Advances in Wireless Communications and Networks
2019; 5(1): 1-12
http://www.sciencepublishinggroup.com/j/awcn
doi: 10.11648/j.awcn.20190501.11
ISSN: 2575-5951 (Print); ISSN: 2575-596X (Online)

A Proposed Combinatorial System Design for Ubiquitous Transaction Processing Systems

Patience Spencer

Department of Computer Science, Ignatius Ajuru University of Education, Rumuolumeni, Port Harcourt, Nigeria

Email address:
patsyspency2013@hotmail.co.uk

To cite this article:
Patience Spencer. A Proposed Combinatorial System Design for Ubiquitous Transaction Processing Systems. *Advances in Wireless Communications and Networks.* Vol. 5, No. 1, 2019, pp. 1-12. doi: 10.11648/j.awcn.20190501.11

Received: June 24, 2019; **Accepted**: July 27, 2019; **Published**: September 3, 2019

Abstract: As computing paradigm shift from a computing paradigm involving one-computer-many people to that involving one-person-one computer and eventually to the one involving one-person-many computers, the need for effective transaction management model for this advancement has also increased. This is because, new transaction management challenges are introduced. These challenges include increased mobile user bank, hybrid of mobile devices and transaction processing architecture related issues. This paper presents a Combinatorial System Design of Transaction Processing Elements for Ubiquitous Computing with the aim of justifying the choice of deploying Mobile-3PC Protocol on Three-tier transaction processing system architecture as the appropriate combinatorial system design for ubiquitous transaction processing systems. To achieve this aim, existing transaction processing systems are critically analysed and Compared against standards that influence transaction processing throughput and response time positively. A systematic analytical approach is used in analyzing the organizational structure of two-tier and three-tier system architectures. Subsequently, 2 Phase Commit and 3 Phase Commit communication protocols are analyzed and deployed on the three-tier system architecture to ascertain which one of the combinational transaction processing system design support ubiquitous computing effectively. The study shows that the Mobile-3 Phase Commit Protocol on Three-Tier system architecture displayed proactive management skill to curb process failures. This signifies higher transaction throughput. The inherent load balancing capability of the three-tier system architecture also shows support for improved response time. It is therefore recommended that the Mobile-3PC Protocol-on-Three-Tier system architecture be adopted as the combinatorial system design for ubiquitous transaction processing systems.

Keywords: Ubiquitous Computing, Combinatorial, Architectural Design, Two-Tier System, Three-Tier System, Mobile 2-Phase Commit Protocol, Mobile 3-Phase Commit Protocol

1. Introduction

The implementation of ubiquitous computing [1] systems without a careful selection of transaction management models is detrimental. This is owed to the fact that the advancement of this computing paradigm introduces new challenges. It is therefore evident that a proactive ubiquitous transaction processing model capable of addressing new challenges associated with huge bank of mobile users, hybrid of mobile devices and transaction processing architectures is required. This paper presents a combinatorial system design of transaction processing elements for ubiquitous computing with the aim of justifying the choice of deploying Mobile-3-Phase Commit (3PC) Protocol [2] on a Three-Tier system architecture [3] to achieve a proactive transaction processing system model for ubiquitous computing environment. To achieve this aim, existing transaction processing architectures are identified, critically analysed and compared against standards that influence transaction processing throughput and response time positively. A systematic analytical approach is used in analysing the organisational structure of two-tier [3] and three-tier system architectures [3]. Subsequently, the Mobile 2-Phase Commit [2] and 3-Phase Commit [2] communication protocols are analysed and deployed on a 3-tier system architecture [3]. The information provided by the study supplement transaction processing system models providing useful information for implementation of transaction processing elements in ubiquitous computing environments.

2. Materials and Methods

Basically, ubiquitous computing paradigm is concerned with the ability of a user with a mobile computing device (wearable and handheld) to be able to access information residing in different computing systems as though the information is in the user's computing system [4].

A schematic diagram of ubiquitous computing environment showing features of ubiquitous computing environment is presented in figure 1.

