Title

The Effect of Failure and Repair Distributions on Consistency Protocols for Replicated Data Objects

Authors

Carroll, JL
Long, DDE

Publication Date

1989

DOI

10.1109/simsym.1989.748300

The Effect of Failure and Repair Distributions on Consistency Protocols for Replicated Data Objects

John L. Carroll[1]
Computer Science Division
Department of Mathematical Sciences
San Diego State University

Darrell D.E. Long[2]
Computer and Information Sciences
University of California
Santa Cruz

Abstract

The accessibility of vital information can be enhanced by replicating the data on several sites, and employing a consistency control protocol to manage the copies.

Various protocols have been proposed to ensure that only current copies of the data can be accessed. The effect these protocols have on the accessibility of the replicated data is investigated by simulating the operation of the network and measuring the performance. Several strategies for replica maintenance are considered, and the benefits of each are analyzed. The details of the simulations are discussed. Measurements of the reliability and the availability of the replicated data are compared and contrasted.

The sensitivity of the Available Copy and Dynamic-linear Voting protocols to common patterns of site failures and repairs is studied in detail. Exponential, Erlang, uniform, and hyperexponential distributions are considered, and the effect the second moments have on the results is analyzed. The relative performance of competing protocols is shown to be only marginally affected by non-exponential distributions, validating the robustness of the exponential approximations.

1 Introduction

Distributed computing systems can provide improved fault tolerance for many applications by replicating data at several sites. The replicas must be managed by a protocol to ensure the consistency of the data object in the presence of site failures. The performance of common consistency protocols has been extensively evaluated [4,5,11–19], but all of these studies are predicated on simplified assumptions about the site characteristics. Analytic solutions to performance measures such as availability and reliability are intractable unless the system is a homogeneous network of sites conforming to standard Markovian conditions. Such studies do not apply to distributed systems comprised of heterogeneous sites, and discount many inescapable events such as periodic down times required for scheduled maintenance.

The behavior of Markov processes is *robust*, that is, Markov processes are good approximations to the behavior of more general stochastic processes. It is important to judge the extent to which the theoretical performance of replicated data on idealized systems agrees with the observed performance of actual systems. The infrequency of site failures in computer systems precludes the compilation of accurate estimates of the failure and repair distributions, and hence simulation is an appropriate vehicle for estimating the robustness of various protocols.

---

[1] Currently visiting faculty at the University of California, San Diego.

[2] Supported by faculty research funds from the University of California, Santa Cruz.

This article investigates the effects that non-exponential distributions have on the fault tolerance characteristics of several consistency protocols. Simulation models are developed for some common performance measures. These models can also be used to collect other performance data, such as the mean time to repair and mean time to failure of the replicated data object. By employing exponential distributions on identical sites, the simulation models can be validated against the results predicted by closed-form and numerical solutions [4,12].

Various protocols have been proposed to ensure consistent access to the copies of the data. The level of fault tolerance achieved depends on both the degree of redundancy and on the consistency protocol employed to manage the data object. Important measures of the utility of consistency protocols include *availability,* which is the steady-state probability that the object is available at any given moment, and *reliability,* which is the probability that a replicated object will remain continuously available over a given time period. Since availability is a measure of the steady-state behavior of the system, it can be directly estimated by calculating the stationary probability distribution determined by an appropriate state transition-rate diagram. However, such analytic solutions are intractable unless the sites are identical and the number of sites is small. Reliability is by definition a measure of transitory effects, and thus even for small numbers of sites, it is best approximated by simulation.

The simulation models are used to investigate the sensitivity of the performance measures to the higher moments of the failure and repair distributions. Exponential, Erlang, uniform, and hyperexponential distributions are used, and the effect the second moments have on the results is considered. The degree to which the relative performance of competing protocols is affected by non-exponential distributions is presented. Reliability is explored in §3, and §4 gives a similar analysis of availability. The availability and reliability results are also compared with those from a simulation based on site failure data collected from functioning networks over a period of eight months.

2 Previous Work

Replicating information on several sites of a network is a common method for increasing the availability and reliability of data. These attributes are most appropriately investigated by simulating the operation of the network. By employing consistency protocols, access to the replicated data can continue even in the event of failure of one or more of the participating sites. The replicas must be accessed in a controlled fashion to prevent the state of the data from becoming divergent.

Quorum consensus protocols such as Majority Consensus Voting ensure the consistency of replicated data objects by honoring read and write requests only when an appropriate quorum of the replicas of the data object can be accessed [8,22].

There are several methods for achieving this goal. The simplest is to assign a single vote to each site participating in the replication of the data object. When an access occurs, each operational site sends its vote to the access coordinator which determines if a quorum is present. This can be refined by assigning a variable number of votes to each site, depending on the site characteristics.

A weakness of all static quorum consensus protocols is that the quorum is fixed and does not change once the system has begun operation. This implies that the loss of one half of the votes will render the data inaccessible. Davčev and Burkhard [6] proposed a solution to this problem, known as *Dynamic Voting.* Their policy adjusts the quorum of replicas required for an access operation without manual intervention. A group of

replicas, comprised of a majority of the current replicas that can communicate among themselves, is referred to as the *majority partition.*

The basis of the original Davčev-Burkhard Dynamic Voting protocol is the *connection vector.* The connection vector instantaneously records the state of the system with respect to all sites. Each replica of a data object has an associated ensemble of state information consisting of the version number and the partition vector. The *version number* of a replica represents the number of successful write operations to the replica. The *partition vector* at a site records the version numbers of all sites with respect to that site. This protocol requires that system state information must propagate instantaneously to all sites in the computer network.

In its original form, Dynamic Voting allows accesses to proceed so long as a strict majority of the *current* replicas are accessible. In situations where the number of current replicas within a group of mutually communicating sites is equal to the number of current replicas not in communication, Dynamic Voting cannot proceed and the replicated data object becomes inaccessible. A simple extension proposed by Jajodia [10], known as *Dynamic-linear Voting,* enhances Dynamic Voting by resolving ties by applying a total ordering to the sites. The sites holding replicas of the data are given a static linear ordering. When a tie occurs, if the group of communicating sites contains exactly one-half the current replicas and that group contains the maximum element among the group of current replicas, that group is the majority partition.

