters of an aircraft being refueled. Only authorized personnel or vehicles can access the area, but exceptions should be handled in an orderly manner. The airside is also monitored from fixed observatory facilities inside the airside and crews are communicating with wireless devices. Once objects move or park in the airside, their presence will be identified with the help of database systems that furnish object-related information such as airlines, affiliated service organizations, governmental agencies, or access capabilities, just to name a few.

Currently any area in the airside has one of the following designations. The first is for a landing strip or 'runaway' where aircraft take off or land. The second is for 'taxiway' for aircraft waiting on ground before taking off or after landing. The third is for 'under operation' areas needing service works including maintenance or installations. The last is for 'non-designated' areas that require regular cleaning to prevent hazardous debris from building up. Non-designated zones also work as safety buffers between the outskirts of the airport and the airport itself. When an object moves into any area, information of the object and its current location is spotted by Ground Control that continuously checks whether the object is complying with safety guidelines. "Ground Control is responsible for directing all ground traffic except the traffic on runways. This includes planes, baggage trains, snowplows, grass cutters, fuel trucks, and a wide array of other vehicles. Ground Control will instruct these vehicles on which taxiways to use, which runway they will use, where they will park, and when it is safe to cross runways. When a plane is ready to take off it will stop short of the runway, at which point it will be turned over to Tower Control. After a plane has landed, it will depart the runway and be returned to Ground Control"(Wikipedia, 2007).

Although no serious safety concerns or accidents have been addressed in years, the possibility of Ground Control erring in their judgments still looms large to the airport. For example, a vehicle can accidently pass beyond permitted areas or it may be in a collision course with other objects that are not within eyesight. Or unauthorized vehicle may enter into highly restricted areas. And quite often vehicles have to wait idle for clearance from Ground Control before they proceed into the next area. That wait, delayed by human operators, takes a few minutes with no assurance of absolute safety whatsoever. Further exacerbating the safety problem are unfavorable weather conditions such as heavy fog or rain that definitely contribute to delay in making accurate calls and increase the chance of having catastrophic disasters. Having realized potential safety threats, the airport administration decided to revamp the current safety system, which subsequently prompted the research in this paper along with other proposed solutions.

Under the proposed system, individual users will continuously be in touch with information systems, thus making Ground Control depending less on human interventions but more on information systems. It is also expected that safety guidelines are automatically checked so that Ground Control can make finer judgments under harsh conditions. Of course, there are other factors making mobile systems more attractive than other alternatives. In the first place, the airside area is wide enough to warrant mobile technology: its benefits－reduced frequency interference, frequency reuse, scalability of the network in the future, devices consuming less power－far outweigh benefits of other technologies(Kumar, 2006) and the need for inheriting legacy database systems in use is also factored into the equation(Lai *et al.,* 1995). Furthermore full mobility, autonomous computing and data processing capability of a mobile unit(MU), wireless communications between clients, operation transparent to cellular communications, and scalability in adding or deleting mobile units – the essential advantages of mobile database systems(MDBS) coupled with PCS (Personal Communications Services) – are counted toward a mobile-based system (Forman and Zahorjan 1994; Kumar 2006).

In mobile environments, service areas are broken into signal domains(cells) where clients are identified with a MU – a wireless device with antenna, transceiver, and user interface(Kumar 2006). Examples of MUs are, not limited by but including, PDA(Personal Digital Assistants), laptops with wireless antenna, cell phones, workstations, RFID(Radio Frequency ID) objects, etc. Because of its universal capability of identifying and collecting information on objects, RFID, together with a MDBS, can deliver accurate knowledge on structural problems, can better detect potential threats on ground, and can help accumulate statistical data for future analyses. In fact, RFID is just an example of how the proposed system will be most accommodating toward emerging technologies; designers of the proposed system want to make MUs as general as possible so that Ground Control can be better positioned in diagnosing, predicting, detecting, and correcting problems by selecting whatever technologies they will encounter in the foreseeable future.

In a MDBS, a MU wanders from one mobile base station(BS) to another. Before a MU moves into the domain of a neighboring BS, the current BS negotiates with and releases control to the neighboring BS, which we call 'handoff' of a communication channel(Kumar 2006). Technically, before a handoff takes place, two neighboring BSs negotiate with each other to ensure that the tagged MU is seamlessly served by another database server. Should a handoff take place in the middle of transaction service, we say that transactional service handoff occurs(Imielinski and Badrinath 1994). Upon receiving a transaction service request, a database server invokes a transactional service coordinator that conducts a set of activities guaranteeing full completion of the transaction (Elmagarmid *et al.*, 1995a). There are basically two ways of how a coordinator works: the first is for a single coordinator to take charge of entire transaction processing while the second is dynamically to call a series of coordinators that take care of the transaction. The design choice of one coordinator over the other is an important issue that we will illuminate in section 3 and section 4.

The plan of this paper is as follows. We first give an as-is picture of the system in use and discuss requirements for the proposed system in Section 2. Conceptual description of mobile transactions and technical structure for transaction processing will be discussed in Section 3. In Section 4, we show results from a simulation study comparing two coordinators and formally present a scheme for coordinator migration. The final section addresses potential benefits of the model in the context of safety concerns and discusses practicality and robustness of the proposed scheme along with future research issues.

Ⅱ. Design Considerations and Background Information

In this section, we explain the current airside operation of the airport, present grounds for the proposed system, and briefly discuss technological aspects of MDBS architecture of the proposed system.

2.1 Looking back and looking ahead

Typical with many matured organizations, the airport is depending on a number of legacy database systems created to suit one-of-a-kind purposes. For example, the sewage system of the airport, responsible for handling the wastewater coming out of the airport, must meet water quality regulations imposed by the Ministry of Environmental Protection. One prime source of potential violations is deicing agent, called glycol, which reduces the amount of oxygen if discharged in excessive quantities. Despite that the airport operates several 'deicing pad' for recycling the agent, the water discharged from the airport should be under close inspection to prevent glycol runoff from flowing into nearby waterways. To cope with the problem, a dedicated database system collects water quality data from multiple points.