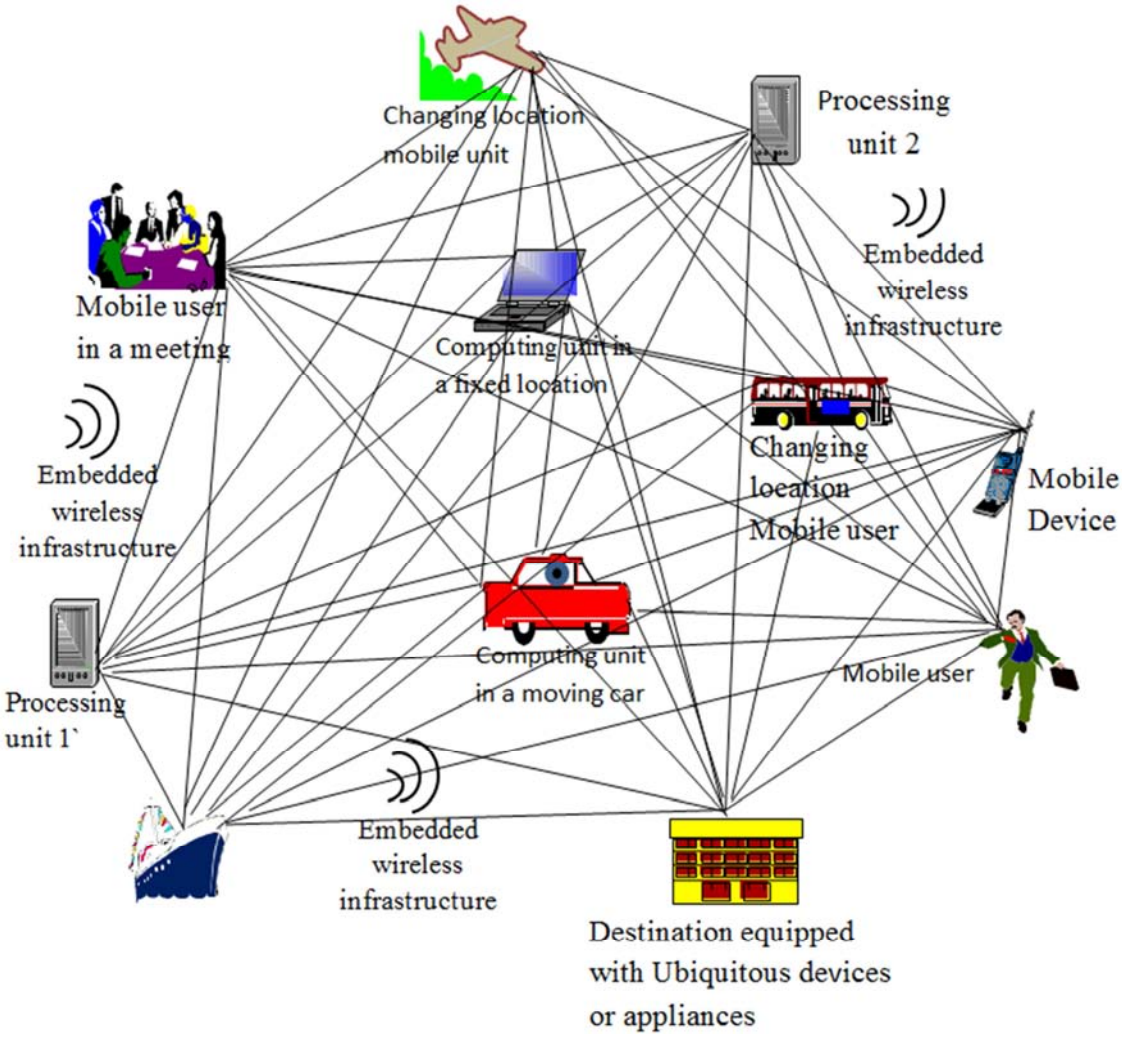

Figure 1. *Ubiquitous computing environment with strong interconnectivity. Adapted from http://sce2.umk.edu/csee/kumarv/mbt-seminar.ppt, by V. Kumar, 2012.*

Figure 1 is a display of what ubiquitous computing environment looks like. In this sample, two processing units labelled processing units 1 and 2 are configured to process requests received from hybrid of mobile devices. Mobile users in this kind of computing environment can access their home appliances from anywhere such as the office, a ship on the sea, a bus, and a taxi.

Common features of ubiquitous computing environment are changing location, mobile units, wireless infrastructure, fixed location mobile units [4]. These features allow users to work from anywhere using any kind of computing device. The activities of a mobile user cause the user to move about, and may be required to access variety of transparent heterogeneous databases with a smart management system (That is, having the ability to learn, collaborate, and be autonomous). The presence of high user interaction with transaction processing systems resulting in low throughput is not desirable in this type of computing environment.

A transaction is defined as a collection of several operations that form a single logical unit of work [5]. An entire transaction that consists of sub-transactions can be distributed across different processing units in the environment. These processing nodes are connected to one another through a communication network infrastructure. Distributed transaction processing and wireless network infrastructure form a backbone for ubiquitous computing [6]. Wireless devices and supporting programs are essential elements of ubiquitous computing environments. This helps mobile users to interact with processing systems within the environment in a transparent manner. To ensure that data is readily available and that mobile users interact with processing systems in an optimal manner, an effective data

communication technique is required. Mobile devices designed for ubiquitous computing can be handheld or wearable devices and a user can carry or wear more than one mobile device.

2.1. Transaction Processing Architectures in Ubiquitous Computing Environment

Transaction processing architectural [3] design moved from the centralized processing system to client/server processing system that allows the deployment of components of transaction processing systems to be organised in two ways. One way is by dividing the processing system into two operational tiers as shown in figure 2. The other way is by dividing the processing system into three operational tiers as shown in figures 3 and 4.

The functionality of the Two-Tier transaction processing model in figure 2 shows that mobile users' devices (laptop and mobile phone) located in Tier 1 hold programs that manage request and reports initiated by the users. The Presentation Services, Application Services (That is, front-end processes and back-end processes) and their management mechanisms form the Client System. In this kind of arrangement, fewer number of communication links are used to establish interaction between the presentation module (indicated as Presentation Service) and application module (indicated as Application Server) of the Client System. This is because the presentation module and application module are located in the same place. The Client System prepares users requests for execution by the Server System. The Server System consisting of the Database management System (DBMS) and the Database System is located in a different computing system indicated as Tier 2 in figure 2. Communication between Tier 1 and Tier 2 is done via wireless communication network. Programs responsible for Application services in Tier 1, communicate with the Databases via Database Management Systems (DBMSs).

Specifically, the responsibilities of the Application Server are [4]:

i. Set transaction boundaries.
ii. Implement user request as a sequence of tasks (that is, doing the functions of a controller).
iii. Act as a router as it affects management of distributed transactions and load balancing.
iv. Manage clients' requests by applying multi-threading skills.

The Two-Tier architecture allows stored procedure interface to be created at the client's location while the stored procedure is stored and maintained at the server location reducing effects of mobile unit unreliability [3]. Also, the use of SQL statements to communicate with the server can be avoided [4], stored procedures have better protection being located at the server machine, network traffic is also reduced thereby improving response time, authorization can be implemented in the procedure, procedures can even be created in advance. To enhancement the performance of the client system, the Application Sever is removed from the mobile users' computing devices and located in a separate computing device configured to be shared by multiple users as shown in figure 3 and figure 4.

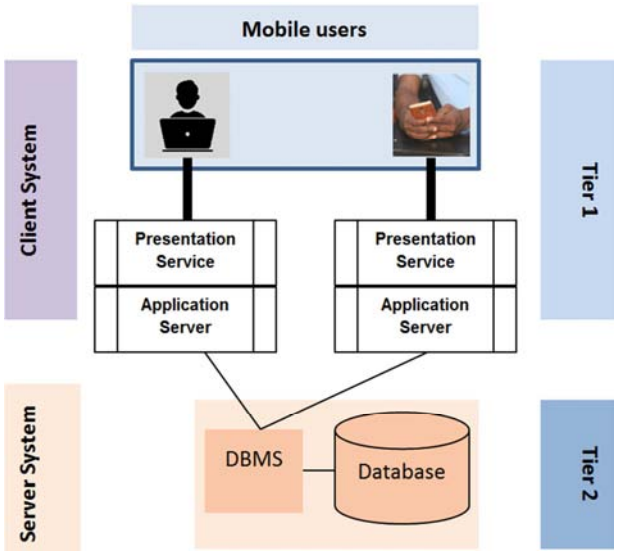

Figure 2. *Schematic diagram of a Two-Tier transaction processing architecture for ubiquitous computing syste.*