The quorum requirement allows voting protocols to correctly manage a replicated data object even if a communication failure partitions the network connecting the sites. When network partitions and other *partial* communication failures cannot occur, *Available Copy* protocols provide higher data availabilities and reliabilities than voting protocols. This communications requirement is satisfied as long as all replicas of the data object are stored on the same carrier sense segment or on the same token ring, but precludes the usage of Available Copy protocols in many environments where sites holding replicas are separated by gateways.

The *Available Copy* protocol [2,3] is a set of policies for providing mutual consistency and concurrency control in a distributed database system.

The Available Copy protocol makes some very strong assumptions about the operating environment. The first assumption is that the communication network must be reliable. The communication network must not be susceptible to partitions, and reliable message delivery must be assured. A more controversial assumption is that failures are detectable in a fool-proof manner. In practice this is often a very difficult task. The method used is to send a message to a site and wait for a response or for it to time-out. This is unreliable in the case of heavily loaded sites.

Due to the strong assumptions that are made about the communication network, the Available Copy protocol guarantees that all available replicas of the data object are current. By assuming that the network will provide reliable delivery of each message, mutual consistency can be ensured by sending each write to every available copy. This also allows reads to be performed at any available copy, and in particular, at the local site if a copy is present.

When a site recovers from a failure it must obtain a current replica of the data object. If there is another operational site that holds a current replica, then the recovering site can request a replica of the data object from that site. In the event of a total system failure the Available Copy protocol must determine the last site to fail since that site must hold a current replica of the data object. The problem of determining the last site to fail has been studied by Skeen [21].

In the original proposal for the Available Copy protocol [2], the goal of detecting the last site to fail was achieved by maintaining several sets of failure information, including: all sites participating in the replication of the data object, those sites that have been specifically included in the set of replicas and those sites that have been specifically excluded from the set of replicas. An *included* site is one that is known to hold a replica of the current version of the data object, and an *excluded* site is one that has failed and the failure has been detected by some other operational site.

When a site s fails, another site t must detect that failure and execute the transaction `exclude(s)`. A failure detection mechanism is assumed to exist and it is assumed to be *fool-proof.* When a site t repairs following a failure, it will attempt to locate another site s that is in an available state. If such a site can be found, then site t will repair from site s and request site s to execute the transaction `include(t)`. In the presence of total system failure, the information maintained by the `include` and `exclude` transactions is used to determine the last site, or set of sites, to fail. The current version of the replicated data object will be found among those sites.

In order to perform an analytic study of replica control protocols, a simplified system model must be developed. The model consists of a set of sites with independent failure modes connected via a network which does not fail. When a site fails, a repair process is immediately initiated at that site. Should several sites fail, the repair process will be performed in parallel on those sites. For theoretical analysis, it is crucial that site failures are assumed to be exponentially distributed with mean λ, and repairs are assumed to be exponentially distributed with mean μ. The ratio of λ to μ is denoted by ρ; smaller values of ρ yield more reliable systems. For measuring availability, the system is assumed to exist in statistical equilibrium. The resulting system is characterized by a discrete-state Markov process [1,23]. These assumptions preclude network partitions, which can cause several sites to become simultaneously inaccessible, and clearly apply only if the participating sites have independent power sources.

Such stochastic process models have been used extensively to evaluate the fault tolerance provided by replica control protocols. However, there are many issues that cannot be addressed by stochastic process modeling. The assumption of an exponential distribution for site repair times is unrealistic, but using other distributions result in intractable analytic models. Discrete event simulation provides the ability to model replicated data objects in a more realistic manner. It allows for the relaxation of many of the assumptions that are necessary for an analytic model of the replicated data object.

3 Reliability Model

Reliability is a measure of the probability that a protocol will continuously allow access to the replicated data object over a given time interval. With sufficient iterations, a reliability simulator can build a comprehensive picture of the time-dependent characteristics of a protocol [13].

Definition 1. *The* reliability $\mathcal{R}_P(n,t)$ *of an n-site system managed by protocol P is defined as the probability that the system will operate correctly over a time interval of duration t given that all n units were operating correctly at time $t = 0$* [23].

The availability simulation model and the reliability simulation model conform to the same basic guidelines. However, reliability can be measured by simulating the system just to the point of the first access failure. Since the availability simulator must also model the recovery mechanisms, the reliability simulator is comparatively simple.

As in the availability simulation, each site is modelled as a SIMSCRIPT process which is initially designated as being in an operational state. This simulation was programmed in SIMSCRIPT II.5 and executed on an Elxsi 6400 computer. As time progresses, these processes alternate between operational and failed states according to the failure and repair distributions attributed to the corresponding site. When a site fails, its process is responsible for consulting the requirements of the protocols to determine whether the replicated object can still be accessed from other sites; the first access denial of each protocol is recorded. In this fashion, the behavior of several competing protocols for given failure and repair distributions can be analyzed in a single simulation. An iteration terminates after all the protocols would deny access to the data object, that is, when all sites are simultaneously in a failed state.

The reliability graphs were obtained by simulating the repairs and failures of a system of n sites until all sites failed, and noting the time at which the protocol would first deny access to a replicated data object. The process was repeated 1000 times, and the results were sorted to obtain an approximation of the reliability function.

The degree to which analytic models predict the behavior of actual systems is of great practical concern. To investigate this relationship, the failure and repair patterns of a network of nine sites was recorded for a period of eight months. These data form the basis for more realistic failure and repair distributions [12]. As shown in Figure 1, the behavior of the system based on this observed data is satisfactorily approximated by the exponential models. In each of the reliability graphs, ρ is fixed at 0.1, a typical value for modern systems. The horizontal time axis is measured in units which correspond to the average time to perform a single site repair.