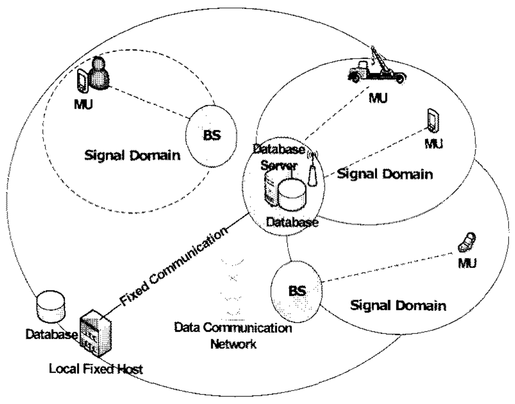
Another database system measures noise levels. When the airport was established several decades ago, the neighboring areas had fewer residents and the noise level was almost non-issue. The picture has radically changed, however, due to the rapid urbanization around the airport, presence of dense residential blocks, and dramatic increase in number of flights. Today, the noise generated from avionic activities is one of the most serious problems to residents living nearby and controlling noise is a serious business. Particularly in concern is the noise coming from a grounded aircraft's engines revved at maximum levels during routine maintenance checks. The collected data is used to draw a noise contour map, to identify areas most susceptible to noise complaints, and to schedule maintenance operations so that noise level may not exceed the maximum allowed by regulating agencies. The data are further used to evaluate complaints filed by residents nearby and for external audits.

The airport has also been inhabited by flocks of birds that should jeopardize aircraft flying around the airport. As such, Ground Control intermittently dispatches its team to critical areas where birds may endanger departing or arriving aircraft, exercising various methods to scare birds away from the airfield. As much as dispelling birds is important, it is also essential to deter flocks of birds from settling at the airport. This includes maintenance of the grass height and elimination of potential plants that birds may feed by using weed killer. Combined with other tools for birds and plants control, the Ground Control makes sure that the habitat is least favorable for birds' roosting. Currently, all activities related to bird control is rendered manually through human inspection via high-power binoculars, bird-control devices attached to a vehicle, and often direct intervention of onsite crew. Although crew members are rewarded by enumerating potential problems during reconnaissance trips, only a smaller fraction of reports as to the exact locations and related activities are actually filed for possible lack of convenient mobile devices that may help such reporting fed into database systems. Equipped with a mobile system and a grid map, Ground Control will be able to dispatch its crew in a precise and timely manner and systematically to gather activity-related data which can be used for external audits and other improvement projects. Last but not the least is the database system for airport management which provides services like aeronautical billing, flight information systems, property management, incident management, or internal staff network. Detailing such services central to the airport operation is clearly beyond the scope of this paper and thus omitted.

Having considered that the legacy database management systems should remain an integral part of the proposed system, we recommended mobile database architecture with each signal domain having a separate database server. The architecture may relatively cost more, need careful synchronization of replicas, and involve solving technical problems such as cache coherence, yet at least three strong advantages seem to offset such drawbacks. First, the airport should be operable on 24-hour basis and replicated database will work for better system performance and higher reliability. Second, the maintenance and upgrades of the legacy system are currently done on individual basis and there is no centralized cost control. Therefore more efficient management would be put into practice if legacy systems are integrated. Third, the architecture will enable individual legacy systems to be retrofitted nicely into the proposed system by defining individual legacy systems as one service domain respectively.

2.2 Technical Description on the Mobile Architecture

In mobile networks, each signal domain has a mobile base station(BS) that is a computer connected to a fixed network. A BS is characterized by a unique frequency that separates one signal domain from another. If a MU walks into a signal domain of a BS, it opens a channel with the BS which takes the role of connecting a MU to a database server associated with the signal domain. When a MU is about to cross a boundary between two neighboring signal domains, the physical signal strength from the original base station weakens while that from the neighboring domains intensifies. And it is best for mobile clients to have as strong a signal as possible.



〈Figure 1〉 Mobile Database System Model

A MDBS generally has a few salient characteristics. First, it provides database functionalities via PCS or GSM by incorporating a set of distributed database servers to PCS(or GSM) so that mobile clients can transparently access databases as if they are using a fixed database(Kumar 2006). Second, a MDBS can sometimes base itself on replicated databases for better availability and reliability(Alonso and Korth 1993). If a query request can be answered by a database server in the same signal domain, there would be less number of transmissions to and from remote servers, which effectively shortens transmission time and thus enhances reliability in wireless communications. However, the operational gain comes at the cost of having to maintain the consistency of distributed databases, effectively increasing communication overhead between database servers and base stations.

Complicating the issue would be a transaction service request that guarantees consistency of databases. In mobile environments, when a mobile client happens to move from one domain to another in the middle of a transaction, a transaction service coordinator will be called for. Transaction service coordinators are usually embedded in mobile support base stations because base stations, transmitting messages back and forth between database servers and pertinent MUs, will be engaged with transaction processing of MUs(Kumar 2006).

There are two architectural alternatives in dealing with transaction-related crossovers. The first approach is for one coordinator to continuously perform the processing without releasing the control while the second is for a coordinator to relinquish the control to another coordinator in the next service domain. The former architecture is called transaction processing of coordinator settlement(TP/CS)(Meng *et al.*, 2002) and the latter transaction processing of coordinator migration(TP/CM)(Zhou *et al.*, 1996). With TP/CS architecture, one coordinator takes control of the transaction to the completion of the transaction even after a handoff occurs and after the tagged MU moves outside of the current signal domain. That also means that communication signals will be physically weakened as the tagged MU walks farther away from the original base station. Although communicating through *two* or more base stations would be possible and the simple architecture of TP/CS may work well for smaller signal domains, it does not work well for wider signal domains. Further the number of messages between base stations will increases with TP/CS as a single coordinator should be in control of *all* base stations involved with a specific transaction.

On the other hand, TP/CM architecture tends to be more complex as it should transfer control to another coordinator and unfinished transaction has to be annulled only to be restarted from scratch in the next signal domain. That is, the TP/CM may penalize the system performance by forcing a transaction to start afresh repeatedly. Yet TP/CM potentially works better under weak signals than TP/CS, thus making itself a more flexible solution. For the other reasons stated in section 4, we focus on TP/CM for the rest of the paper except when we do simulation analysis.

