PERFORMANCE ANALYSIS OF DISTRIBUTED DATABASE RECOVERY PROTOCOLS

V. Constantinidis
Department of Computer Technology
Monash University

ABSTRACT

An early recovery protocol that was investigated at length and is currently implemented in most commercial Distributed Database Management System products is the two-phase commit protocol. However, there is one main drawback in using this, namely blocking, that is, operational sites having to wait for the recovery of a failed site in order to complete a transaction. The two-phase commit protocol continues to be implemented in commercial Distributed Database Management System products even though alternatives to it have been developed to alleviate the blocking drawback. The alternatives are the three-phase commit protocol and a vendor developed replication technology. A critical and systematic analysis of these recovery protocols was conducted to determine the justification for continuing to implement the two-phase commit protocol in current commercial Distributed Database Management System products. This paper will present the results of the critical analysis.

1. Introduction

The growth and geographical expansion of organisations in the mid sixties to the early seventies revealed many of the limitations of centralised database systems. Accessing information via telecommunication lines proved to be slow. Even worse, if a failure occurred all of the organisation's processing had to cease until the central database system was back on-line. Distributed database systems were developed to circumvent many of the problems of centralised database systems. Various definitions of a distributed database system exist in the literature. The definition presented in this paper is based on that given by Date (1987) who defines a distributed database system as consisting of: 'a collection of sites connected by a communications network in which each site is a database system in its own right. But the sites have agreed to cooperate so that a user at any site can access any data in the network exactly as if the data were all stored at the user's own site'.

An organisation's data is an important asset and in the event of failure data integrity and durability must be maintained by the database system's recovery mechanism. Recovery control deals with 'the problem of preserving transaction atomicity in the presence of failures' (Ceri and Pelagatti 1988). The issue of recovery control and preserving transaction atomicity in a centralised database system has been well researched. One of the reasons for investigating recovery control in *Distributed Database Systems* (**DDBSs**) is because an efficient and resilient protocol which preserves transaction atomicity has not yet been implemented in commercial Distributed Database Management System (**DDBMS**) products. For example, the two-phase commit (**2PC**) protocol has the problem of blocking which detracts from its performance in certain circumstances. For this reason there is a need to investigate alternatives to the 2PC protocol.

A detailed performance analysis of the 2PC protocol, the three-phase commit (**3PC**) protocol

and Sybase's Replication Server was conducted to establish the efficiency and the performance of each of the protocols and to provide suggestions for an optimal recovery protocol. The *efficiency of a protocol* was defined as 'the number of messages per transaction that are required to be exchanged in a distributed database system in different cases of time-out and failure'. The aim of this paper is to present the results of this critical analysis.

2. Atomic Commitment Protocols

Atomic commitment in a DDBS environment means that either all the operations of a distributed transactions will be committed or the distributed transaction will be aborted. The operations of a distributed transaction are referred to as sub-transactions or *agents.* Agents are sent to the sites in the distributed database system which are required to participate in the distributed transaction.

The 2PC and 3PC protocols are atomic commitment protocols and are described in this paper. Atomic commitment protocols are discussed and modelled in terms of *states,* and *transitions* between those states. A *state* does not involve any processing, but rather it could be described as representing 'some behaviour of the system that is observable and that lasts for some finite period of time' (Yourdon 1989). A *transition* can be described in terms of '*conditions* that cause a state of change and the *actions* that a system takes when it changes state' (Yourdon 1989). A *condition* is typically an interrupt or signal or the arrival of data, for example any reading of messages. An *action* is any processing that must be performed, for example any sending of messages.

3. The Two-phase Commit Protocol: A Blocking Protocol

The *standard centralised 2PC protocol* which has been implemented in a number of widely used commercial DDBMS products such as Sybase, Ingres and ORACLE will be described in this paper. The 2PC protocol is described as a *blocking* protocol. Following Skeen (1981), the definition of blocking is where '... operational sites sometimes wait on the recovery of a failed site. Locks must be held on the database while the transaction is blocked'. This results in a high rejection rate of transactions entering a blocked participant site which require the locked data. The problem of blocking will be highlighted in the various failure cases which are discussed in the following sections.

Figure 3.1 shows a state transition diagram of the 2PC protocol, which does not consider failures. The notation uses rectangular boxes to represent states, and arrows between the rectangular boxes to represent transitions. A final state is represented by a double-edged rectangular box. The conditions and actions are shown next to the arrows which connect two related states. Conditions, those events that cause a transition out of a state, for example any messages received, are shown above the horizontal line and actions, those actions that the system takes while in a transition and before entering into another state, for example any messages sent, are shown below the line. For simplicity, the diagram includes only one participant site. In reality, more than one participant site could be involved in the execution of an agent sent to it by the coordinator site.