Figure 1 also illustrates the performance differences of the major consistency control protocols. Available Copy clearly has vastly superior reliability in comparison to the other protocols. Indeed, it has been shown to have the highest reliability that can be achieved for a given number of sites [9]. Among the protocols that can ensure consistency in the presence of network partitions, Dynamic-linear Voting outperforms the other voting protocols.

Results are reported for $n = 3$ sites, but the conclusions presented here also hold for larger networks of sites. For three copies, the superiority of Dynamic Voting over Majority Consensus Voting is lost. The following exposition concentrates on Available Copy and Dynamic-linear Voting, the protocols of choice in non-partitionable and partitionable networks, respectively.

Four representative distributions were investigated in this and the following section. Exponential distributions were contrasted with Erlang-4, uniform, and hyperexponential distributions. The shape of each repair distribution was similar to the shape of the corresponding failure distribution; mixing the types of distributions within a single simulation did not enhance the analysis. For similar reasons, only homogeneous networks were considered: all sites were assigned identical characteristics. In each simulation presented in this section, the mean time to failure was always ten times the mean time to repair.

In exponential distributions, the mean and standard deviation are equal, yielding a "moment ratio" of 1. The standard deviation of an Erlang-4 distribution is just half the mean: 0.5 was the smallest moment ratio considered in this study. The uniform distribution varied from 0 to twice the mean, and has a moment ratio of about 0.58. The ratio of the first and second moments in the hyperexponential distribution is approximately 1.5, the largest in this study. This is achieved by blending two exponentials, one with a mean nine times larger than the other.

Figures 2 and 3 illustrate the simulated reliability of the Available Copy and Dynamic-

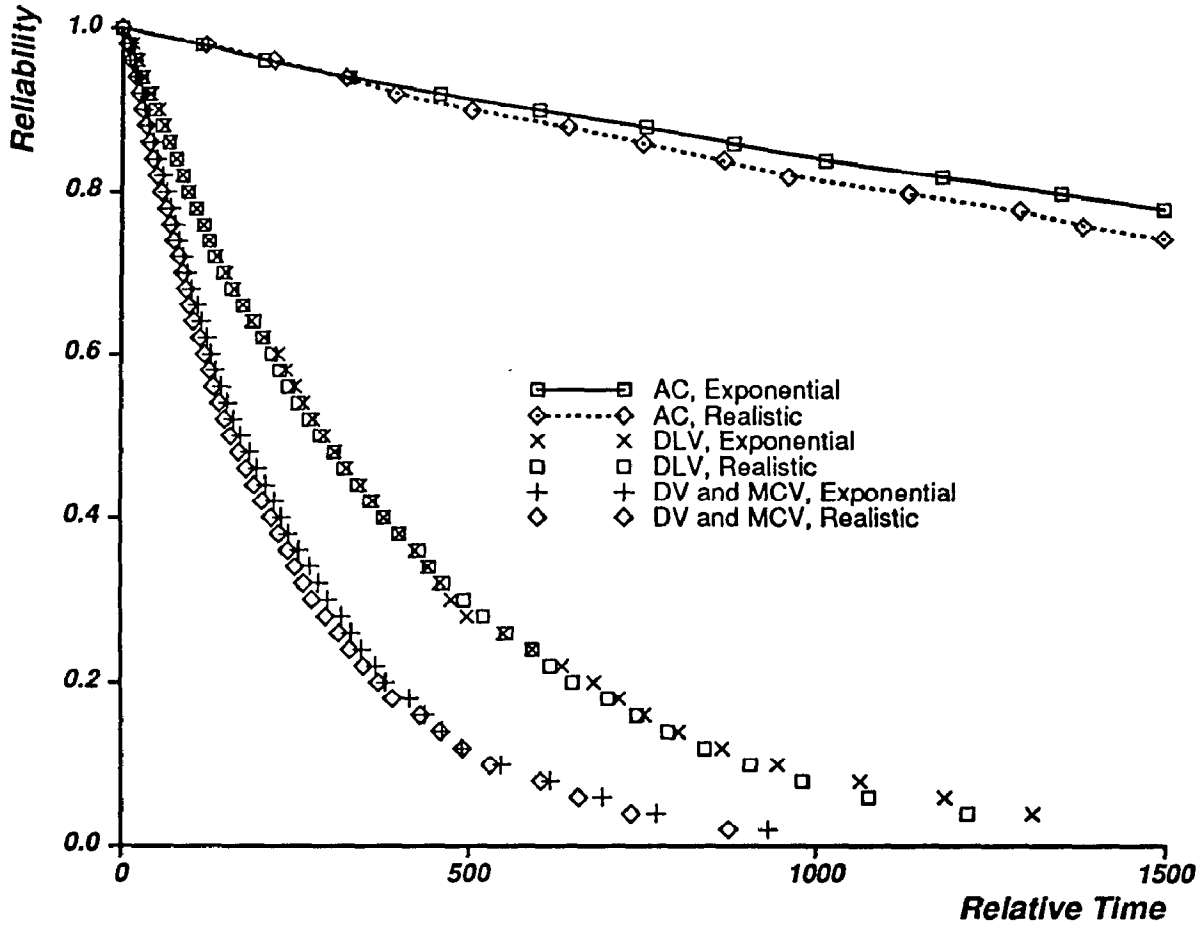

Figure 1: Reliabilities with three copies, Exponential versus Realistic Distributions

linear protocols, respectively. The four sample distributions had identical first moments but disparate higher moments. The differing times scales in the two figures reflect the much higher reliability of Available Copy.

The graphs show that the reliability of these systems is relatively insensitive to the shape of the distributions. As a rule, the uniform distribution has the poorest reliability, while the hyperexponential distribution has the best reliability. These are the distributions with the smallest and largest second moments, respectively.

For large networks, a larger moment ratio seems to ensures higher reliability, as illustrated in Figure 2. The correlation between the moment ratio and the reliability is somewhat weaker in the Dynamic-linear simulation, since it is possible to fail if only two of the three sites are down. In particular, the construction of the hyperexponential from two disparate exponentials implies that there will be both a larger number of simulation runs that experience quick failures, and a few instances in which the data object remains accessible for an unusually long period. The crossing of the hyperexponential curve in Figure 3 reflects this behavior. These effects are mitigated when a large number of sites are required for failure.