Ⅲ. Location Management of Mobile Units

In identifying a MU, two database management systems—one associated with a MU and the other with database servers germane to the MU—should be engaged together(Kumar 2006). That is, how database servers recognize a MU and communicate with a MU is pretty much similar to how servers do so with clients in typical client/server architecture; database servers in MDBSs have information on individual MUs; MUs are uniquely identified by their unique IDs independent of their current positions(Jing *et al.*, 1999; Dunham *et al.*, 1997). Yet the full mobility and autonomous computing capability of MUs pose additional set of requirements. For example, wireless communications are inherently unreliable and thus can make a MU disconnected unexpectedly; a MU can shut itself down or turn into doze mode if battery power goes low; types of MUs can be extremely diverse, ranging from RFID-based device and PDA, to remote sensors collecting onsite data(Shen *et al.*, 2001; Xu 2003); a MU moves from one signal domain only to be disconnected temporarily for some reason before it gets reconnected to another signal domain. Thus locating different types of MUs seems highly crucial to MDBS.

3.1 Location Management Model

In MDBSs, three components work for locating a MU: a MU itself, mobile base station, and fixed database server. A MU communicates with database servers through a mobile base station which defines a signal domain and provides secure communications for a MU and pertinent database servers. Before a MU starts a transaction, its location will first be reported to the closest base station which then sets up proper communication links to database servers. Thus the location of the MU needs to be constantly tracked down and location management is extremely important to completing a transaction request.

3.1.1 Database Management Systems of a Mobile Unit

A MU, being a computer itself, has a database management system and mobile communications controller(MCC) that sets up a connection with database servers, as shown in <Figure 2>. A database management system inside a MU consists of query processor(QP), transaction manager(TM), and storage manager(SM). Query processor verifies representation form of queries, transforms queries into operations, optimizes queries, and transmits their operations to a transaction manager. Transaction manager executes operations requested by query processor, and then sends data to storage manager if data are found in database servers. If not found, the manager fetches the query through mobile communication controller so that results can be obtained from database servers of other signal domains(Pitoura and Samaras 1997).

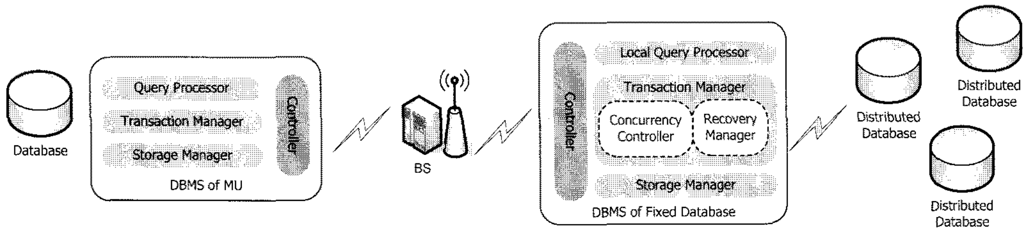
Storage manager maintains data dictionary of data objects that are stored in its own database. After analyzing an operation requested by transaction manager, storage manager reads data from database servers and stores the result in input/output queue so that transaction manager transmits the result to query processor.

3.1.2 Database Management Systems of Fixed Database Servers

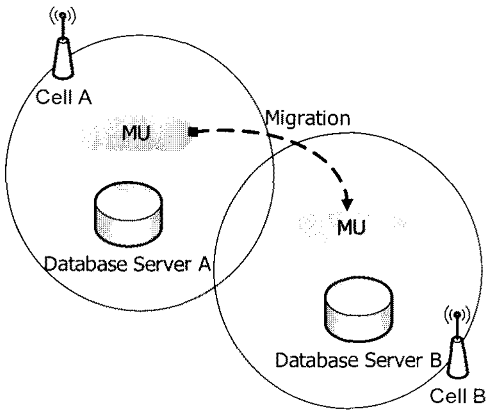
Database systems in database servers consist of local query processor(LQP), transaction manager(TM), and storage manager(SM). Local query processor handles queries requested by fixed transaction terminals, verifies presentation forms of queries, and classifies them into operations for database service before it optimizes queries for better performance. To get appropriate results from parallel transactions, transaction manager audits a transactional procedure and coordinates completion or abortion of transactions. It comprises concurrency controller(CC) and recovery manager(RM): CC is for getting appropriate results from parallel transactions and RM for recovering database in case of errors before completion of a transaction. CC also controls parallel operations of transactions from a local query processor and from MUs. Storage manager performs operations sent from CC that has data dictionary of database in servers. With the help of data dictionary, storage manager reads data from database before it stores them in input/output queue.

3.2 Mobile Transaction Processing Scheme by Coordinator Migration

In spite of significant research on transactional service handoff in the past, we were unable to find an appropriate solution for the proposed MDBS. For example, although Pitoura and Bhargava(1994a, 1994b) and Chrysanthis(1993) introduced a location management model and concepts such as transaction structure, data consistency, and transaction relocation, they