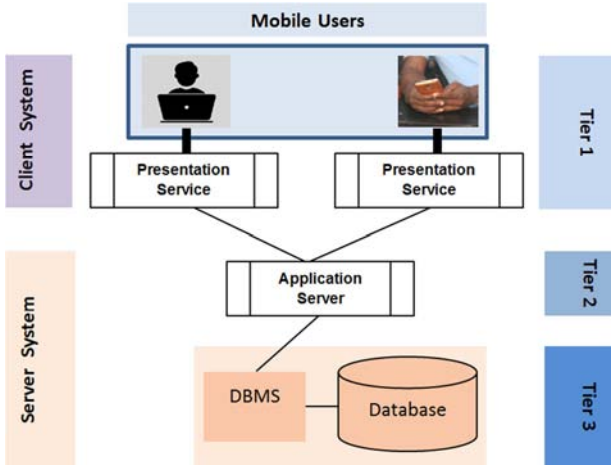

Figure 3. *Schematic of a 3-tier transaction processing system showing the removal of the Application Services from the Mobile Users' devices and putting it in a separate computing device located in a shared middle tier tagged Tier 2.*

Figure 3 shows that the separated Application server machine forms the middle tier tagged Tier 2. In so doing, the client machine only holds the user interface and the presentation services. In this way, different client machines can communicate with the database server through the application server.

Figure 4 is an illustration of a three-tier schematic where the DBMS is relieved of matters concerning stored programs. In this architecture, the stored procedures are removed from the Database Server to a separate server known as Transaction Server. The Transaction Server is directly connected to the Database Server. The sole function of the Transaction Server is to manage transaction segments. The application server uses the transaction server to execute stored procedures [3].

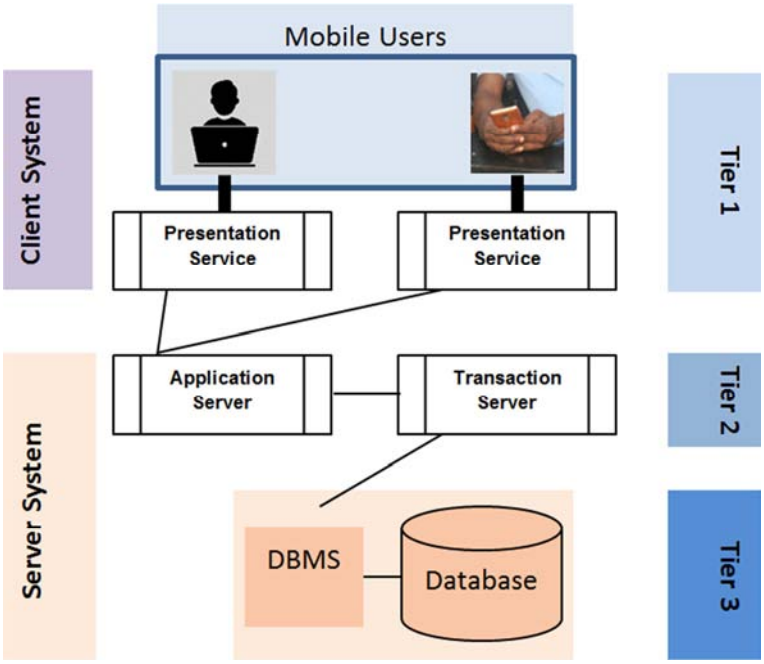

Figure 4. *Schematic of a 3-tier transaction processing system showing the application server and transaction server forming the middle tier tagged Tier 2.*

2.2. Data Availability Support for Ubiquitous Computing

Making data available for mobile users in ubiquitous computing environments can be challenging due to the following factors:

i. Different users with different computing devices, from different location can access different databases located in different distributed database servers and
ii. Location data can change at runtime.

For these reasons, a transaction manager must be equipped with the skills of processing different transactions (in what looks like being done simultaneously) optimally by making sure that:

i. Access is always granted to hybrid of applications' request to a database
ii. When more than one user access same content of a database, there should be no conflict
iii. A user should not be aware of other users of the same database
iv. Context awareness is maintained all through the processing stages of transactions

Data integrity must be sustained.

Mobile users can be found in different locations and places including meetings, on the sea, in buses, on the road, and in their homes as implied in figure 1. When ubiquitous computing environments are faced with uncontrollable system and connectivity related issues, they are reduced to traditional mobile computing environments. For example, if any of the links in figure 1 experiences issues resulting from outright disconnection or intermittent connection or system failure, it will be difficult to achieve ubiquitous computing. The two processing units (Processing unit 1 and 2) in figure 1 support different mobile units and users. For example, a mobile user in a bus at a remote location can control or access ubiquitous home appliance such as a refrigerator from the bus. Let us assume that the mobile user wants to have an idea of the stock level of a particular item in his or her refrigerator before arriving home from work. The reason could be that so the user could stop by a shop to get some more quantities of the item that is of limited quantity. Accessing the refrigerator and getting the required information from the refrigerator while in a moving bus distance away from home is done at ease and timely in a ubiquitous computing environment supported by appropriate communication architecture, communication protocol and a mature wireless infrastructure. In a situation where there is more than one type of refrigerator in the apartment, choosing the required refrigerator and the right location for the item are also done at ease with the implementation of appropriate database technology.

Generally, poor hardware (input, processing, output, and telecommunication devices) and software (processing instructions for transaction management, application management, recovery management, and database management systems) infrastructural designs are basic issues militating against the successful implementation of ubiquitous computing environment. This phenomenon is unacceptable as the whole idea of invisible technology and visible impact [7] associated with ubiquitous computing is greatly threatened.

2.3. Analysis of Transaction Commit Protocol Implementation on a Three-Tier Transaction Processing System Architecture

Information processing in ubiquitous computing environment is difficult [8] due to the inherent complex nature of distributed systems found in the environment. In this section, the Mobile 2-Phase Commit and proposed Mobile 3-Phase Commit data communication Protocols are critically analysed.

The combinatorial effect of implementing each of these protocols on a 3-tier transaction processing system architecture is also analysed. This is to ascertain the most productive combination for transaction processing systems in ubiquitous computing environment. It is worth noting that all transaction processing system architectures have the same basic transaction processing elements [9] which are identified as, End-User Device, Front-End Program, Request Controller, Transaction Server, and Database System.

Communication of data from a client machine to a server machine is achieved via standard network infrastructure like the TCP/IP network infrastructure. The communication of transaction processes or operations (also referred to as messages) from the client machine to the server machine is conventionally done with the use of software structures known as Send and Receive pairs.