Each of the above experiments reflect a failure-to-repair ratio ρ of 0.1. It can be shown that a small increase in the reliability of the individual sites leads to an impressive increase in the reliability of the replicated object. However, the relative performance of the various protocols are unchanged, even when heterogeneous sites with varied distributions and mean failure and repair rates are simulated [4].

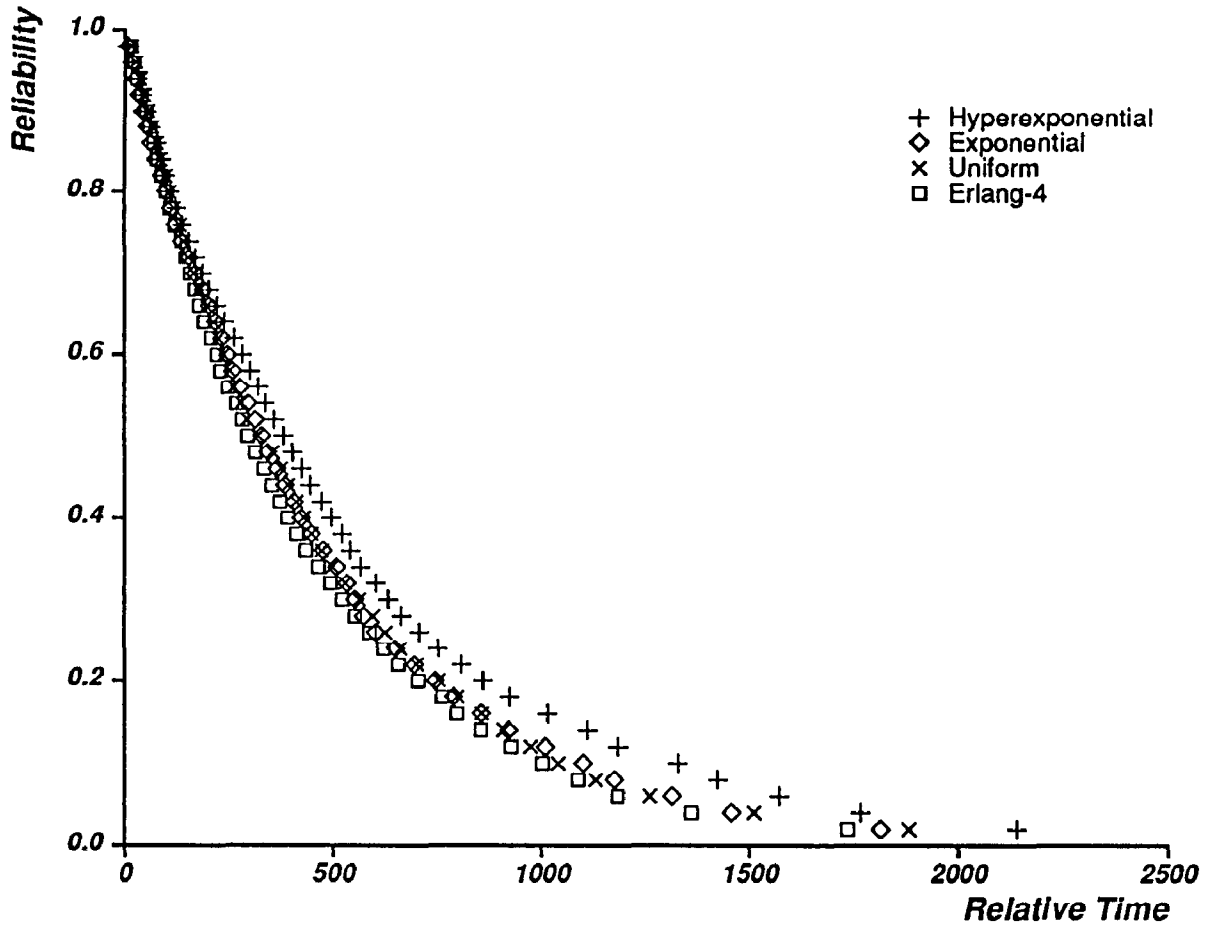

Figure 2: Reliability of three sites managed by Available Copy

4 Availability Model

In this section, the effect of various failure and repair distributions on the availability provided by the consistency protocols is considered. Availability is the steady-state probability that access to the data will be granted.

Definition 2. *The* availability *of a replicated data object consisting of n replicas and managed by a replica control protocol S, denoted $A_S(n)$, is the stationary probability of the system being in a state where the replica control protocol will grant access to the data object.*

The simulation to measure availability is necessarily more complex than the simulation for reliability since it must model the entire replica control protocol, including recovery from total system failure. This is an interesting contrast to analytic models, where the reliability model is the more complex.

A replicated data object was modeled as a set of replicas residing on sites distributed around the computer network. Sites not operating correctly were assumed to immediately stop operations and all messages were assumed to be delivered to their destinations without alterations in the order in which they were sent. Malicious failures were expressly excluded [20].

The simulation model was programmed in SIMSCRIPT II.5 and executed on a SUN-4/110 computer. Using the process interaction approach [7], a set of site processes model the behavior of the sites and a single client process model the access process.