<Figure 2> Mobile Database System Model

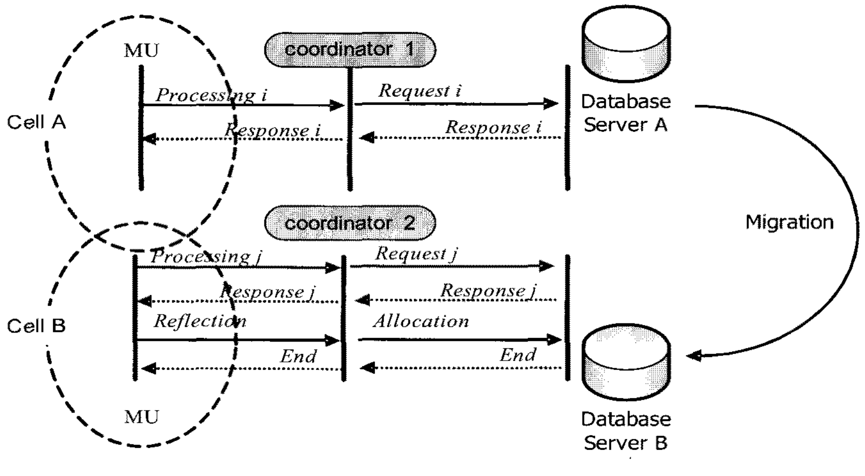


〈Figure 3〉 Transaction Processing Model for Mobile Database Systems

did not show how a MU and databases can be interwoven together to initiate and complete a transaction, let alone proposing a transaction-related scheme; Alonso and Korth(1993) addressed a problem caused by connection failures between mobile and non-mobile units but did not offer a solution practical enough; Jing *et al.*(1995) applied optimistic two-phase locking (O2PL) algorithm to mobile environments only to leave much room for improvements. The scheme presented later in this paper would like to fill the gap left by those previous works.

In transaction processing of coordinator migration(TP/CM), a coordinator migrates into another signal domain to transfer its control to the next coordinator in case a transactional service handoff occurs. Since all operations related to a transaction have to be executed atomically, unfinished transaction would have to be rolled back when coordinator migration takes place. The next coordinator then has to have the transaction processed from scratch. <Figure 4> shows an example illustrating how such coordinator migration works.

In <Figure 4>, we assume that operation *i*, part of a transaction requested by a MU, is being executed in signal domain 1. Operation *i* is transmitted to coordinator 1 and the coordinator 1 asks database server A for processing. If an operation *i* is doable, a message response *i* of approval is transmitted to coordinator 1 and the coordinator 1 transmits the response *i* to the MU accordingly. Now suppose that the MU that receives response *i* moves from signal domain 1 (Cell A) to signal domain 2 (Cell B). And also



〈Figure 4〉 An Example of Transaction Processing Using Coordinator Migration suppose that the transaction is reprocessed by coordinator 2 from scratch and that all operations except the last operation *j* is performed.

If operation *j* is transmitted to coordinator 2 from the MU, coordinator 2, upon receiving operation *j*, asks database server B to have operation *j* processed. Database server B transmits response *j* to coordinator 2 if processing the operation is allowed. And coordinator 2 transmits response *j* to the MU in signal domain 2. Finally, having found that all operations are completed, the MU issues a commit operation to coordinator 2 to finalize the transaction, which triggers coordinator 2 to ask database server to commit to the transaction. If allocation of data is finished, a completion message from database server B is sent back to a MU through coordinator 2.

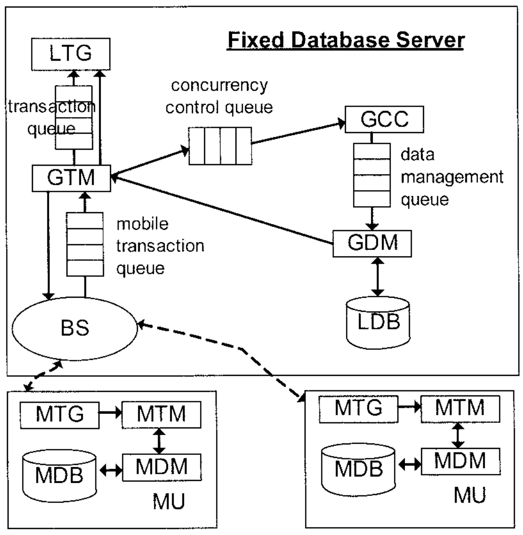
Ⅳ. A Mobile Transaction Processing Scheme for the Airport

Having come up with two alternatives for transaction processing coordinators, we first compare their performance, using a simulation analysis by varying the number of MUs and the velocity of vehicles. Data structure and associated pseudo-code follow, explaining how they can be integrated to meet specific requirements for the case of coordinator migration.

4.1 Simulation Settings

In this subsection, we compare two alternatives through a simulation study by focusing on throughput and average response time when we vary the number of active MUs and the velocity of moving vehicles respectively. In an intuitive sense, the response time will increase (and the reverse is true with throughput) as more of active MUs are present and/or vehicles are driven at faster speed. For the sake of simplicity, we assume that a single user is associated with a MU at any given time and that communication delay between a MU and a related server is identically distributed. The simulation was performed on SunSparc 20' workstation(OS: Solaris 2.5.1), using CSIM, a package suited for simulating a dynamical environment and also known to generate highly reliable results(Schwetman 1992).

As is shown in <Figure 5>, MUs and database servers in the simulation are supposed to interact with each other dynamically given parameter values. A MU in the simulation study has mobile transaction generator(MTG), mobile transaction manager(MTM), mobile data manager(MDM), mobile communication manager(MCM), mobile database(MDB), while a fixed local server consists of local transaction generator (LTG), global transaction manager(GTM), global concurrency controller(GCC), global data manager (GDM), communication manager(CM), and local database(LDB)(Weikum and Vossen 2001). MTG of a MU instantiates a transaction before it sends the new-



〈Figure 5〉 Simulation Model for Mobile Transaction ly created transaction to MTM. If the transaction is 'read' only, it will be sent to MDM that checks whether the data item can be retrieved from a local database server. If found locally(the case of cache hit), the retrieved data will be sent to MDM that fetches the retrieved data to a MU. In case that the item is not in a local database server, MDM sends 'cache miss' message to MTM which relays the message to a fixed database server. If the transaction is related to 'update', the request is sent to a database server associated with an appropriate signal domain.

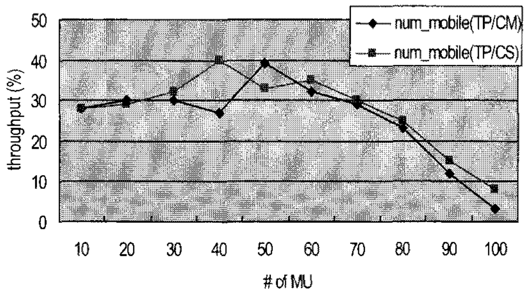
The server-side transaction manager(GTM) takes care of transactions occurring between fixed database servers, transactions occurring between a MU and a fixed server, and transactions that should be sent to other database servers residing in other signal domains because the data item is not found locally. These transactions will be sent to GCC that prioritizes the order of processing based on optimistic two phase locking protocol(O2PL). The result of GCC is stored into data management queue and it will be retrieved by GDM later. More detailed description of the simulation model can be secured by asking the authors.

