Proceedings of the Annual Simulation Symposium (to appear).

# A Simulation Study of Replication Control Protocols Using Volatile Witnesses

*Perry K. Sloope* *Jehan-François Pâris* *Darrell D.E. Long*

Department of Computer Science
University of Houston
Houston, TX 77204-3475

Computer and Information Sciences
University of California
Santa Cruz, CA 95064

## Abstract

Voting protocols guarantee the consistency of replicated data objects by disallowing all access requests that cannot gather a sufficient quorum of replicas. The performance of voting protocols can be greatly enhanced by adding to the replicas small independent entities that hold no data but can attest to the state of the replicated data object. It has been recently proposed to store these *witnesses* in volatile storage. Volatile witnesses repond faster to write requests than those stored in stable storage. They can also be more easily regenerated as many local area networks contain a majority of diskless sites.

We present a simulation study of the availability afforded by two voting protocols using volatile witnesses and investigate the impact of access rates, network topology and witness placement on the availability of the replicated data.

**Keywords:** replicated data management, replication control protocols, voting, witnesses, discrete event simulation.

## 1. Introduction

Many distributed systems maintain multiple copies of some critical data either as protection against equipment failures or to improve read access times. Managing replicated data presents a difficult challenge as site failures and network malfunctions are likely to result in inconsistent updates. Special *replication control protocols* have been proposed to prevent this occurrence. These replication control protocols define a new abstraction that has the same semantics as an unreplicated instance of the object being replicated while providing a higher resiliency.

Replication control protocols vary greatly in their complexity, the number of messages sent, and the level of protection they provide against site failures and network malfunctions. As a result, the evaluation of their performance has become an area of great practical interest. In most cases, the most important aspect of this performance is the *availability* of the replicated data object managed by the protocol. The availability of a replicated data object represents the steady-state probability that the object is available at any given moment.

Several techniques have been used to evaluate the availability of replicated data. Combinatorial models are very simple to use [19] but cannot represent complex recovery modes like those found in voting protocols with witnesses [16] and dynamic voting protocols [2]. Stochastic models have been extensively used to study replication protocols [5, 16, 15] but suffer from two important limitations: First, stochastic models quickly become intractable unless all failure and repair processes have exponential distributions. Second, stochastic models do not describe communication failures well since the number of distinct states in a model increases exponentially with the number of failure modes being considered.

Discrete event simulation does not suffer from these limitations. Simulation models allow for the relaxation of most assumptions that are required for stochastic models. They can also represent systems with communication failures. For all of its advantages, simulation has one major disadvantage: it provides only numerical results. This makes it more difficult to predict how the modeled system would behave when some of its parameters are modified because the simulation must be run again when a parameter is changed.

We present a simulation study of the performance of two new voting protocols that use *regenerable volatile witnesses*. Witnesses [16] are small entities that keep track of the state of a replicated data object but hold no data. They are used by replication control protocols to reduce the number of replicas required to achieve a given

level of availability [4, 8]. There have been recent proposals to store these witnesses in volatile storage [18] and to regenerate witnesses that fail instead of waiting for them to recover [15]. Our study completes the stochastic analyses found in [18] and [15] by providing availability data on configurations of replicated data objects that could not be analyzed using the stochastic approach.

The remainder of this paper is organized as follows: Section two introduces voting protocols with volatile witnesses. Section three discusses our simulation model and section four presents our results. Finally, our conclusions are presented in section five.

## 2. Voting protocols with volatile witnesses

*Majority consensus voting* (MCV) [3] ensures the consistency of replicated data objects by requiring that all read and write requests collect the votes of a majority of replicas. MCV offers the important advantage of operating correctly in the presence of network partitions. This robustness comes at a price: MCV requires at least *three* replicas to guarantee full access to the data in the presence of a *single* site failure.

*Voting with witnesses* [16] reduces the number of replicas necessary to achieve a given level of fault-tolerance by replacing some replicas with *witnesses* that hold no data but can attest to the state of the replicated data object. Storage requirements can be significantly reduced since two replicas and one witness provide almost the same availability as three complete replicas while saving 33% of the storage cost.

*Volatile witnesses* are witnesses that are stored in volatile memory and so can be placed on diskless sites. Volatile witnesses have one major limitation: they cannot recover on their own from a site failure and will need to read the current state of the replicated data object from a valid quorum of witnesses and replicas. The recovery process associated with each site that holds one or more volatile witnesses thus needs to put its recovering volatile witnesses in a state of *temporary amnesia* and to prevent them from participating in quorums until they can read the current status of the replicated data object.

Because of their very small size, new volatile witnesses can be created inexpensively and stored on any site in the network. *Voting with regenerable volatile witnesses* [15] takes advantage of this property and replaces failed witnesses instead of waiting for the failed site to recover. This technique provides protection similar to additional witnesses but at a much lower cost.