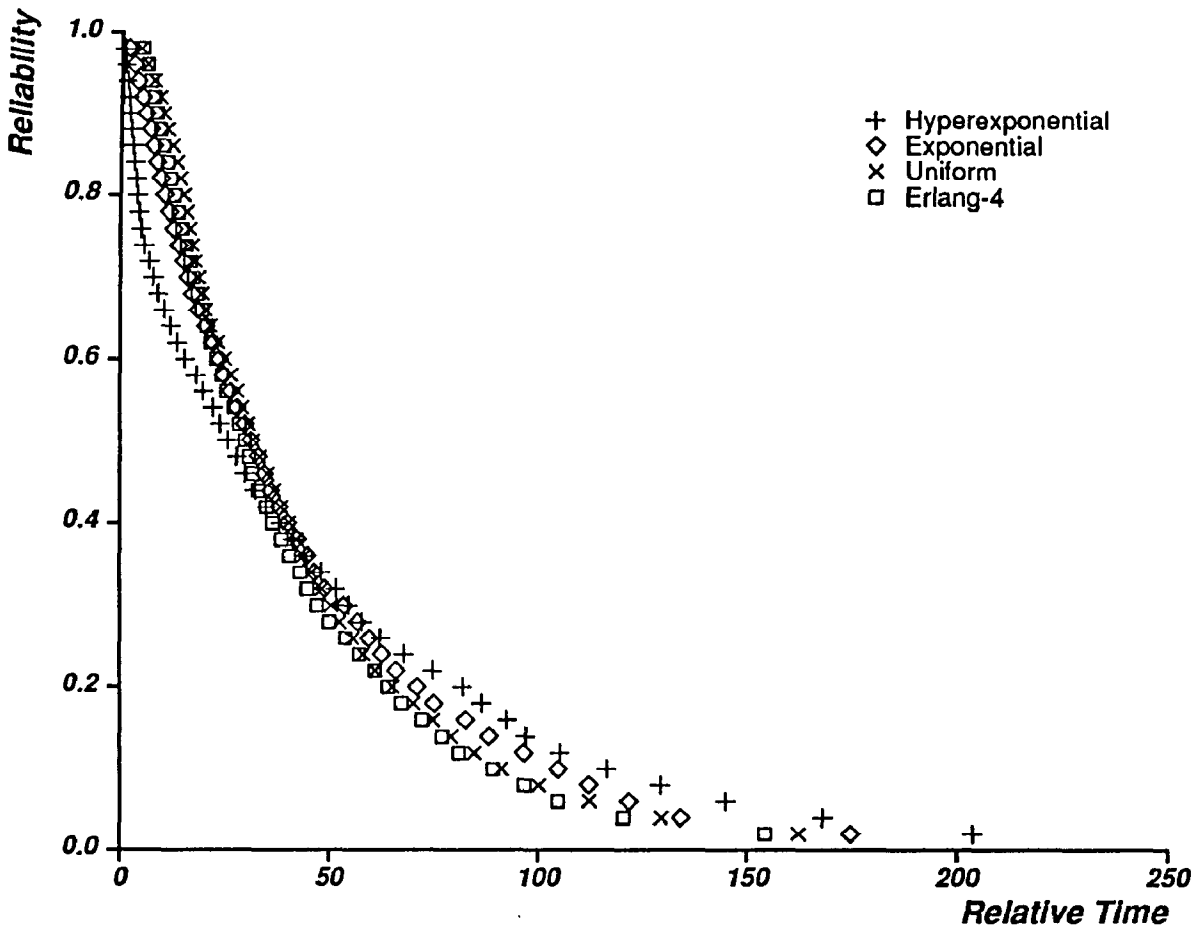

Figure 3: Reliability of three sites managed by Dynamic-linear Voting

Central to the simulation are the site processes, one per site, which models the behavior of a physical site. The site process remains accessible for a period of time determined by the failure distribution, and is then designated inaccessible. The repair distribution governs the period of time it remains in that state.

Once the site has been repaired, an access check is made to determine if the repair of this site has made the replicated data object available again. This facilitates the determination of the global availability of the data object. Similar requests are generated when a site fails.

The client process serves several purposes. Its primary function is to act as an accessing agent, allowing it to maintain the system configuration information required by the replica control protocols. This also allows it to monitor per-site availability. The monitoring of global system availability is distributed throughout the simulator, wherever the state of the replicated data object may change.

The access routines are implementations of the protocols described earlier. Both the client processes and the site processes cooperate in the implementation of various parts the replica control protocols.

Estimating the parameters for the simulation model in order to make it realistic was found to be a difficult task. The first attempt involved asking system administrators how often their machines failed. From this experience it was learned that system administrators are hopelessly optimistic, and that their usual response is "almost never."

System administrators are more helpful when it comes to determining the service

Table 1: Site Characteristics

Site	MTTF (hours)	MTTR U	MTTR E	Restart (minutes)
A	80.47	24.0	4.0	330.0
B	149.50	24.0	4.0	255.0
C	90.11	44.0	4.0	570.0
D	92.50	44.0	4.0	90.0
E	610.10	24.0	4.0	366.0
F	210.07	24.0	4.0	323.4
G	260.39	24.0	4.0	510.6

time to repair a component failure (MTTR). There are two distinct types of failure with different characteristics. From the data gathered, about 10% of these involve hardware and the remainder are software failures. The service contract for each machine specifies the period of time during which the service technician must arrive. By observing service technicians at work, a good estimate of how long it takes to diagnose the failure can be gained.

Some simplifying assumptions were necessary for the simulation model. The first is that the service technician will arrive within a known and bounded period of time. This is justified by the usual service contract clause that requires a service technician to arrive within a specified interval. The service technician is assumed to require a uniformly distributed period of time to diagnose the failure. Once the technician has diagnosed the problem, the failed component must be replaced. In some cases the component will not be available and will require the technician to retrieve it from the warehouse. It is reasonable to assume that the portion of the service time during which the machine is actively being repaired is exponentially distributed.

In order to estimate the mean times to failure (MTTF) and the time to restart machines following a failure, system monitoring processes were installed around the computer network. The Berkeley UNIX system provides a convenient method of collecting these statistics. Each site periodically sends a broadcast packet that lists the number of users, the load average and the time that the system has been operational. Since sites tend to be very reliable, it was impossible to gather enough data points in order to accurately determine the failure and repair distributions. The statistics gathering-processes were placed at several installations, including the University of California, San Diego and the Naval Ocean Systems Center.

Several items of interest were encountered during the course of this effort. The first is that systems fail much more often than system administrators would would lead the users to believe. This is not a conscious deceit on their part; it just seems that they do not regard "a quick reboot" as a system failure. Another is that significant periods of time can elapse between when a system fails and when it is restarted. The fluctuations in usage during different periods is a significant factor. The time to restart the system seems to be dominated more by the use of the machine than by the machine type. If a machine is in constant use, it is more likely to be rebooted quickly than one that is seldom used.