4.2 Simulation Results

We use an identical set of input data for simulation runs and have simulation results averaged so that the results are within the boundary of 90% confidence interval. In order to avoid initial bias, the first 5% of simulation output are truncated(Law and Kelton 1999). Also values showing excessively long delays(10% of the total) are not counted in the final analysis as they are believed to be malfunctioning (thus causing exorbitant delays) MU devices.

The number of active MU's in the airside is used as an independent variable of the simulation. We also set the transaction data size to 10, the chance of having 'write' operations to databases to 0.25, the range of a signal domain to 2km, and average speed of a moving vehicle to 10 km/hour respectively. The 0.25 chance of having 'write' operation signifies that 'read' operations such as locating and/or identifying MUs take place more frequently than 'write' operations. Using these parameter values and changing the number of MUs by 10, we first come up with <Figure 6> showing throughput performance where throughput is defined as:

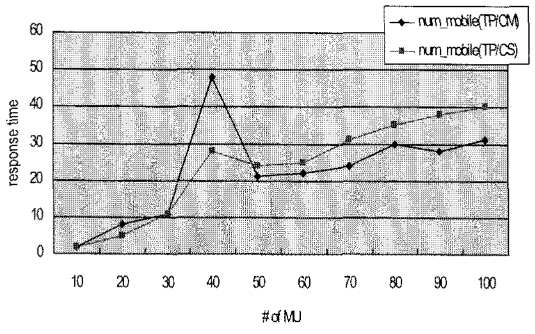
throughput(%) = (number of processed transactions/simulation time) * 100.



⟨Figure 6⟩ Throughput vs. Number of MUs per unit time

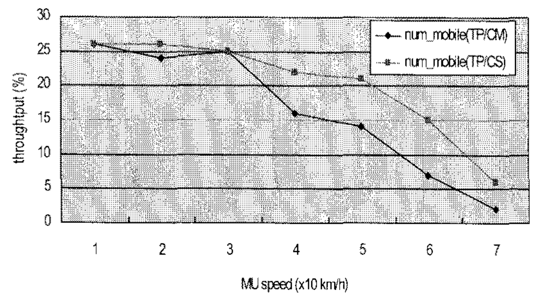
<Figure 6> does not show significant advantage of one scheme over another except the apparent disparity observed in the interval between 30 and 50. We suspect that the system is run under capacity when the number of MUs is less than or equal to 30(for the case TP/CS) and 40(TP/CM), but after reaching these values, the system is suffering from increased migration overhead compounded by cache misses in fixed database servers. That could be why both graphs spike before decreasing steadily, which is expected.

<Figure 7> is about the average response time(in millisecond) versus the number of MUs where we



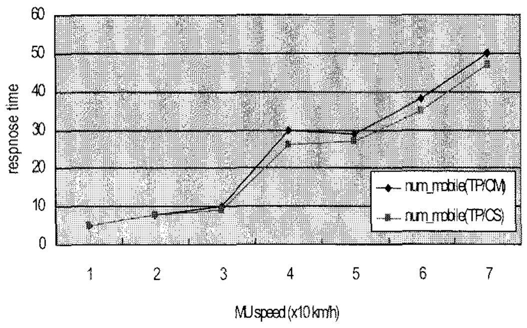
〈Figure 7〉 Average Response Time vs. Number of MUs

aggregate the total service time(including delays) of completed transactions and divide the aggregation by the number of completed transitions. Both alternatives peak at about 40 MUs and stabilize thereafter, which we believe are attributable to migration overheads. The migration overhead of TP/CM slightly dominates that of TP/CS until the number of MUs reaches 50 but TP/CM performs relatively better afterwards.



〈Figure 8〉 Throughput vs. Velocity of MUs

We also examine the impact of velocity on throughput as is depicted in <Figure 8> Here TP/CS works slightly better with the velocity equal to or less than 30 km/hour, and the disparity between the two alternatives widens at higher speed. As we pointed out, a vehicle(or MUs) with higher speed tends to change between signal domains more often than not and the migration overheads accelerate with speed increasing, which the graph confirms consistently.

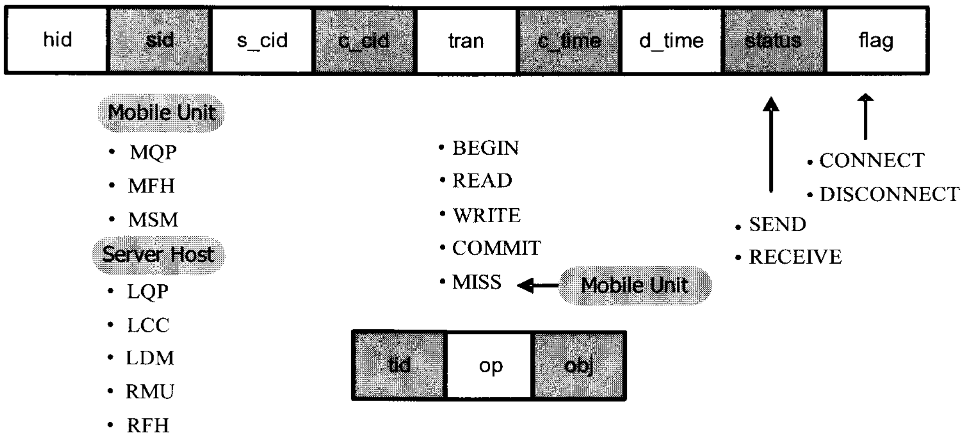


〈Figure 9〉 Average Response Time vs. Velocity of MUs

Finally <Figure 9> shows that velocity affects response time negatively and that the disparity between two alternatives remains marginal in all velocity values. As is the case with other simulation results, the response time increases steeply after 30 km/hour, suggesting that our simulation model is robust or at least consistent with the other independent variable (i.e., the number of MUs) and our conjectures stated previously may hold in this case too.