2.3.1. Message Communication Algorithm in Mobile 2-Phase Commit Protocol

Figures 5 is a diagram representing the first phase of the Mobile 2-Phase Protocol implemented on a 3-Tier Transaction Processing System Architecture whereas figure 6 shows the second phase of the protocol. The desire to shift from fixed processing nodes to mobile processing nodes motivated Nouali et. al. [2] to develop a Mobile-Two Phase Commit (M-2PC) protocol that extended the execution framework of the conventional 2PC protocol. In their work titled, "A Two-Phase Commit Protocol for Mobile Wireless Environment", the 2PC protocol principles are adopted but the fixed nodes are replaced with mobile clients and servers that communicate over wireless network infrastructure. Just like in 2 Phase Commit Protocol, in Mobile-2PC Protocol, the execution process of a transaction is divided into two phases. The Mobile-2PC protocol aims at providing an Atomic Commitment Protocol (ACP) with the specific objective of globally committing fragments of mobile transactions distributed to more than one processing node for execution. As shown in figures 5 and 6, a Mobile User connects to a Global Server Machine representing a Base Station (BS) or Mobile Service Station (MSS) [10]. A Base Station [10] is a computer augmented with a wireless interface to communicate with mobile devices and different Base Stations can be interconnected via wired links. Each Base Station covers a geographical area called a cell.

A Mobile User can directly communicate with a Mobile Service Station covering the geographical area in which the user resides. The Mobile User may move from one cell to another while transactions involving Distributed Database Systems are being executed. When this happens, a handoff process is required to keep the active process from failure. The Mobile-2PC model puts the handoff process in the hands of the Mobile Service Station. The first phase of transaction execution process as shown in figure 5 occurs between the 2nd and 3rd tiers of the system architecture. The readiness of the host of database management systems in the processing units located in the 3rd tier of the ubiquitous computing environment is confirmed for the commencement of the execution process. The actual execution and commitment of action transactions as shown in figure 6 form the 2nd phase of the execution process. Transmission of finished transaction to mobile users and release of resources also take place in this phase.

It is assumed that the ubiquitous computing environment under consideration deals with:

i. computing nodes that are stationary including the client machine
ii. embedded wireless network infrastructure
iii. cohorts (processing nodes) of similar characteristics that cannot be down at the same time but may operate at low and different bandwidth.
iv. Certain fixed servers equipped with public databases
v. Certain mobile devices equipped with personal databases.
vi. Base Stations have some processing capability such as interpreting mobile hosts and fixed hosts request.
vii. Mobile devices initiating transactions is initiated and.
viii. Insufficient computing resources and power supply.

Problems associated with the 2PC protocol that the Mobile-2PC protocol attempted to address include:

i. Mobile unit given the task of initiating and coordinating the execution of a transaction
ii. Mobile unit not having sufficient computing resources and power
iii. Mobile unit performing under low and variable bandwidths resulting in communication latency.
iv. Mobile unit having to deal with many Up-stream message exchanges over wireless network also resulting in communication latency.
v. Mobile unit having to host public and private databases posing high risk of data unavailable in the event of failures.
vi. The Mobile unit housing the coordinator making communication between the other participants and the coordinator unreliable in the event of the Mobile unit developing fault.

In minimizing the responsibility of mobile user's device, the coordinator (server) is removed from the Mobile user's device and located in the Mobile Service Station to which the Mobile user's device is attached. This allows free communication between the coordinator and other cohorts whether the mobile device is connected or not, and conserve computing resources in the mobile device.

Considering a situation where the mobile user's device (the transaction initiator) moves from a network coverage under a particular base station to another network coverage under a different base station, there is bound to be loss of communication between the mobile user's device and the coordinator attached to the mobile that mobile user's device if the coordinator does not move with the mobile user's device to the new base station (That is the MSS). The Mobile-2 Phase Commit Protocol addressed this issue in the following way: The Transaction initiator (Client) while under the coverage of a particular base station tagged Home-Base Station sends a commit-request to the coordinator in the

Home-Base Station. At this point, the Home-Base Station serves as the Current-Base Station. Since the base station holds the coordinator, the transaction can be executed partially or completely in the Home-Base Station.