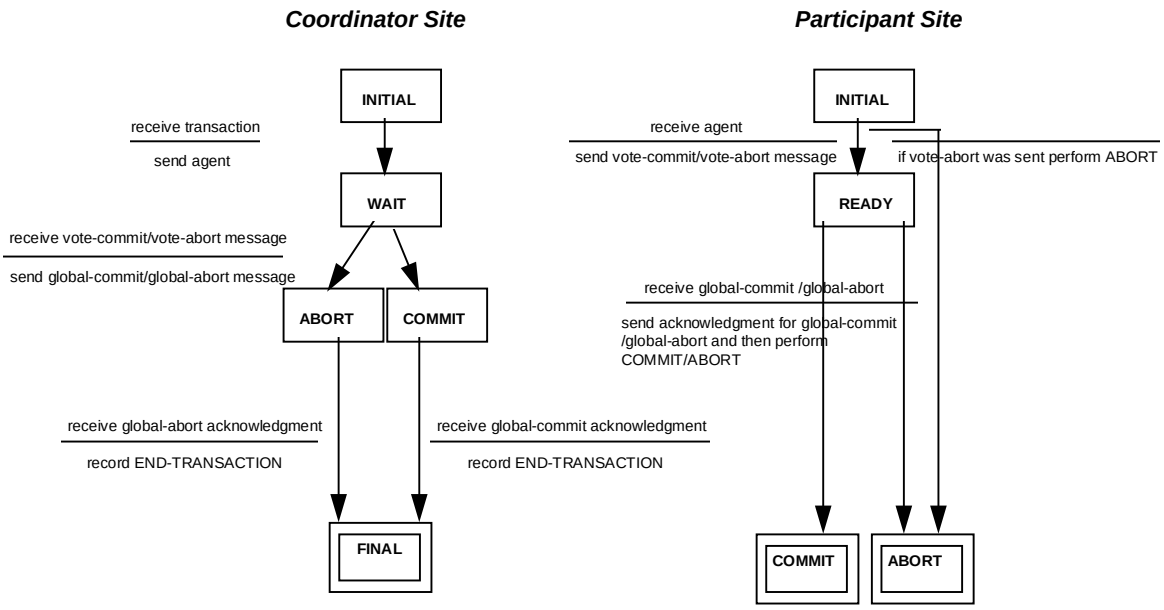

Figure 3.1 Two-phase Commit State Transition Diagram.

The following section gives a brief description of the *standard centralised 2PC protocol*, as described in Gray (1979), Skeen and Stonebraker (1983) and Ozsu and Valduriez (1991). The protocol is firstly described as it might occur in an ideal, error free, theoretical framework.

3.1 Transaction Execution Using the Two-phase Commit Protocol

The site of origin of a distributed transaction is the coordinator site. Other sites required to participate in the distributed transaction are called participant sites. The protocol functions can be divided into two phases. The function of the first phase is for the coordinator site to instruct each participant site to prepare the agent sent to it for commitment. Each participant site must then determine whether or not it can achieve this and indicate this to the coordinator site. The function of the second phase is for the coordinator site to collect the responses sent by the participant sites and then to base its decision to globally commit or globally abort the distributed transaction on the responses received. The coordinator site then has to inform each participant site of its global decision. It then waits for acknowledgment from each participant site. A detailed description of the operations in each phase is given in Constantinidis (1994).

3.2 Distributed Termination and Recovery Protocols for the Two-phase Commit Protocol

Termination protocols are also known as time-out protocols because they are invoked upon time-out. The termination protocols in the 2PC protocol are invoked when a destination site does not receive an expected message from a source site within an expected time period. To enable a destination site to terminate from such a state, timers are used in the 2PC protocol. Time-outs are usually caused by network or site failures. However, slow response times in the network can also cause a time-out.

Recovery protocols are used in the 2PC protocol by the coordinator site and participant sites to recover their states upon re-start from failure. Failures can occur when either the

coordinator site or each participant site is in a state or in a transition. Note that while in a transition a number of possible points of failure exist but all of these will not be identified in the following. Constantinidis (1994) provides a detailed description of the specific processing that needs to be performed for every possible point of failure.

Coordinator site Time-outs and Participant site Failures

The coordinator site can time-out in the WAIT state, COMMIT state, or ABORT state. A coordinator site time-out in WAIT state is caused by a participant site failure in INITIAL state or in transition out of INITIAL state. A coordinator site time-out in COMMIT or ABORT state is caused by a participant site failure in READY state or in transition out of READY state. In the latter time-out case the coordinator site waits for participant sites to acknowledge that they have either committed or aborted the agents that were sent to them. The coordinator site repeatedly sends either a *global-commit* or a *global-abort* message to the non-responding participant sites, and waits for their acknowledgment while remaining *blocked*.

Participant site Time-outs and Coordinator site Failures

A participant site can only time-out in its READY state. A participant site time-out in the READY state is caused by a coordinator site failure in transition out of INITIAL state, in WAIT state, or in transition out of WAIT state. Since it is in the READY state, the participant site must have sent a *vote-commit* message to the coordinator site and it therefore cannot change its decision at this stage and unilaterally abort. The participant site may also not decide to unilaterally commit since it is possible that another participant site may send a *vote-abort* message to the coordinator site. In this case the participant site will remain *blocked* and will not be able to release any locks on the database until it can learn from the coordinator site the ultimate decision of the transaction.