Although the simulation study is rendered with limited set of parameter values, the outcome of the study is rather consistent with findings from other research except that the performance measures takes off or peaks after reaching a certain value(i.e., 30 km/hour or 40 MUs). Once again, the dramatic change in performance may be due to the divulgence of migration overheads complicated by cache misses.

The simulation study delivers somewhat mixed results. First and foremost, the performance difference between the two alternatives seems insignificant except the third result (the throughput vs. velocity) which shows that TP/CS works better with MUs running over 30 km/hour. The seemingly perplexing out-



〈Figure 10〉 Data Structure for Mobile Transaction

come may indicate that further refinement of the simulation is needed. For example, using trace-driven simulation may be considered for the refinement of the model, as is the case with typical performance simulation study in computer networks. Furthermore, the simulation was not run on a two-dimensional variable: to make the model closer to reality, we should have defined the joint distribution of number velocity of MUs and their velocity. Yet, for lack of sufficient data available, we were not able to pursue into that direction. Actually, a stronger ground favoring TP/CM comes from the communication overhead that TP/CS inherently pays dearly: if a coordinator with TP/CS does not migrate to another service domain, it suffers from weaker signals when a MU walks farther from the original base station, which necessitates increased communication overheads. Therefore, in the next subsection, we propose a transaction processing scheme for TP/CM.

4.3 Transaction Scheme with its Data Structure

The data structure used for transaction processing is given in <Figure 10> and <Table 1> lists value types of individual fields used in the data structure. Note that data items are used either by a MU or a fixed data server and thus can basically be broken into two disjointed categories.

The pseudo-code in <Table 2> illustrates how a transaction manager works. The right column of the table has explanations about corresponding pseudo-code counterparts. As we saw in Section 3, MUs do not have separate concurrency controller and the task of controlling concurrency is relegated to the concurrency controller of a(remote or local) database server. The first line of the code is **receive(msg)** indicating that the beginning of transaction processing starts with checking the sender of the message. If the message is from a local query processor(LQP) asking a transaction processing, then the request will be passed to a local concurrency controller(LCC). Otherwise it must be from a remote MU that initiated the transaction processing a while ago and the message should be processed accordingly. Brief comments in the right column of the table expound how each corresponding segment of codes works in the subsequent codes.

〈Table 1〉 Identifiers and Value Types for Mobile Transaction

Identifiers	Value Types
hid	A unique identifier assigned to a MU. This information is a part of transactional data and database servers exchange messages with MUs, using this id.
sid	Specify transaction type by recording the entity that requested the current transaction. The value for sid should be one of the following. ◦ MQP: A message sent by Mobile Query Processor ◦ MSM: A message sent by Mobile Storage Manager ◦ MFH: A message sent by a local database server(a.k.a. Local Fixed Host) ◦ LQP: A message sent by Local (server-side) Query Processor ◦ LCC: A message sent by Local (server-side) Concurrency Controller ◦ LDM: A message sent by Local (server-side) Data Manager ◦ RMU: A message sent by a Remote Mobile Unit(i.e., not within the current domain) ◦ RFH: A message sent by a Remote Fixed Host(database server)
s_cid	A unique identifier of a coordinator that initiated transaction processing for the first time. The ID is stored into the mobile base station associated with the coordinator.
c_cid	A unique identifier of a coordinator in the current signal domain. If s_cid is not same as c_cid, that means handoff has occurred before moving into the current domain.
tran	Data structure of a transaction ◦ tid: a unique identifier assigned to a transaction ◦ op: transaction-specific operations (one of the following will be assigned to 'op') BEGIN: starting time of transaction processing READ: read operation WRITE: write operation ABORT: abort operation COMMIT: reflection on database MISS: object for operation does not exist in mobile database at the time.
c_time	Time to be connected to a fixed database server for the first time
d_time	Time to sever connection with a fixed database server
status	State information regarding a MU (either one of the following): ◦ SEND: sending messages ◦ RECEIVE: receiving messages
flag	Indicator as to the connection state with a fixed database host(either of the following) ◦ CONNECT: connected state ◦ DISCONNECT: disconnected state

V. Concluding Remarks

This paper is a byproduct of research project initiated by a major airport in Korea in pursuit of overhauling its airside safety. Faced with potential safety-related problems, they explored a few IT solutions to enhance Ground Control service. After carefully examining safety goals, requirements, and specific settings, and the need to accommodate existing legacy systems, a PCS-based MDBS was proposed. The

〈Table 2〉 A Scheme for Coordinator Migration

Start dynamic_server_transaction **Begin**	◦ A scheme for TP/CM manager

receive(msg)
case(msg->sid)
	◦ After receiving a message, the manager recognizes the sender by reading the message.

LQP: send(LCC, msg->tran) **break**	◦ If the message is from a local query process, send the message to the local concurrent controller(LCC).
RMU: if(msg->s_cid != c_cid) Begin	◦ If it is from a remote MU(RMU), check if handoff has occurred somewhere else.
import_task(s_cid) **msg->s_cid = c_cid** **End**	◦ If so, get the original identifier of the transaction and replace the current ID with the original ID

send(LCC,msg->tran)
break
	◦ Send the message to the server-side concurrency controller

LCC: msg->tran->op = ABORT
if(msg->sender = RMU) Begin
msg->status=ABORT
send(msg->hid, msg)
End
else send(LQP, msg->tran)
break
	◦ If the message is from server-side concurrency controller, this is the case where LCC is not able to process the transmission to full extent. That being the case, we designate the transaction as ABORT and send the message to either LQP or RMU.

LDM: if(msg->sender = RMU) **send(msg->hid, msg)** **else send(LQP, msg->tran)** **break** **End** **End** **End start**	◦ If the message is sent from a server-side data manager(LDM), we send the message back to its sender(either a remote MU or a local query processor).

search for a workable solution lead us into a rather technical problem known as transactional service handoff. Putting the problem into the airport's unique perspectives and under a simulation study, we forformalize a transaction processing scheme for TP/CM, which is an important issue but has not been adequately addressed to date.