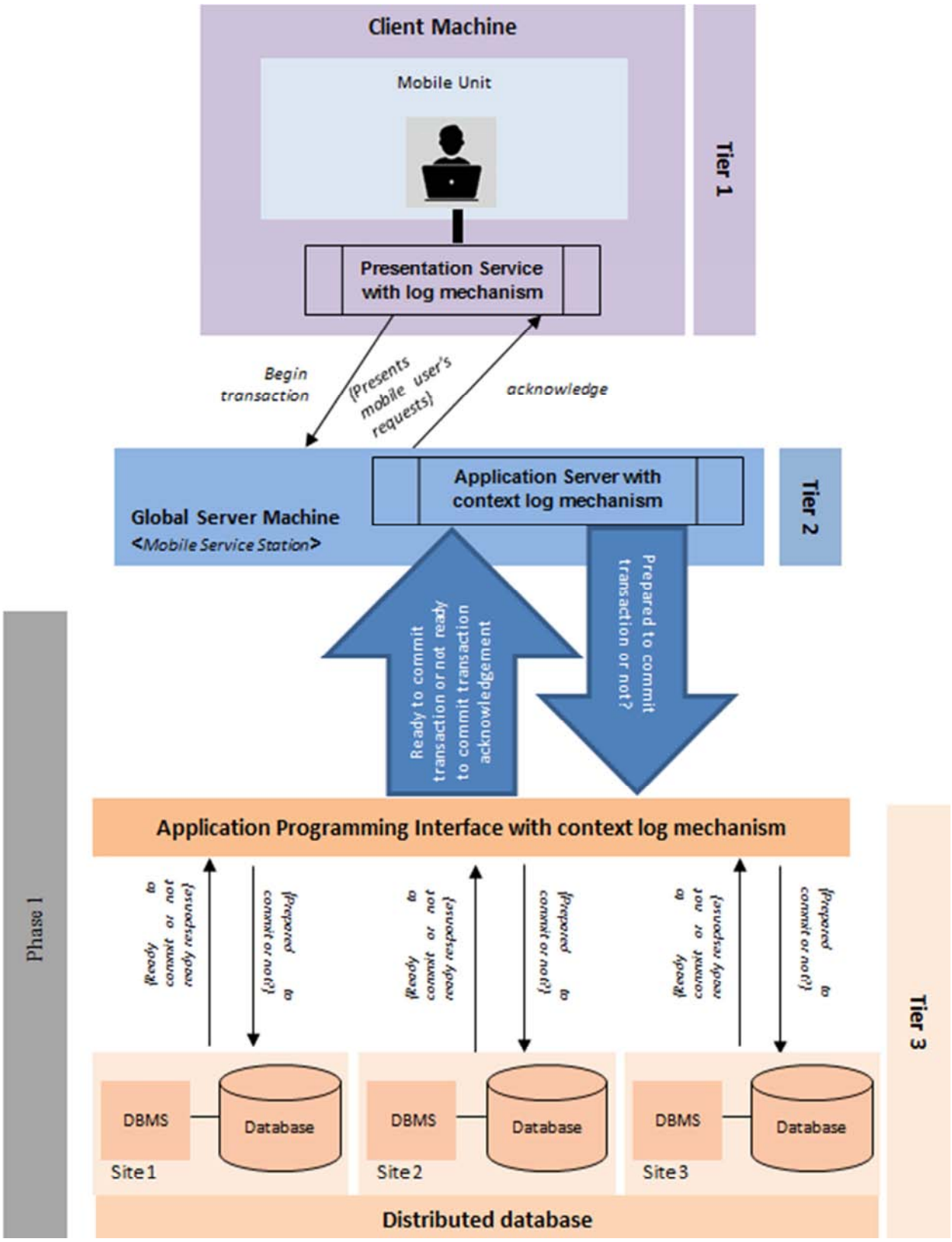

Figure 5. *A schematic representing the first phase of a Mobile 2-Phase Protocol implemented on a 3-tier transaction processing system architecture.*

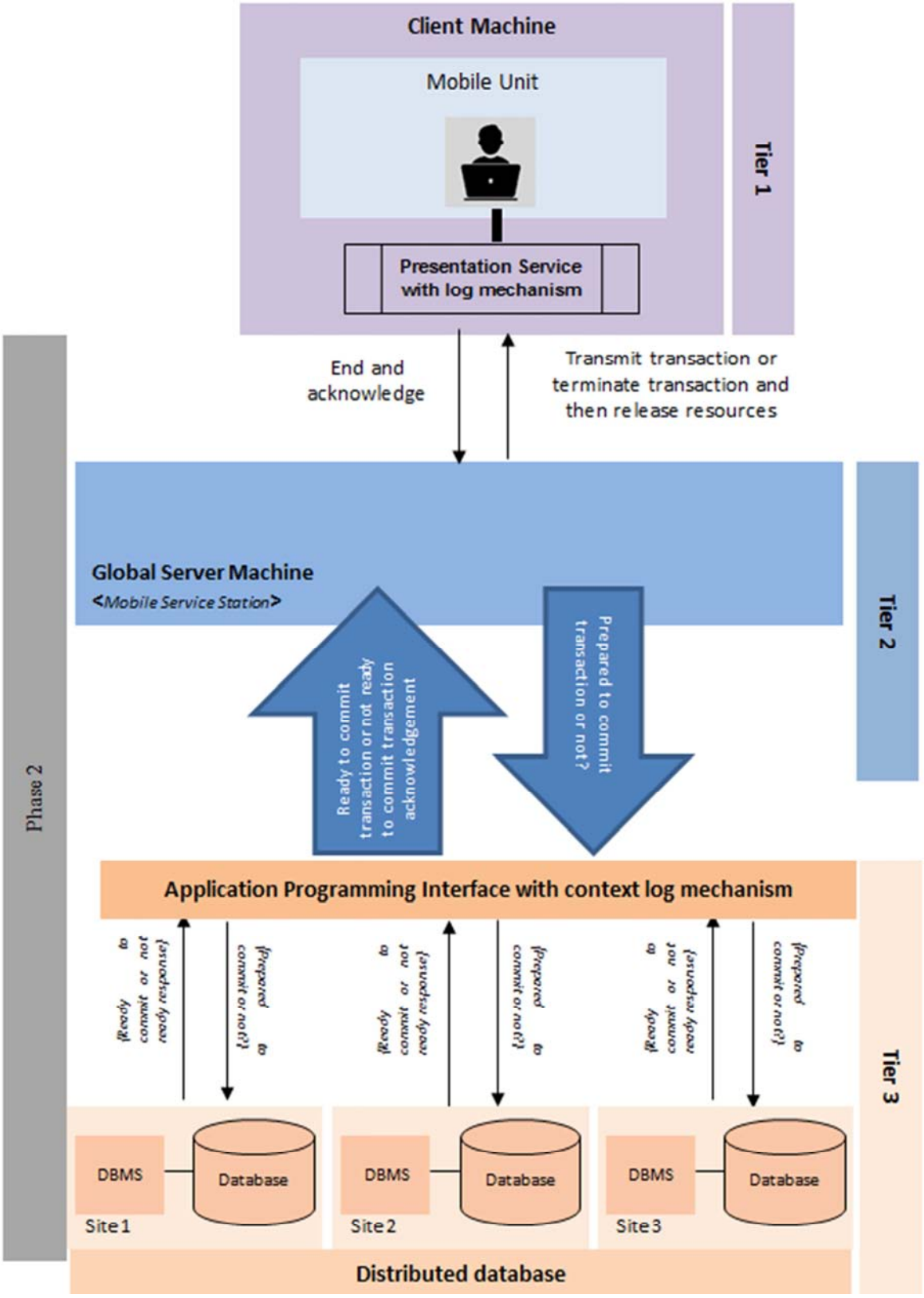

Figure 6. *A schematic representation of the second phase of a Mobile 2-Phase Protocol implemented on a 3-tier transaction processing system architecture.*

2.3.2. Proposed Combinatorial System Design for Transaction Processing System in Ubiquitous Computing Environment

Mobile-3 Phase Commit Protocol on a Three-Tier Transaction Processing System Architecture is identified as the proposed combinatorial system design for ubiquitous transaction processing system. Figures 7, 8, and 9 illustrate the stage-wise transaction execution process.