4. Non-blocking Alternatives

It is desirable for distributed termination protocols to be non-blocking because this will significantly lower the rejection rate on new transactions that enter at a blocked participant site and which require the same data that the participant site had been granted locks on. The following sections will describe two non-blocking alternatives:

(1) The 3PC protocol, and,

(2) A replication technology which has been developed by vendors of DDBMS products, as an alternative to the 2PC protocol. Sybase's Replication Server will be discussed in this paper as an example of the replication technology.

4.1 The Three-phase Commit Protocol: A Non-Blocking Protocol

The 3PC protocol unlike the 2PC protocol is a non-blocking protocol and is described in Skeen (1981), Skeen and Stonebraker (1983), and Ozsu and Valduriez (1991). It was developed using the *fundamental non-blocking theorem* as a basis. The fundamental non-blocking theorem states that 'when a site failure occurs, the operational sites must reach a consensus on committing the transaction by examining their local states' (Skeen 1981). This theorem has been used as a basis for the termination protocol in the 3PC protocol, which is discussed in section *4.1.2*. The 3PC protocol executes similarly to the 2PC protocol, except that there is one additional state, called the *PRE-COMMIT* state, at the coordinator site and participant sites which ensures the protocol to be non-blocking. The PRE-COMMIT state acts

as a buffer where the process is ready to commit but has not yet committed.

Figure 4.1 shows a state transition diagram of the 3PC protocol, which does not consider failures. The notation used is exactly the same as for the 2PC protocol. For simplicity, the state transition diagram includes only one participant site.

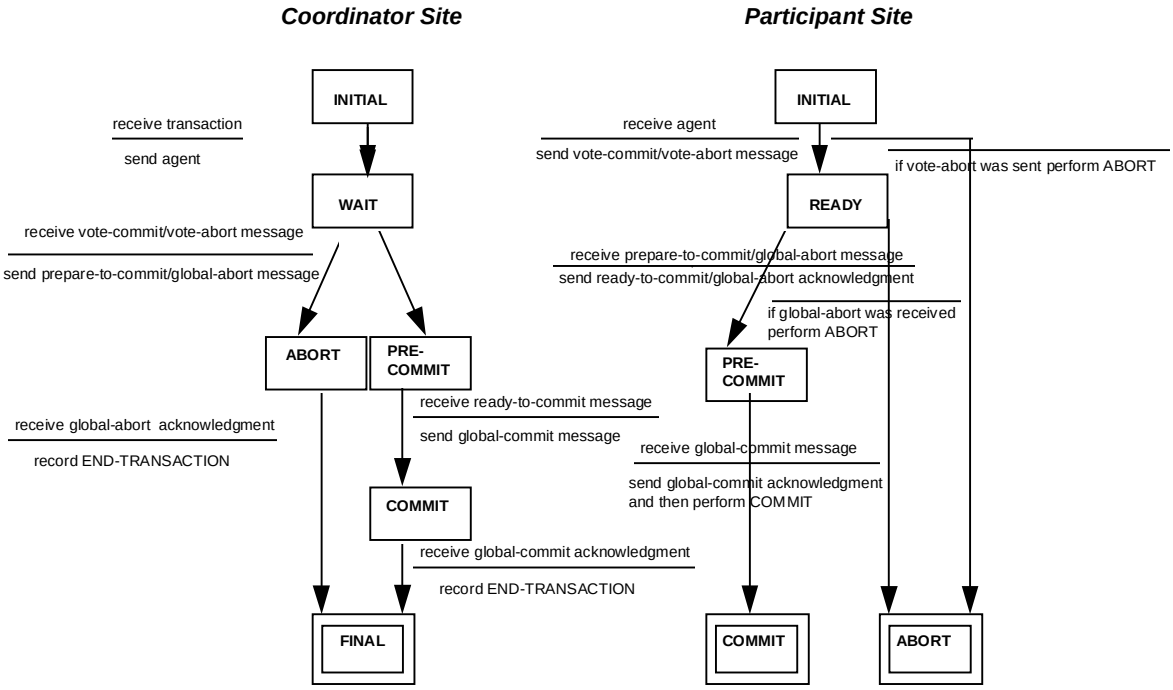

Figure 4.1 Three-phase Commit State Transition Diagram.

4.1.1 Transaction Execution Using the Three-phase Commit Protocol

The site of origin of a transaction in the 3PC protocol is the coordinator site. Other sites required to participate in the transaction are called participant sites. The protocol functions can be divided into three phases. The function of the first phase is for the coordinator site to instruct each participant site to prepare the agent sent to it for commitment. Each participant site must then determine whether or not it can achieve this and indicate this to the coordinator site. The function of the second phase is for the coordinator site to collect the responses sent by the participant sites and then base its decision to either prepare the participant sites for commitment or to globally abort the distributed transaction on the responses received. The coordinator site then has to inform each participant site of its decision. The third phase is executed only if it was decided by the coordinator site in the second phase to prepare the participant sites for commitment. The function of the third phase is for the coordinator site to collect responses from participant sites indicating that they are ready to commit. A *global-commit* message is then sent by the coordinator site to each participant site. It then waits for acknowledgment from each participant site. A detailed description of the operations in each phase is given in Constantinidis (1994).