Let us mention some potential benefits that the proposed system could offer. First, we note that there are at least 1,700～1,800 cases of endangering objects annually reported by Ground Control. The proposed system could help Ground Control wirelessly sending such safety-related data to database servers without a hitch. Those data, accumulated over years, should help predict which area will be most vulnerable to threats from hazardous objects. We also suggest that the mobile system should be able to accommodate various mobile platforms that can provide convenient means of reporting on-site case information to database servers. Secondly, Ground Control will be able to dispatch vehicles in a timely and precise manner to cope with such emergencies as unexpected fire or forced landings. Thirdly, onsite crew members with RFID-based tags can be identified with their access privileges in real time, which should improve operational efficiency. Fourthly, the new system will make the job of controlling traffic in the airside much automated and will thus contribute to minimizing human intervention as much as possible. And there would be little chance for unauthorized vehicles to enter highly protected areas, which is quite problematic even today.

Of course, the scheme proposed here is open to questions as to its practicality and robustness. We admit that the algorithmic performance has yet to be validated by actual usage data. The airport is serving 130 planes per day and there are about 800 vehicles that run on routine schedules on 24-hour basis. A more detailed simulation study that factors in these numbers and schedules should help refining our proposal and could be studied in the future. A closely related, yet highly technical, issue is the impact of using optimistic two-phase locking. Because any locking scheme will negatively affect the system performance, the question boils down to how often and how long a locking instance happens on the average and how serious it can slow down the system performance in worst cases(Mannino 2005). Although this technical question may have to be answered in another paper and lock-free algorithms can also be considered as an alternative, we expect that the chance of freezing the system due to locking would be fairly small and the system performance will remain robust, given that there is little chance for 800 vehicles working simultaneously and that general business environments of the airport have been fairly stabilized.

Lastly we note that a sensor-based network was considered as an attractive alternative to our proposal, but its drawbacks – sensors having to be densely deployed let alone their limited power, penchant for failure, limited computational ability and upgradability – seem to eclipse advantages of using a sensor-based solution in our case. Besides we later learned that another team is focusing on this avenue of research for handling passenger and baggage claims inside passenger terminal areas, thereby deciding not to pursue the option.

References

Alonso, R. and H. F. Korth, "Database Issues in Nomadic Computing", in *Proceedings of The ACM SIGMOD Conference on Management of Data*, Washington D. C., U.S.A., May 1993, pp. 388-392.

Carey, M. J. and M. Livny, "Conflict detection tradeoffs for replicated data", *ACM Transactions on Database Systems*, Vol.16, No.4, 1991, pp. 703-746.

Chrysanthis, P. K., "Transaction Processing in Mobile Computing Environment", in *Proceedings of the IEEE Workshop on Advances in Parallel and Distributed Systems*, Princeton, New Jersey, U.S.A., October 1993, pp. 77-83,

Dunham, M., A. Helal, and S. Balakrishnan, "A mobile transaction model that captures both the data and movement behavior", *Mobile Networks and Applications*, Vol.2, 1997, pp. 149-162.

Elmargamid, A. K., J. Jing, and O. Bukhres, "An Efficient and Reliable Reservation Algorithm for Mobile Transactions", *Proceedings of the 4th International Conference on Information and Knowledge Management*(CIKM 1995), 1995, pp. 1-26.

Elmagarmid A. J. Jing and T. Furukawa "Wireless client/server computing for personal information services and applications", *ACM SIGMOD*, 1995.

Forman G. H. and J. Zahorjan, "The Challenges of Mobile Computing", *IEEE Computer*, Apr 1994, pp. 38-47.

Imielinski, T. and B. R. Badrinath, "Mobile Wireless Computing: Challenges in Data Management", *Communications of the ACM*, Vol.37, No.10, Oct 1994, pp. 18-28,

Jing, J., A. Halal and A. K. Elmargamid, "Client-Server Computing in Mobile Environments", *ACM Computing Surveys*, Vol.31, No.2, June 1999, pp. 117-157.

Jing, J., O. Bukhres and A. K. Elmargamid, "Distributed Lock Management for Mobile Transactions", *Proceedings of the 15th IEEE International Conference on Distributed Computing Systems*(ICDS'95), Vancouver BC, Canada, May 1995.

Kumar, Vijay, *Mobile Database Systems*, John Wiley and Sons, 2006.

Lai, S. J., A. Zaslavsky, G. P. Martin, and H. L. Yeoa, "Cost Efficient Adaptive Protocol with Buffering for Advanced Mobile Database Applications", *4th International Conference on Database Systems for Advanced Applications*, DASFAA 1995, Singapore, Apr 1995, pp. 10-13,

Law, A. and W. D. Kelton, *Simulation Modeling and Analysis*, 3rd edition, McGraw-Hill, 1999.

Mannino, M. V., *Database Design, Application Development, And Administration*, 3rd Edition, McGraw Hill, 2005.

Meng, X., J. Su, and Y. Wang, "Advances in Web-age Information Management", *Lecture Notes in Computer Science*, Vol.2419, Springer, 2002.

Pitoura, E. and B. Bhargava, "Building Information Systems for Mobile Environments", in *Proceedings of The 3rd International Conference on Information and Knowledge Management*, Gaithesburg, MD, U.S.A, Nov 1994, pp. 371-378.

Pitoura, E. and B. Bhargava, "Revising Transaction Concepts for Mobile Computing", in *Proceeding of the IEEE Workshop on Mobile Systems and Applications*, Santa Cruz, CA, Dec 1994.

Pitoura, E. and G. Samaras, *Data Management for Mobile Computing*, 1st Edition, Springer, 1997.

Schwetman, H., *CSIM User's Guide for Use with CSIM*, Revision 16, Microelectronics and Computer Technology Corporation, 1992.

Shen, C., Srisathapornphat C., and C. Jaikaeo, "Sensor information networking architecture and applications", *IEEE Personal Communications*, Vol.8, Aug 2001, pp. 52-59.