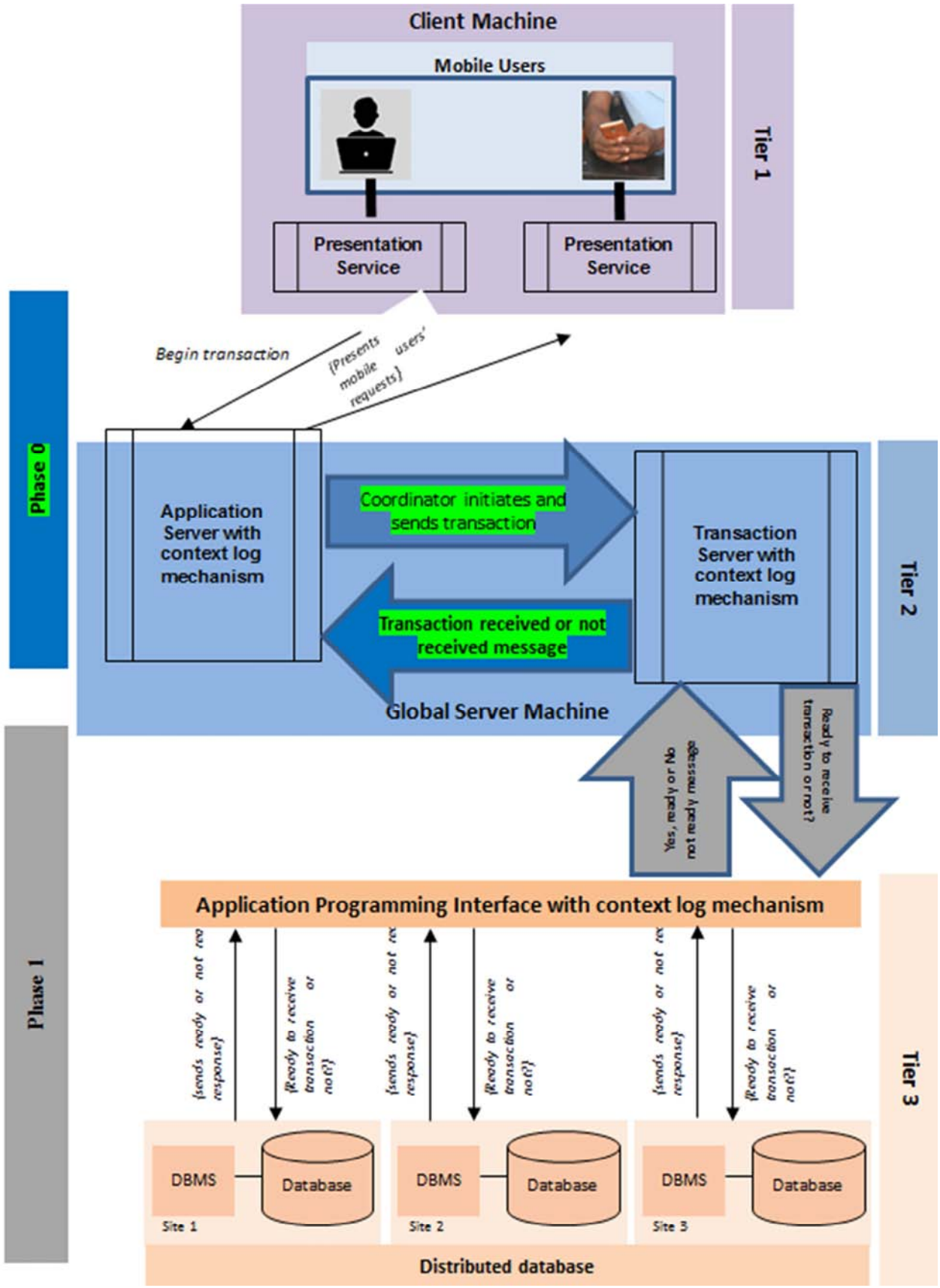

Figure 7. *Schematic of the first phase of the proposed Mobile 3-Phase commit protocol implemented on a 3-tier transaction processing system architecture.*