4.1.2 Distributed Termination and Recovery Protocols for the Three-phase Commit Protocol

The *central site termination protocol* is available in the 3PC protocol to handle participant site time-outs caused by coordinator site failures. When a participant site times-out in either

the READY or PRE-COMMIT state, the central site termination protocol is invoked. This involves choosing another coordinator site from the remaining operational participant sites. The chosen coordinator site is referred to in Skeen (1982) and Skeen (1981) as a *backup coordinator site*. The backup coordinator site must direct the remaining participant sites to either commit or abort the distributed transaction. The methods used to elect a backup coordinator site will not be discussed in this paper. Garcia-Molina (1982) describes a number of election or voting algorithms that the sites can use to choose a new coordinator site. Hammer and Shipman (1980) describe an algorithm for choosing a new coordinator site which is based on a pre-assigned ranking. Constantinidis (1994) describes how the backup coordinator terminates the distributed transaction in the various time-out cases. The blocking problem is alleviated. However, there is an increase in the number of messages exchanged in the system where a failure of the coordinator site occurs.

Recovery protocols are used in the 3PC protocol by the coordinator site and participant sites to recover their states upon re-start from failure. The possible state and transition failures will be identified in the following sections. Constantinidis (1994) provides a detailed description of the specific processing that needs to be performed for every possible point of failure.

Coordinator site Time-outs and Participant site Failures

The coordinator site can time-out in the WAIT state, PRE-COMMIT state, COMMIT state, or ABORT state. A coordinator site time-out in WAIT state is caused by a participant site failure in INITIAL state or in transition out of INITIAL state. A coordinator site time-out in PRE-COMMIT or ABORT state is caused by a participant site failure in READY state or in transition out of READY state. A coordinator site time-out in the COMMIT state is caused by a participant site failure in PRE-COMMIT state or in transition out of PRE-COMMIT state.

Unlike the 2PC protocol, if a participant site does not respond before the time-out period lapses, then further messages will not be sent by the coordinator site, to the non-responding participant site(s) because the non-responding participant site(s) will, upon re-start from failure, rely on their recovery protocol.

Participant site Time-outs and Coordinator site Failures

A participant site can time-out in the READY state, or PRE-COMMIT state. The time-outs are caused by coordinator site failures. A participant site time-out in READY state is caused by the coordinator site failure in INITIAL state, in transition out of INITIAL state, in WAIT state or in transition out of WAIT state. A participant site time-out in PRE-COMMIT state is caused by the coordinator site failure in transition out of WAIT state, in PRE-COMMIT state or in transition out of PRE-COMMIT state. The central site termination protocol is invoked in both cases.

4.2 Replication Technology

A new technology is being implemented by most DDBMS or client-server vendors as an alternative to solely using the 2PC protocol. The vendors and their products include, Ingres' *ASK INGRES/Replicator,* Software AG's *Transaction Propagator* and Sybase's *Replication Server*. The technology used in the various vendor products is replication. Note that unlike the 2PC and 3PC protocols, the replication technology is not an atomic commitment protocol.

The replication technology does not require all sites to be operating, when a distributed transaction is being executed. However, the replication technology should not be used for distributed transactions which involve '... electronic funds transfer, which is highly time-sensitive ...' (Pallatto 1992). *Highly time-sensitive* distributed transactions are those distributed transactions that require to be atomically committed by the DDBMS and therefore the replication technology is inappropriate. However, '... Mike Schiff, director of data-management programs with Software AG in Reston, Va., said "... as much as 80 percent of the time users only want to distribute a subset of the master server data to a remote site" ...' (Pallatto 1992). Organisations have conducted surveys to determine the requirements in distributed database systems. For example Sybase Inc. (1992) found that: 'the appropriate business decision is most often to perform the updates at the sites that are available while deferring the update at the off-line site until it is again available. Total consistency across all sites is rarely required'.

The following section briefly describes Sybase's Replication Server only, because it was the only product commercially available at the time the critical analysis was conducted.

4.2.1 Sybase's Replication Server

The alternative answer that Sybase has additionally provided to overcome the blocking drawback and the inefficiency of the 2PC protocol is the Replication Server which is one of the five products in *SYBASE System 10*, a 'family of next-generation client/server products' (Sybase Inc. 1992). However, Sybase has not replaced the 2PC protocol, with the Replication Server. The 2PC protocol is still implemented in Sybase's Sequel Server product and is used by highly time-sensitive distributed transactions. When a distributed transaction enters the SYBASE System 10 system, the default is for the transaction to be executed automatically by the Replication Server. However, when a highly time-sensitive distributed transaction enters the system, the application program will recognise it as such and will execute it using the 2PC protocol.

How the Replication Server Operates

Sybase's Replication Server has a predefined location called the *primary copy* which contains all the data in a system. 'All updates are applied to the primary copy and then replicated' (Sybase Inc. 1993). One copy of the data is designated as the primary version. All other copies are replicas of the primary version. Sites that contain replicas, (referred to as 'replicate sites' (Sybase Inc. 1993)), are maintained by the data server at each of the replicate sites. A store-and-forward system handles the distribution of changes to copies of data reliably.