In this study, we investigate two replication control protocols using regenerable volatile witnesses. The first protocol is a majority consensus voting protocol where some of the replicas are replaced by regenerable volatile witnesses. Regeneration is attempted every time the protocol detects that one or more witnesses does not respond to a quorum call. Hence, frequent accesses ensure a full complement of witnesses and improve the availability of the replicated object. Accomplishing a regeneration requires a majority of the replicas and witnesses. Once a quorum has been formed, spare sites are selected and new volatile witnesses are created.

Inconsistencies could occur if the replaced witnesses could vote again. To prevent this, we add a *generation number* [17] to all voting entities. This generation number is incremented each time regeneration occurs. When a quorum call is made, the maximum generation number is calculated. Witnesses that do not have the maximum generation number are excluded from the vote. A quorum is present if either a majority of the votes were collected and at least one current replica was present, or all replicas responded to the quorum call. The second condition is necessary to guarantee that a quorum can be found in the event that all witnesses have failed and the replicas do not constitute a majority quorum.

The second protocol we investigated is the optimistic dynamic voting protocol that was proposed by Pâris and Long in [15]. Unlike MCV and other static voting protocols, dynamic voting protocols exclude from quorum computations all replicas and witnesses that have become unreachable until they again become reachable and can be reintegrated into the current majority [2, 5, 11]. The optimistic dynamic voting protocol uses sets of site names, called *partition sets*, to record the identity of the replicas and witnesses that belong to the current majority. These partition sets are maintained at each site holding a replica or a witness. The partition set for a site contains the sets of replicas and witnesses that participated in the last successful operation that included that site. These partition sets are recalculated whenever the data are accessed or a site recovers from a failure. In addition to these partition sets, each site holding a replica or a witness maintains a *version number* and an *operation number*. Version numbers are incremented every time the replicated data is modified while operation numbers are incremented every time the partion sets are modified. This is similar to the information required by the original *optimistic dynamic voting* protocol [11].

Unlike the MCV protocol, the ODV protocol uses a two-tier voting strategy to minimize the impact of witness failures on the availability of the replicated data objects. A quorum is present if any of the following four conditions holds:

1. The set of responding replicas constitutes a majority of replicas in the previous quorum.
2. The set of responding replicas contains exactly half of the replicas from the previous quorum and a majority of the current witnesses also respond.
3. The set of responding replicas contains exactly half of the replicas from the previous quorum and exactly half of the current witnesses respond. In this case, a quorum exists if the highest ranking witnesses in some total ordering is present.
4. There were no witnesses present in the previous quorum, exactly one half of the replicas in the previous quorum have responded and the set of responding replica includes the highest ranking replica in some total ordering of the previous quorum.

As in the MCV protocol, regeneration is attempted every time the protocol detects that one or more witnesses do not respond to a quorum call. Generation numbers are not required as witnesses that do not respond to a quorum call are automatically expelled from the current majority.

## 3. The simulation model

Most studies of replicated data availability have depended on probabilistic models to evaluate the availability of replica control protocols. These models do not generally consider the effect of network partitioning, because of the enormous complexity that would be involved. As a result, the data that they present are for ideal environments that are unlikely to exist under actual conditions. We use discrete event simulation to observe the behavior of two replica control protocols under more realistic conditions. We also develop a heuristic for witness placement that improves availability by considering the effect that network partitioning will have on the protocols.

Many parameters can effect the availability of replicated data. Not all sites in a network will be equally reliable. The mean time to fail (MTTF) and mean time to repair (MTTR) will vary among the sites, as will the types of failures. A failure due to software errors will involve a simple reboot of the machine taking only minutes, while a hardware failure may result in a site being down for hours or even days. Some sites may be taken down periodically for maintenance, while others may never be maintained until they fail. Periodic installation of new software may result in one or more sites being removed from general use for a few hours.

Simulation models that consider these factors have been developed. In [14], a simulation study of voting protocols based on a network existing then at the UCSD computer science department was presented. Our simulation models are equally capable of modeling all of these possible failure modes and have the additional capability to simulate arbitrary network configurations [21].

The modeled networks are assumed to be collections of carrier-sense segments or token rings linked together by gateway sites or repeaters. Site failures are assumed to be fail-stop [20], ceasing operations immediately if not operating correctly. The network may be partitioned if a gateway site or repeater fails, but sites on the same carrier-sense segment or token ring always remain capable of communicating with each other. Messages can be lost due to gateway failures but messages that reach their destination are always delivered intact, in the order they were sent. All sites are capable of storing a replicated data object and manipulating the data.

Users are assumed to be able to access the replicated data object from any point in the network and are modeled as a single process. Hence, the replicated data are assumed to be available as long as they can be accessed from at least one site in the network.