There are some limitations of the simulation model. It was infeasible to execute the simulations long enough to get accurate measures for some of the better protocols when realistic failure and repair rates were used. In particular, Available Copy with five

Table 2: Available Copy

Configuration Replicas	$\mathcal{A}$	$\mathcal{U}$	MTTF (Days)	MTTR (Days)
AB	0.994200	0.005781	45.21003	0.26394
ABC	0.999097	0.000907	314.24409	0.28374
ABCD	0.999937	0.000063	4397.56470	0.28486
CD	0.990716	0.009300	26.39872	0.24778
CDE	0.999756	0.000245	1369.40235	0.33168
FG	0.997800	0.002204	146.38771	0.32152

Table 3: Dynamic-linear Voting

Configuration Replicas	$\mathcal{A}$	$\mathcal{U}$	MTTF (Days)	MTTR (Days)
ABC	0.983699	0.016284	23.33426	0.38660
ABCD	0.998548	0.001459	168.50073	0.24714
ABCDE	0.999950	0.000051	9958.99994	0.51857
ABCFG	0.998496	0.001498	251.46865	0.37801

replicas did not fail in over 100,000 simulated days.

The results of the simulations are summarized in Tables 2 and 3. The columns of the tables indicate the configuration, the *availability,* denoted $\mathcal{A}$, that was observed for that configuration, the *unavailability,* denoted $\mathcal{U}$, which should be close to $1 - \mathcal{A}$. The observed mean-time-to-failure (MTTF) of the replicated data object and the observed mean-time-to-repair (MTTR) were also measured. The unavailability is displayed since it allows the differences in the availability provided by the various protocols to be more clearly seen.

The Available Copy protocol provides the best availability of any of the replica control protocols. The mean-time-to-fail was best for the Available Copy protocol. This result is not unexpected considering the reliability results that were obtained in §3. What is surprising is that Dynamic-linear Voting has very poor mean-times-to-failure. Although the reliability results indicate that they should be worse than the Available Copy protocols, the degree of difference is impressive. This can be partially explained by the way that consensus protocols recover from a total system failure. While Available Copy protocols must wait for the last copy to fail to recover, consensus protocols only need to wait for a quorum to be present. The result is that while the consensus protocols fail more often, they have much quicker recovery.

The site characteristics greatly influenced the availability afforded by the various protocols. But, if the site characteristics are considered, it can be seen that site D in Table 1 is available a greater portion of the time than site C. This illustrates the effect of different site characteristics on the performance of the replica control protocol, since site D will make up for the poor behavior of site C.

These values also compare favorably with the results obtained using the most realistic data available. The results, even when mixing sites with disparate characteristics, closely match those obtained using an exponential distribution with a composite mean.

As in the reliability simulation, four representative distributions were investigated. Exponential distributions were compared with Erlang-4, uniform, and hyperexponential

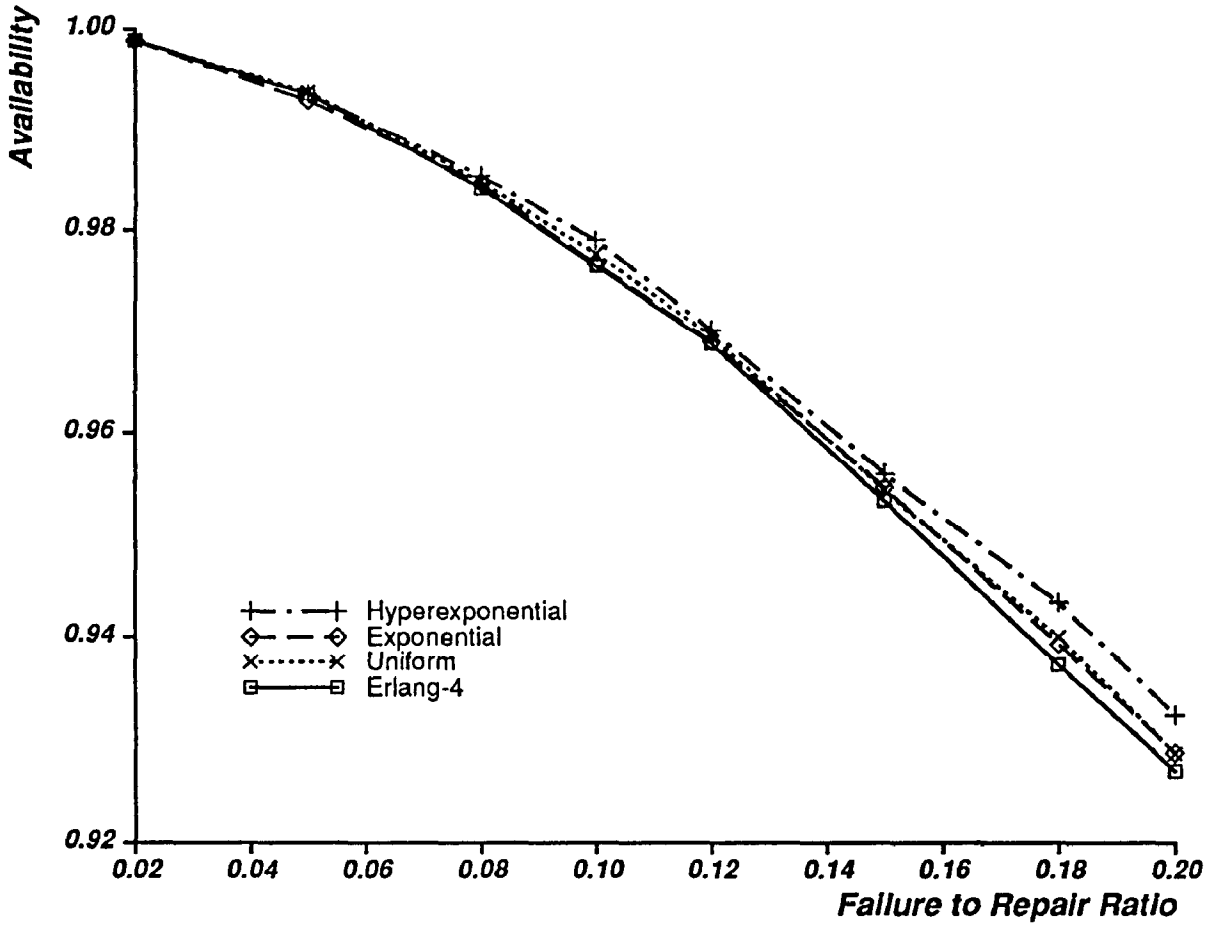

Figure 4: Availability of three sites managed by Dynamic-linear Voting

distributions. The "moment ratio" was again varied between 0.5 and 1.5.