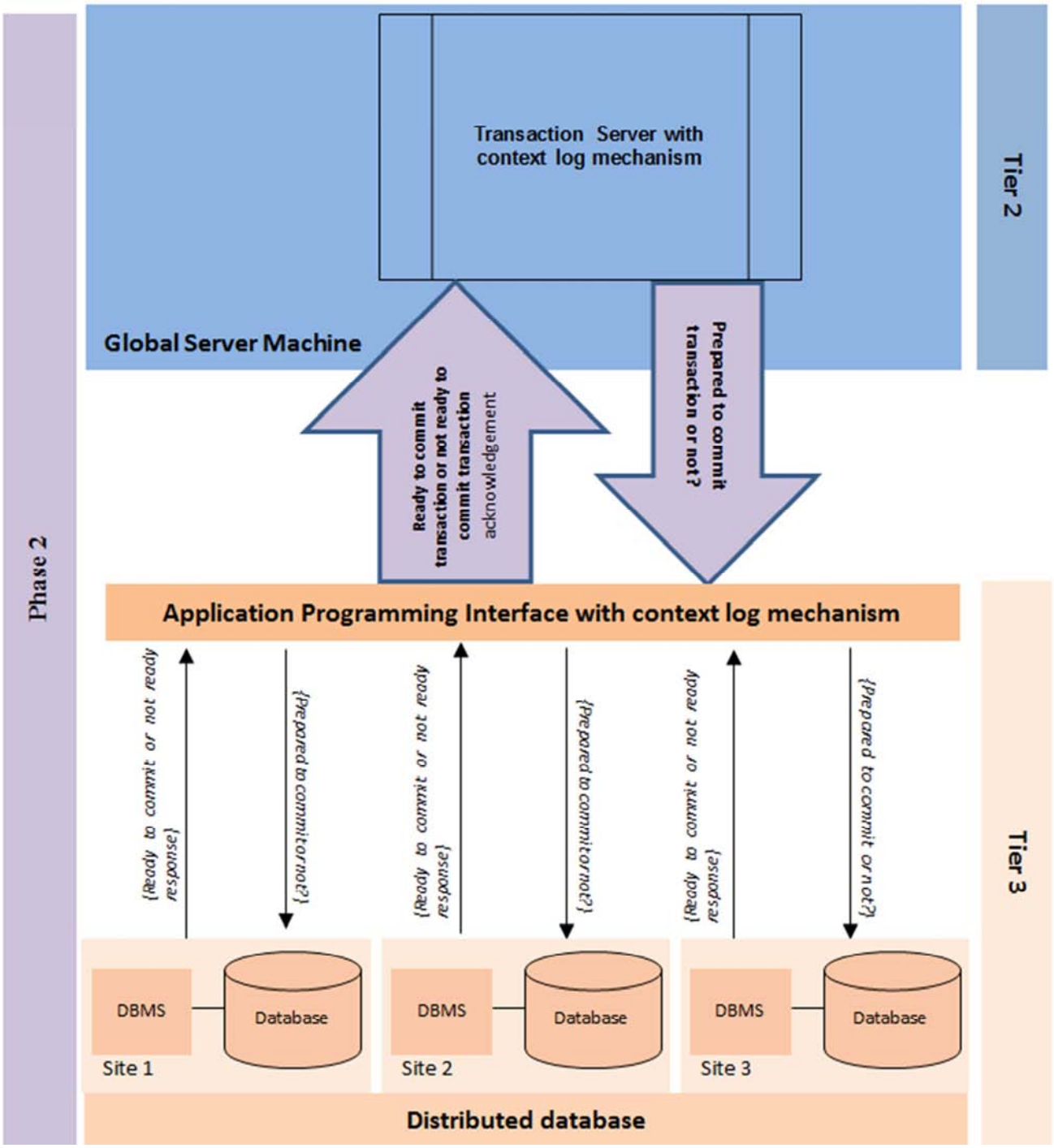

Figure 8. *A schematic diagram representing the second phase of the proposed.*

The proposed system design as presented in figures 7, 8, and 9 show that, mobile users connect to the application server with their mobile devices. The Application Server (AS) represents the coordinator of mobile units' applications, Transaction Server (TS) represents the coordinator of database processing units' applications, and Database Processing Units involved in the execution of transactions are referred to as cohorts (Participants).

The proposed system design attempts to solve the problem of increased message overhead and latency associated with the existing Mobile 2-Phase Commit Protocol. This is achieved by making the mobile unit a light weight processing device and the transaction servers proactive in managing database transactions.

The responsibility of the mobile device application is reduced to just request initiation and interfacing with the Application Server through an Agent-based User Preference Management System (UPMS) or a sensory framework. The Transaction Server is designed to be mobile during transaction execution. This means that it can access different data sources at run-time without losing connection to participating processing units.

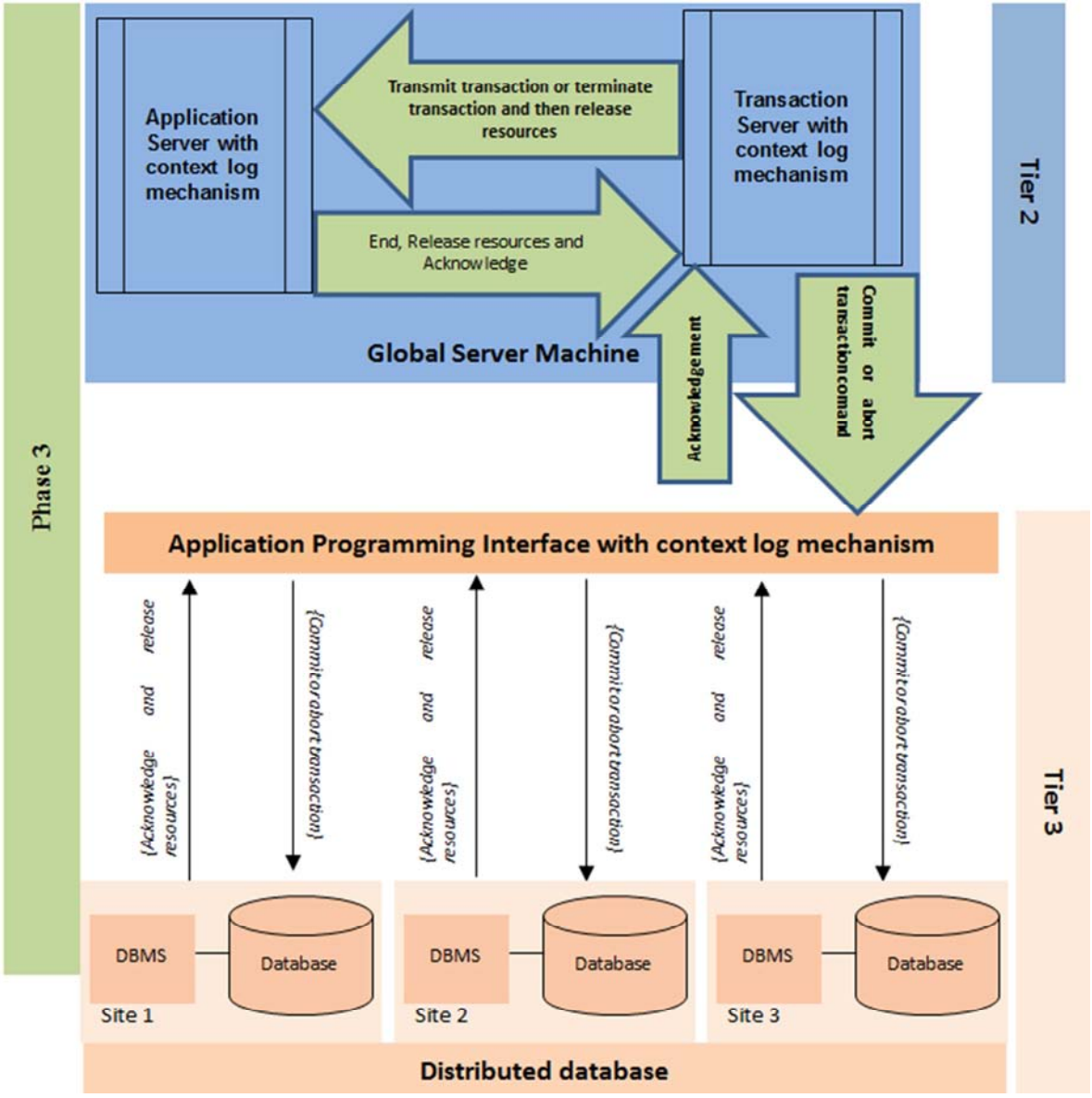

Figure 9. *A schematic diagram representing the third phase of the proposed Mobile 3-Phase Protocol implemented on a 3-tier transaction processing system architecture.*

The specific functions of the proposed Application Server and Transaction Server are listed thus:

Functions of the Application Server

i. Interface with n interface mobile agents Al within the ubiquitous network.
ii. Keep record of n user-end context information Uc collected from nAl.
iii. Update context information collected from nAl at runtime
iv. Interface with Transaction Server (TS)
v. Initiate n Transaction T (by activating the send message primitive)
vi. Feed Transaction Server with dynamic context information collected from n interface agents nAl at runtime
vii. Accept recovery request from transaction server
viii. Communicate recovery information to transaction server on demand
ix. Functions of the Transaction server
x. Interface with application server
xi. Accept messages (that is transactions) from application server
xii. Register transaction information into appropriate logs
xiii. Schedule transaction distribution and initiate send primitives (that is, start the execution of registered transactions)
xiv. Keep track of dynamic state parameters of transactions from the application server
xv. If any change in state parameter is perceived while execution is on, register the changes and update the transaction at runtime else just keep track of

transaction until commit state is reached

xvi. Interface with DBMS within the ubiquitous computing environment
xvii. Interface with backend context-aware mobile agents
xviii. Synchronize register of processing nodes in the ubiquitous environment with backend mobile agents
xix. Synchronize transaction details with backend agents
xx. Distribute sub-transactions to eligible nodes for processing
xxi. Keep track of nodes status and events until end of transaction
xxii. If an active node's standard is compromised register the state and then broadcast handover request to neighbouring nodes
xxiii. Wait for readiness to take over from neighbouring nodes
xxiv. Handover control to neighbouring node/nodes using the best judgment on receiving readiness message from neighbouring nodes.
xxv. Accept required data/instructions retrieved from Databases
xxvi. Wait for commit message from all participating processing nodes
xxvii. On receiving commit message from all participating processing nodes, converge processed sub-transactions from different processing nodes
xxviii. Save processed message and then communicate processed message to application server.
xxix. Get acknowledgement message from application
xxx. Commit transaction and then release held down resources

3. Result and Discussion

Figure 7 shows that the Application server interfaces between the client machine and the Transaction Server. This design takes care of issues associated with the mobile device being host to both the client and the coordinator. The 3-Phase Commit Protocol also takes care of the problem of the mobile host (That is, the mobile device) having the responsibility of announcing its position to a new base station. In the proposed model, that responsibility is handed over to the Application Server which is part of the global machine. It uses available wireless network infrastructure to connect to the transaction server in order to initiate transactions and other related operations. This phase is represented as "phase 0" in figure 7.

In phase 1 of the communication algorithm as shown in figure 7, the transaction server (specifically the operations coordinator) sends a "request to send" message to all eligible processing units [6]. On getting a "Ready to Receive" or "Not Ready to Receive" message from the cohorts (That is, eligible processing units hosting the database management systems), the coordinator moves to phase 2 of the protocol as shown in figure 8 where it transfers the transaction logs plus the "prepare to commit" message to cohorts that are ready to receive transactions for execution.

On getting this message, cohorts are expected to start the execution process and respond to the coordinator via the "Prepared to Commit" message when they are done with the execution or "Not Prepared to Commit" message when they are not done with the execution. Due to the varying system and network parameters, it is impossible for all cohorts to finish execution at the same time. However, all participating cohorts are expected to complete execution of their transaction branch within a predetermined time of one second.

The coordinator monitors the execution process of all the cohorts via their respective local mobile agents. If the coordinator predicts any possible failure or unnecessary delay in any of the cohorts, the state details of the transaction is logged and prepared for rescheduling (that is, for migration to a suitable processing unit)

At the expiration of one second, if all cohorts respond with a "prepared to commit" message, the coordinator moves to phase 3 as shown in figure 9 and sends a "commit" message to cohorts. If any or all cohorts' response is "not prepared to commit", the coordinator moves to phase 3 and sends an "Abort" message to cohorts. Commit or Abort message triggers the release of tied down system resources in cohorts after which cohorts send "Acknowledgment" message plus transaction log (That is, parameters that defines the state of the transaction) to the coordinator. On getting this message, the coordinator sends a decision message plus transaction log to the application server. The application server on receiving this message stores the transaction log received from the transaction server and then sends an "acknowledgement" message to the transaction server. The transaction server on getting this acknowledgment releases all tied down resources and then terminates the transaction.

4. Conclusion

This study presents a combinatorial system design of transaction processing elements for ubiquitous computing. A systematic analytical approach is used in analysing the organisational structure of two-tier and three-tier system architectures and subsequently the data communication algorithm of, 2 Phase Commit and 3 Phase Commit Protocols. A Mobile 2-Phase Commit Protocol is deployed on a 3-tier system architecture. This is compared with a proposed Mobile 3-Phase Commit Protocol deployed on a 3-tier system architecture. The result of the study shows that Mobile-3 Phase Commit Protocol on Three-Tier system architecture displayed proactive management skill to curb process failures which signifies higher transaction throughput. The inherent load balancing capability of the three-tier system architecture also shows support for improved response time.

References

[1] Poslad, S. (2011). Ubiquitous computing: smart devices, environments and interactions. John Wiley & Sons.

[2] Nouali, Nadia, Anne Doucet, and Habiba Drias. "A two-phase commit protocol for mobile wireless environment." Proceedings of the 16th Australasian database conference-Volume 39. Australian Computer Society, Inc., 2005.

[3] Kifer, M., Bernstein, A., and Lewis, M. P. Database systems: an application-oriented approach: complete version. Pearson Addison-Wesley, 2006.

[4] Spencer, P., and Nwachukwu. E. O. Light-weight Client/server transaction processing architecture for ubiquitous computing. African Journal of Computing and ICT. 8 (4), 201-208.

[5] Silberschatz, A., Korth, H. F., and Sudarshan S. Database system Concepts. McGraw-Hill, New York, 2002.

[6] Spencer, P., and Nwachukwu E. O. (2016) Identification and Classification of Processing Unit Eligibility for Ubiquitous Computing Using Feature Selection Mechanism and Artificial Neural Network. International Journal of Wireless Communication and Mobile Computing. 4 (2), 18-24.

[7] Hooft, M. V. and Swan K. Ubiquitous Ccomputing in Education: Invisible technology, Visible Impact. London: Lawrence Erlbaum Associates, 2007.

[8] Puder, A., Römer, K., and Pilhofer, F. Distributed systems architecture: a middleware approach. Elsevier Inc, 2006.

[9] Bernstein, P. A., and Newcomer, E. Principles of transaction processing. Morgan Kaufmann, 2009.

[10] Kumar, V., Prabhu, N., Dunham, M. H., and Seydim, A. Y. (2002). Tcot-a timeout-based mobile transaction commitment protocol. IEEE Transactions on Computers, 51 (10), 1212-1218.