To execute distributed transactions, each of the replicate sites should have the appropriate logic included in application programs so that a replicate site will be able to generate agents. For both local and distributed transactions the relevant updates to the replicate site can be applied before the transaction is sent to the primary copy site. The replicate sites have two sets of data. One set is updated as soon as a transaction enters the site and before sending a copy of the transaction to the primary copy site. This set of data is called the 'Local Data' (Sybase Inc. 1993). Once the transaction is received by the primary copy site, the primary data server processes the transaction and then copies of the changes are sent to the appropriate replicate sites, including the replicate site that originally sent the transaction to the primary site. When the replicate site that had sent the transaction receives the update from the primary site, the update is made to the second set of data called 'Replicated Procedures

and Data' (Sybase Inc. 1993). 'An offsetting transaction is made to the local database to indicate that the transaction has been committed to the primary data and copied to the replicate database' (Sybase 1993).

Failures of Sites in the Network

There is no concept of time-outs when using the replication technology because the primary site in the Replication Server does not wait on any other site before it can execute a distributed transaction that is not highly time-sensitive. Therefore blocking cannot occur in this technology. Note that, because Replication Server is not an atomic commitment protocol there is no concept of states and transitions. The store-and-forward system handles the distribution of changes to copies of data reliably. This means that even if a site or a number of sites are not operational when a distributed transaction, which is not highly time-sensitive, enters the system, the transaction is queued for later delivery at the primary copy site.

5. A Comparative Analysis of the Two-phase Commit Protocol and its Alternatives

The results of the critical analysis will be presented in this section. Both the efficiency, in terms of the number of messages per transaction that are required to be exchanged in the system, and the performance of the protocols will be considered. This will be followed by a comparison of the message traffic in a failure-free environment and then for the various cases of time-out and failure for the protocols.

5.1 Critical Analysis of the Two-phase Commit Protocol

Constantinidis (1994) provides a detailed analysis of the number of messages required in each case of time-out. Both a formula for the total number of messages in the general case, is given as well as the "minimum" number of messages. The formula for the total number of messages considers the coordinator site and all the participant sites involved in the execution of the distributed transaction. The "minimum" number of messages, that is, the number of messages in the test case, is calculated between the coordinator site and *three* participant sites. To maintain consistency in the analysis three participant sites are included in the test case for the 2PC protocol because three participant sites are required to describe the different participant site time-out cases for the 3PC protocol. The following presents the results in the test case.

5.1.1 Coordinator site Time-out and Participant site Failures

Table 5.1 presents the number of messages required when the coordinator site times-out which is caused by a participant site failure. The worst case scenario is only included in the table. The number of messages required for every possible case of participant site failure is presented in detail in Constantinidis (1994).

Coordinator Time-out:	WAIT STATE		COMMIT or ABORT S TATE	
Point of Failure of Participant:	INITIAL	Trans_INIT	READY	Trans_RDY
Number of Messages:	12	12	13	13

Table 5.1 Messages Required for Coordinator site Time-outs for the 2PC Protocol.

5.1.2 Participant site Time-outs and Coordinator site Failures

Table 5.2 presents the number of messages required when a participant site times-out which is caused by failures at the coordinator site. The worst case scenario is only included in the table. The number of messages required for every possible case of coordinator site failure is presented in detail in Constantinidis (1994).

Participant Time-out:	READY STATE		
Point of Failure of Coordinator:	Trans_INIT	WAIT	Trans_WT
Number of Messages:	8	12	12

Table 5.2 Messages Required for Participant site Time-outs for the 2PC Protocol.

5.2 Critical Analysis of the Three-phase Commit Protocol

Constantinidis (1994) provides a detailed analysis of the number of messages required in each case of time-out. The same formula, test case and assumptions, described in section *5.1,* were used to determine the number of messages for the 3PC protocol.

5.2.1 Coordinator site Time-outs and Participant site Failures

Table 5.3 presents the number of messages required when the coordinator site times-out which is caused by a participant site failure. The worst case scenario is only included in the table. The number of messages required for every possible case of participant site failure is presented in detail in Constantinidis (1994).

Coordinator Time-out:	WAIT STATE		PRE-COMMIT or ABORT STATE		COMMIT	
Point of Failure of Participant:	INITIAL	Trans_INIT	READY	Trans_RDY	PRE-CO	Trans_PC
Number of Messages:	9	11	17	19	19	17

Table 5.3 Messages Required for Coordinator site Time-outs for the 3PC Protocol.

5.2.2 Participant site Time-outs and Coordinator site Failures

Table 5.4 presents the number of messages required when a participant site times-out which is caused by failures at the coordinator site. The worst case scenario is only included in the table. The number of messages required for every possible case of coordinator site failure is presented in detail in Constantinidis (1994).

Participant Time-out:	READY STATE			PRE-COMMIT STATE	
Point of Failure of Coordinator:	Trans_INIT	WAIT	Trans_WT	PRE-CO	Trans_PC
Number of Messages:	8	8	17	17	19

Table 5.4 Messages Required for Participant site Time-outs for the 3PC Protocol.