Figures 4 and 5 display the availability provided by Dynamic-linear Voting and the Available Copy protocols, respectively. For both protocols, three replicas of the data object are assumed. The results are similar for larger numbers of replicas. In the case of both protocols, the effect of varying the higher moments of the failure and repair distributions is small.

As the failure to repair ratio increases, the differences become more pronounced. Although not apparent in the graphs, the differences for smaller values of the failure to repair ratio are distinguishable although, as is obvious from the scale, all the availabilities are extremely high. As with reliability, an increase in the second moment leads to a slight increase in availability.

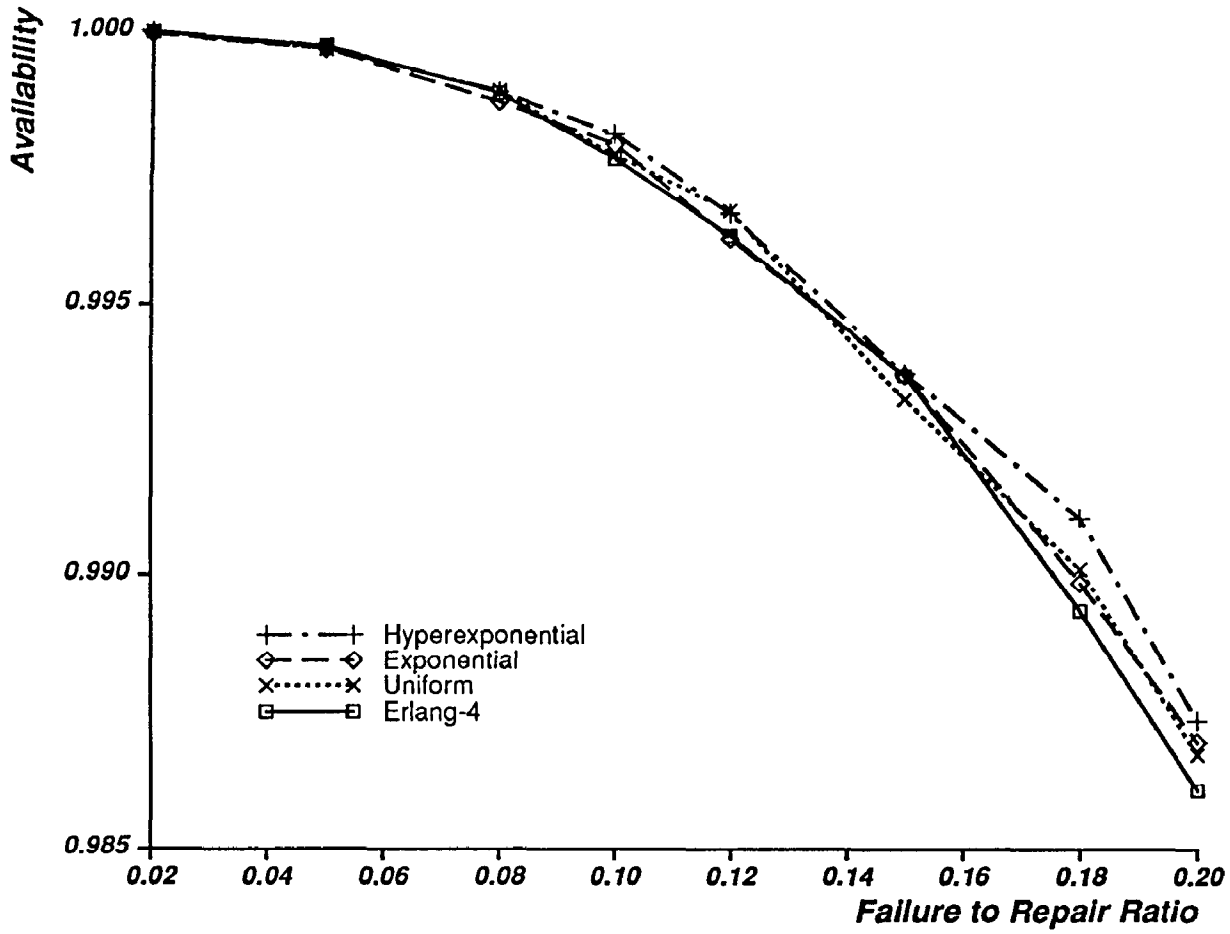

Figure 5: Availability of three sites managed by Available Copy

5 Conclusions

Two simulation models aimed at comparing the performance of several consistency control protocols for replicated data objects in real-life situations have been presented. By using discrete event simulation, the results obtained using Markov models can be validated and the robustness of the exponential approximations judged. The distributions investigated here cover a wide range of second moments, and clearly support the contention that the actual system distributions will produce results that are consistent with the behavior predicted by exponential distributions. This enhances the importance and applicability of the results obtained by theoretical analysis, which depend on Markovian assumptions.

Since no replicated object can ever be accessed without having at least one current replica, Available Copy protocols are known to provide the highest possible reliability figures of all consistency protocols that do not incorporate new sites to replace the ones that have failed [9,16]. Available Copy also has the highest availability of any of these protocols. While the availabilities of the various schemes are somewhat similar, the reliabilities differ by orders of magnitude. Where reliability is important, the Available Copy protocols are clearly superior [17]. It is clear from Figure 1 that the shapes of the failure and repair distributions in no way affect the relative merits of the protocols.