Weikum, G. and G. Vossen, *Transactional Information Systems: Theory, Algorithms, and the Practice of Concurrency Control*, 1st edition, Morgan Kaufmann, 2001.

Wikipedia, http://en.wikipedia.org/wiki/Airport, Feb 2007.

Xu, N., "A Survey of Sensor Network Applications", Survey Paper for CS694a, Computer Science Department, University of Southern California, 2003.

Zhou, T., C. Pu, and L. Liu, "Adaptable, Efficient, and Modular Coordination of Distributed Extended Transactions", *Proceedings of the fourth international conference on Parallel and distributed information systems*, 1996, pp. 262-273.

Information Systems Review
Volume 10 Number 2
August 2008

A Transaction Scheme for Managing Mobile Units in Wireless Environments: A Case of Airport Safety Control

Yong Jae Kim[*] · Hyunsup Ahn[**] · Seogjun Lee[***] · Doowoon Hyun[****] ·
Yonggoo Choi[*****] · Sangwon Lee[**]

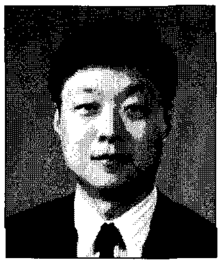
Abstract

This research is a byproduct of a project initiated by a major airport in Korea. The airport aimed to enhance airside safety and weighed a few IT options that may strengthen its current safety practices. In order to satisfy requirements unique to the airport and to inherit existing legacy systems, a PCS-based mobile database system was proposed. We first discuss technical aspects of the system as well as design issues in mobile systems before we compare two specific alternatives for transactional service handoff. A simulation study comparing two types of coordination follows before we present a scheme that manages transaction processing and locating service in mobile environments. Finally, we discuss the limitations of the scheme from the perspectives of the airport in question along with other research issues.

Keywords: Mobile Database Systems, Legacy Systems, Transaction Coordinator, Simulation

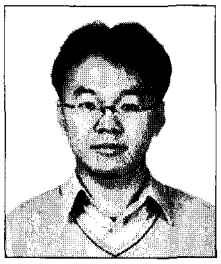
[*] College of Business, Konkuk University
[**] KAIST Business School
[***] College of Business, Konkuk University
[****] Asiana IDT Inc., SI Delivery Team
[*****] Dong-Seoul College, Department of Computer Information

◐ 저 자 소 개 ◑



김 용 재 (yjaekim@gmail.com)

서울대학교 경제학과를 졸업하고, SUNY Stony Brook에서 경제학석사, University of Kansas에서 Computer Science 석사, University of Washington에서 Information Systems 전공으로 경영학박사를 받은 다음, 건국대학교 경영정보학과에 교수로 재직 중이며, University of Colorado Denver에서 Information Systems 전공 조교수로 2년 반을 재직하였다. 주요 연구관심분야는 네트워크 효율성을 위한 가격 정책, 시스템 평가방법론 등이다.

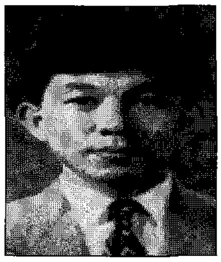


안 현 섭 (coolahnphd@gmail.com)

건국대학교 경영정보학과 학사, KAIST 경영공학 석사를 취득하고 현재 박사과정에 재학 중이다. 요구형성 방법론 비교, 데이터베이스 응용 등에 관심을 두고 있다.

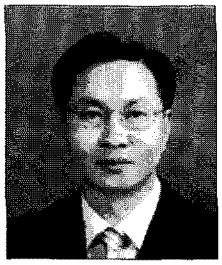
이 석 준 (seogjun@konkuk.ac.kr)

고려대학교 산업공학과에서 학사와 석사 학위를 취득하였고 University of Wisconsin에서 산업공학 박사학위를 취득했다. 현재 건국대학교 경영정보학과 교수로 재직하고 있다. 주요 관심분야는 정보화 성과관리 및 평가, Enterprise Architecture, Business Intelligence, 정보기술 관리, e-Health 등이다.



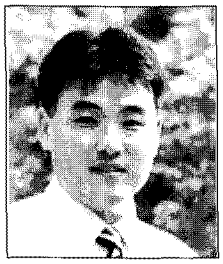
현 두 운 (dwhyun@asianaidt.com)

한국항공대학교를 졸업한 후, 아시아나 항공에 입사 현재 아시아나 IDT 영업지원 팀 차장으로 근무 중이며, 공항 및 항공분야에서 네트워크, 보안등 정보통신 분야에서 15년 이상 일하면서, 인천국제공항공사와 한국공항공사의 u-Airport 마스터플랜에 참여하였고, 인천국제공항공사 공용단말 환경 시스템 구축하였으며, 현재 공항시설 안전 관리를 위한 시스템 개발, 현재 차세대 지능형 공항 개발 구축 R&D 사업 책임 연구원으로도 일하고 있다. 공항 및 항공분야에 특화된 중장기 마스터 플랜 수립과 이에 관련된 컨설팅 업무 그리고 RFID/USN 등 최신 IT기술을 활용한 공항 운영과 관리가 주요 연구관심사이다.



최 용 구 (ygchoi@dsc.ac.kr)

KAIST에서 경영공학 박사학위를 취득하였고, 현재 동서울 대학교에서 부교수로 재직 중이다. 관심분야는 데이터베이스 시스템, XML, 데이터마이닝 등이다.



이 상 원 (sangwonlee@business.kaist.ac.kr)

한양대학교 수학과를 졸업 한 후, 2002년 KAIST MIS-MBA를 받은 후, 대우정보시스템에서 근무하면서 정보시스템 전략기획, 시스템 분석 및 설계 업무에 종사했고, 현재 KAIST 경영공학전공 박사과정에 재학 중이며, 엔터프라이즈 데이터 모델링, 데이터 웨어하우징 그리고 XML 기술응용과 이들 기술이 경영에 미치는 영향평가에 연구관심을 두고 있다.

논문접수일 : 2008년 07월 04일 게재확정일 : 2008년 07월 27일