5.3 Critical Analysis of Sybase's Replication Server

The number of messages exchanged in the system, when executing a distributed transaction that is not highly-time sensitive, using Sybase's Replication Server is the same whether or not a failure occurs in the system.

Total number of messages is derived by using the following formula:

Total = (1 transaction is sent by a replicate site to the primary site) +
(1 agent X total number of replicate sites that need to have their data updated by the primary site).

In the test case of three replicate sites this gives *a total of 4 messages, that is 1 transaction is sent by a replicate site to the primary site, and then 1 agent is sent to each of the three replicate sites by the primary site.*

5.4 Comparison of the Two-phase Commit and the Three-phase Commit Protocols

As can be determined from Figure 5.1 the 3PC protocol compared to the 2PC protocol needs to exchange six additional messages (i.e. thirty-three percent more messages) when a distributed transaction is globally committed in a failure-free environment.

A comparison of the number of messages exchanged in the system in various time-out and failure cases is illustrated in *Figures 5.2* and *5.3*. Note that where indicated on the graphs, an **N/A** means that there is no such state or transition available in the 2PC protocol.

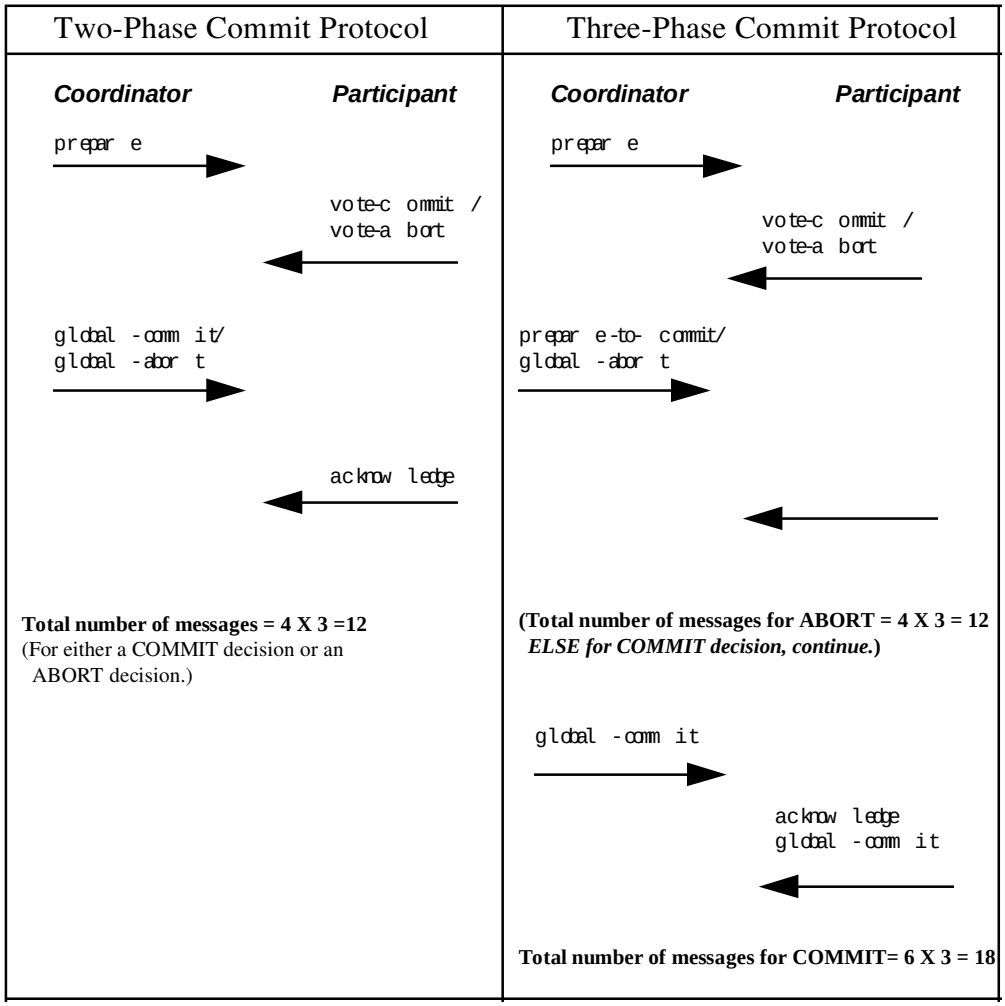

Figure 5.1 Messages Required for the 2PC and 3PC Protocols in a Failure-Free Environment.