However, it must be noted that the Available Copy protocol cannot guarantee consistency in the presence of network partitions, and are therefore inappropriate in environments in which partial communication failures are possible, such as when sites are separated by gateways. The Available Copy protocol is guaranteed to function correctly

if all replicas of the data object are stored on the same carrier sense segment or token ring, and hence it is both appropriate and desirable in many applications. Voting protocols guard against the partitioning problem at the expense of performance.

In the absence of contrived circumstances, the first moments of the failure and repair distributions are the overwhelmingly predominant factors. With equal first moments, larger second moments slightly increase performance, and third and higher moments seem to have no measurable effect.

It is possible to postulate very unrealistic distributions that yield results at variance with the exponential predictions. Sites with identical constant failure and repair distributions in which the failures were synchronized could result in accessibility that was no better than that provided with a single site. On the other hand, constant distributions in which the failures were staggered could instead ensure perfect reliability and availability. In a similar fashion, using distributions with very small variances can result in accessibility that is either insignificantly better than a single site or almost perfectly accessible, depending on the initial staggering of the distributions.

Discrete event simulation also provided the ability to study more realistic scenarios using data obtained from real systems. While each of a wide range of distributions behaved similarly to the exponential distributions, the realistic data yielded results that were gratifyingly close to the exponential approximations.

Acknowledgements

The authors wish to thank Jehan-François Pâris and Alexander Glockner for their contributions and suggestions. Simulation results were obtained with the aid of SIMSCRIPT, a simulation language developed and supported by CACI Products Company of La Jolla, CA.

Bibliography

[1] Allen, A.O. *Probability, Statistics and Queueing Theory.* Computer Science and Applied Mathematics Series. New York: Academic Press, 1978.

[2] Bernstein, P.A. and N. Goodman. "An Algorithm for Concurrency Control and Recovery in Replicated Distributed Databases." *ACM Transactions on Database Systems* 9 (December 1984): 596–615.

[3] Bernstein, P.A., V. Hadzilacos and N. Goodman. *Concurrency Control and Recovery in Database Systems.* Reading, Massachusetts: Addison-Wesley, 1987.

[4] Carroll, J.L., "Analysis of the Availability and Reliability of Replicated Data Objects." In *Proceedings of the 1989 Western Multiconference,* San Diego: 1989.

[5] Carroll, J.L., D.D.E. Long and J.-F. Pâris. "Block-Level Consistency of Replicated Files." In *Proceedings of the* 7^{th} *International Conference on Distributed Computing Systems,* Berlin: 1987.

[6] Davčev, D. and W.A. Burkhard. "Consistency and Recovery Control for Replicated Files." In *Proceedings of the* 10^{th} *ACM Symposium on Operating Systems Principles,* Orcas Island: ACM, 1985, 87–96.

[7] Fishman, G.S. *Principles of Discrete Event Simulation,* New York: Wiley and Sons, 1978.

[8] Gifford, D.K. "Weighted Voting for Replicated Data." In *Proceedings of the* 7^{th} *ACM Symposium on Operating System Principles,* Pacific Grove: ACM, 1979, 150–161.

[9] Glockner, A. "Optimal Consistency Protocols for Replicated Files." In *Proceedings of the 8th International Conference on Computers and Communication,* Phoenix: IEEE, 1989, to appear.

[10] Jajodia, S. "Managing Replicated Files in Partitioned Distributed Database Systems." In *Proceedings of the 3rd International Conference on Data Engineering,* Los Angeles: IEEE, 1987, 412–418.

[11] Jajodia, S. and D. Mutchler. "Dynamic Voting." In *ACM SIGMOD International Conference on Data Management,* ACM, 1987, 227–238.

[12] D.D.E. Long. "The Management of Replication in a Distributed System." Ph.D. dissertation, Department of Computer Science and Engineering, University of California at San Diego, 1988.

[13] D.D.E. Long, J.L. Carroll and K. Stewart. "The Reliability of Regeneration-Based Replica Control Protocols." *Submitted for publication.*

[14] Long, D.D.E. and J.-F. Pâris. "On Improving the Availability of Replicated Files." In *Proceedings of the 6th Symposium on Reliability in Distributed Software and Database Systems,* Williamsburg: IEEE, 1987, 77–83.

[15] Long, D.D.E. and J.-F. Pâris. "A Realistic Evaluation of Optimistic Dynamic Voting." In *Proceedings of the 7th Symposium on Reliable Distributed Systems,* Columbus: IEEE, October, 1988.

[16] Long, D.D.E. and J.-F. Pâris. "Regeneration Protocols for Replicated Files," In *Proceedings of the 5th International Conference on Data Engineering,* Los Angeles: IEEE, 1989, to appear.

[17] Long, D.D.E., J.-F. Pâris and J.L. Carroll. "Reliability of Replicated Data Objects." In *Proceedings of the 8th International Conference on Computers and Communication,* Phoenix: IEEE, 1989, to appear.

[18] Noe, J.D. and A. Andreassian. "Effectiveness of Replication in Distributed Computing Networks." In *Proceedings of the 7th International Conference on Distributed Computing Systems,* Berlin: IEEE, 1987, 508–513.

[19] Pâris, J.-F. and W.A. Burkhard. "On the Availability of Dynamic Voting Schemes." Technical Report 86-090, San Diego: Department of CSE, University of California, San Diego, 1986.

[20] Schlichting, R.D. and F.B. Schneider. "Fail Stop Processors: An Approach to Designing Fault-Tolerant Computing Systems." *ACM Transactions on Computer Systems* (1983): 222–238.

[21] Skeen, D. "Determining the Last Process to Fail." *ACM Transaction on Computer Systems* 3 (1985): 15-30.

[22] Thomas, R.H. "A Majority Consensus Approach to Concurrency Control." *ACM Transactions on Database Systems* 4, (1979): 180–209.

[23] Trivedi, K.S. *Probability & Statistics with Reliability, Queueing and Computer Science Applications.* Englewood Cliffs, New Jersey: Prentice-Hall, 1982.