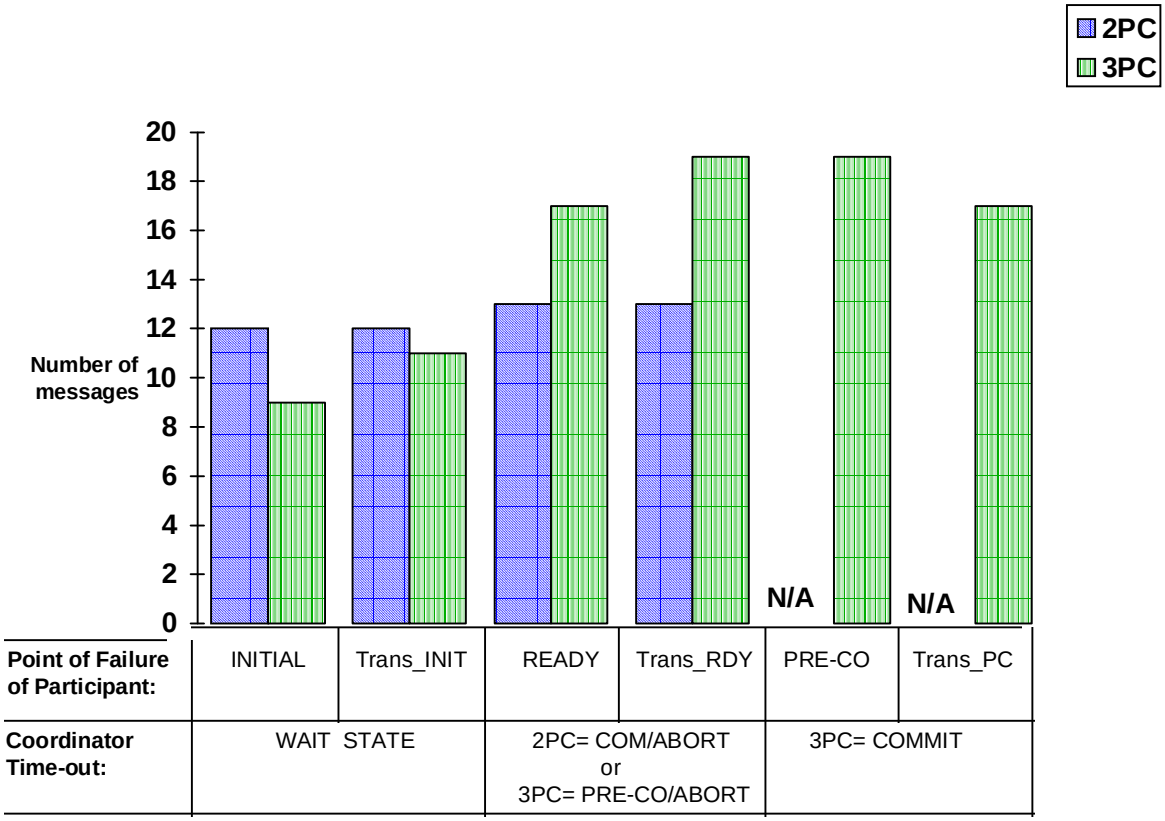

Figure 5.2 Messages Required for Coordinator Time-outs.

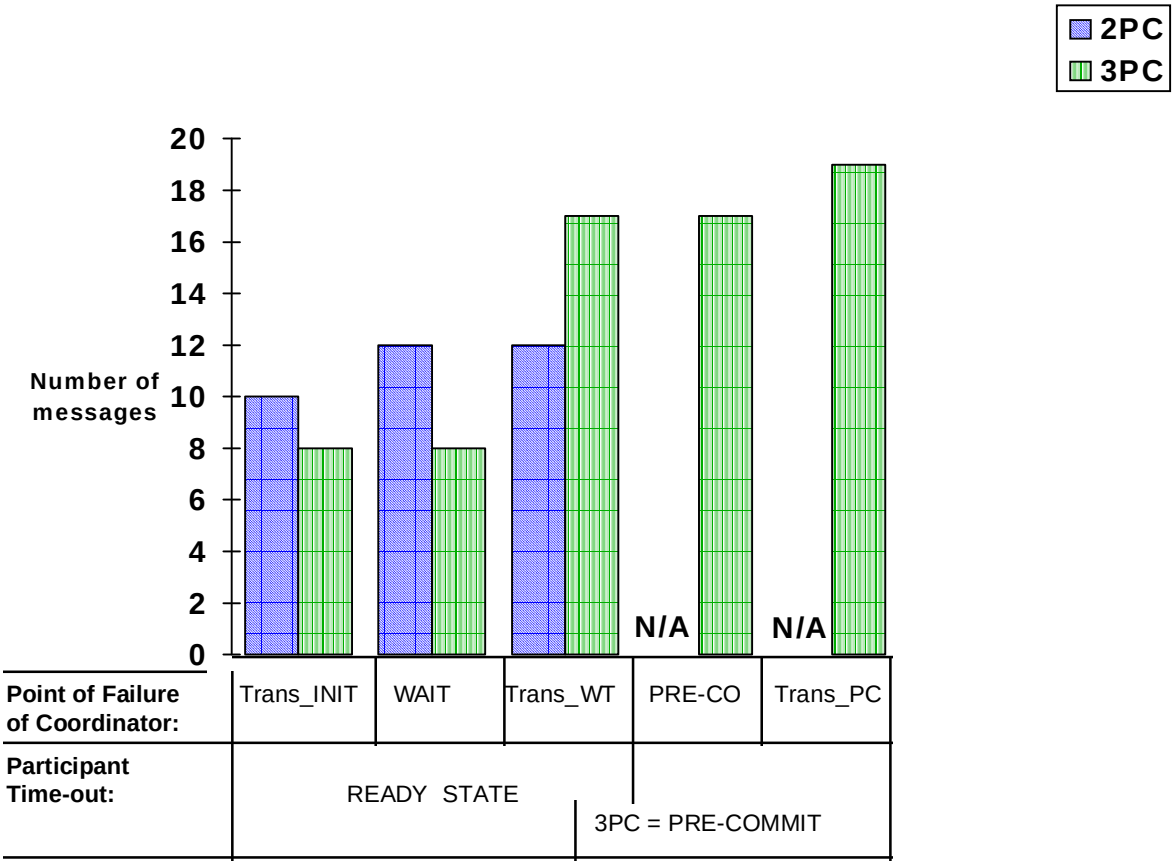

Figure 5.3 Messages Required for Participant Time-outs.

5.5 Comparison of the Two-phase Commit Protocol and Sybase's Replication Server

A comparison of the number of messages required for the 2PC protocol and Sybase's Replication Server in various time-out and failure cases cannot be achieved. Unlike the 2PC protocol, Sybase's Replication Server is not an atomic commitment protocol and therefore does not execute in terms of states and transitions. The number of messages required when using Sybase's Replication Server is minimal when compared to the 2PC protocol. Sybase's Replication Server significantly reduces the message traffic on the network. However, atomic commitment of a transaction cannot be performed. Therefore real-time synchronisation of the data in the distributed database system is not possible.

6. Conclusion

The 2PC protocol was found to be the more efficient atomic commitment protocol. However, it has the drawback of blocking which can result in a significantly high rate of rejection of the transactions entering a blocked participant site. The 3PC protocol fulfils its design objective of circumventing the blocking problem, but the number of messages increases during the termination of a distributed transaction when the coordinator site fails. Sybase's Replication Server also effectively alleviates the blocking problem and the number of messages required is minimal when compared to the 2PC protocol. However, Replication Server is not an atomic commitment protocol and is therefore inappropriate for use by highly time-sensitive transactions. Thus, Sybase has retained its Sequel Server product which implements the 2PC protocol and which runs alongside Replication Server.

Most vendors of DDBMS products currently implement the 2PC protocol and what can be deduced from this choice is that low message traffic takes precedence over a high rate of rejection of transactions entering a blocked participant site. However, because it has been determined in studies that distributed transactions do not require to be atomically committed eighty percent of the time, it can be conjectured that over the next five to ten years, DDBMS products will have a similar architecture to that of Sybase, where a replication technology will be used eighty percent of the time and the 2PC protocol will continue to be implemented for use only by highly time-sensitive distributed transactions.

REFERENCES

Ceri, S., Pelagatti, G. (1988). *Distributed Databases Principles & Systems.* New York: McGraw-Hill.

Constantinidis, V. (1994). *A Comparative Analysis of Protocols Used for Recovery Control in Distributed Relational Database Systems.* Master Thesis, Dep. Computer Technology, Monash University, Australia.

Date, C.J. (1987). '12 Rules for Distributed DB', *Computerworld,* (3 July), pp. 31-39.

Garcia-Molina, H. (1982). 'Elections in a Distributed Computing System', *IEEE Transactions on Computers,* Vol. C-31, no. 1, (January), pp. 48-59 .

Gray, J.N. (1979). 'Notes on Database Operating Systems', in *Operating Systems: An Advanced Course,* (eds.) R. Bayer, R.M. Graham, et al. Springer-Verlag. pp. 393-431.

Hammer, M., Shipman, D. (1980). 'Reliability Mechanisms for SDD-1: A System for Distributed Databases', *ACM Transactions on Database Systems,* Vol. 5, no. 4, (December), pp. 431-466.

Ozsu, M.T., Valduriez, P. (1991). *Principles of Distributed Database Systems.* Englewood Cliffs, N.J.: Prentice-Hall.

Pallatto, J. (1992). 'Adabas RDBMS to gain remote transaction tool. (Software AG introduces Transaction Propagator) (Product Announcement)', *PC Week,* Vol. 9, no. 48, (30 November), pp. 79-80.

Skeen, M.D., Stonebraker, M. (1983). 'A Formal Model of Crash Recovery in a Distributed System', *IEEE Transactions on Software Engineering,* Vol. SE-9, no. 3, (May), pp. 219-228.

Skeen, M.D., (1982). *Crash Recovery in a Distributed Database System.* Ph.D. Thesis, Dep. Elec. Eng. Comput. Sci., Univ. Calif., Berkeley.

Skeen, M.D., (1981). 'Nonblocking Commit Protocols', in *Proceedings of the ACM SIGMOD International Conference on Management of Data, April-May, 1981,* Ann Arbor, Michigan. pp. 133-142.

Sybase Inc. (1993). 'Replication Server: A Component of SYBASE System 10'. (April, 1993). Emeryville, Calif., Sybase Inc. 26 pages.

Sybase Inc. (1992). 'Enterprise Client/Server Architectural Issues and Options for Distributed Systems'. Emeryville, Calif., Sybase Inc. 13 pages.

Yourdon, E. (1989). *Modern Structured Analysis.* Englewood Cliffs, N.J.: Prentice-Hall.