Jajodia and Mutchler have recently introduced another definition of data availability, which we will call *user availability* [5]. They define the availability of replicated data as the average fraction of the replicas that remain accessible at equilibrium. User availability is a good index of the performance of consistency protocols distributed over long-haul networks. If a replicated object is not accessible from a local site, access at a remote site is not possible due to the sheer distances involved. It is a less accurate measure of the performance of protocols distributed over local-area networks since it is highly likely that users unable to reach one server will be able to connect to another one.

### 3.1. General organization

The models for the two replication control policies were programmed in SIMSCRIPT II.5 and run on a Sun 3/60. The process approach was selected since it can describe the network sites and the user as independent processes. Information on the mean availability and unavailability of the replicated data were collected using standard SIMSCRIPT II.5 statistical routines. All simulations were started with all sites operating and a time-to-steady-state interval of 360 days; these were considered acceptable by noting the time spent in each state and by examining graphs of the measured values in the early history of the simulations. The simulations were run long enough to establish reasonable 95% confidence intervals for all replicated data object unavailabilities.

To analyze the output data, batch-means analysis was chosen for its simplicity and its general applicability [LaMi87]. After the initial bias was removed by ignoring

the statistics gathered during the time-to-steady-state interval, a very long simulation run was divided into time segments of equal size called *batches.* Pairs of batches were then merged until the resulting batches could be considered statistically independent of each other. The final batch size was chosen using a heuristic developed by Law and Carson [6]: After finding a batch size $m$ where the batches have a correlation of 0.4 or less, the batches of size $10m$ could be considered uncorrelated.

### 3.2. Site processes

There is a separate site process for each physical site in the network, and each site has a unique identity number. Each site works for an exponentially distributed period of time based on the MTTF. Upon failure, the site activates a procedure that determines if the failure of the site changes the availability status of the replicated data object. The site then waits an exponentially distributed period of time, based on the MTTR. When this period expires, it activates a recovery procedure.

The recovery procedure varies slightly depending on the protocol being modeled, but all operate in a similar manner. It first determines if the recovery of the site changes the availability status of the replicated data object. Afterwards, if the site does not contain a replica, it terminates. Otherwise, it performs whatever recovery process is required by the protocol. This generally involves an attempt at gathering a quorum in order to update the replica. If a quorum cannot be gathered, it waits a short period of time and tries again.

### 3.3. The user process

There is one user process that is capable of accessing the replicated data object from any site in the network. The primary purpose of this routine is to periodically activate the access routines. This is critical for dynamic voting protocols, since the availability they provide is dependent on the frequency of the accesses. The user process also calculates the percentage of accesses that were successfully completed. An access is considered successful if it can be performed from at least one site in the network. For the sake of brevity, we present only the total unavailability and not the percentage of successful accesses.

The user process waits an exponentially distributed period of time between accesses. It then decides whether to perform a read or write operation. The choice is made randomly such that a read to write ratio of 4:1 is achieved. Experimental studies of access patterns have shown this to be a reasonable assumption [13, 1]. The user process then activates the read or write routine, which, in turn, activates the appropriate routines to effect the operation for the modeled protocol. If the access fails, the process is repeated on the remaining sites with replicas, until it is successful or all replicas are exhausted. The access times used for the models are discussed in the following sections.

### 3.4. The access routines

These vary depending on the protocol being modeled, but each operates in the manner described for the protocol. Here we will describe the general form for accessing, common to all of them. When the read or write routines are activated, they are given the ID of the site from which the access is to be made. The routine(s) for the modeled protocol, that attempt to gather the quorum of sites needed for an access, are activated, and given the ID of the originating site. These routines in turn activate a routine that determines the set of sites that can communicate with the originating site. The access routine then determines if this set of sites constitutes a quorum for the modeled protocol.

A critical part of the model is determining if two sites can communicate. Since all the protocols rely on communications between sites to determine the status of the replicated data, a fast simple means was needed to determine communication links. For sites on the same carrier-sense network, the solution is simple. If any two sites are up and running, it can be assumed that they can communicate. For sites not on the same network segment, that assumption cannot be made since they may be separated by one or more gateway sites or repeaters. A solution to this was found by viewing the network as a tree structure whose nodes consist of the different network segments, and their respective sites. One segment is chosen as the root. Communications are determined by traversing the tree between two sites. The tree structure is conceptual and is represented by two arrays whose indices represent the identity number of each site. The first of these two arrays indicates if a site is currently operating or is down due to failure. The second array indicates if a site is connected to a higher level node by a gateway site. If a site is in the root segment, then its entry in the array is 0. Otherwise, it contains the identity number of the gateway site.

Given the identity numbers of two sites, the communications routine determines if they can communicate. It begins by checking the operating status of one of the sites. If the site is down, the routine exits with failure. Otherwise, using the identity of the given site, it retrieves the identity of the gateway site from the second array. It then checks the status of the gateway site and retrieves the identity of its gateway site. The process continues until the gateway identity retrieved is that of the root node. If

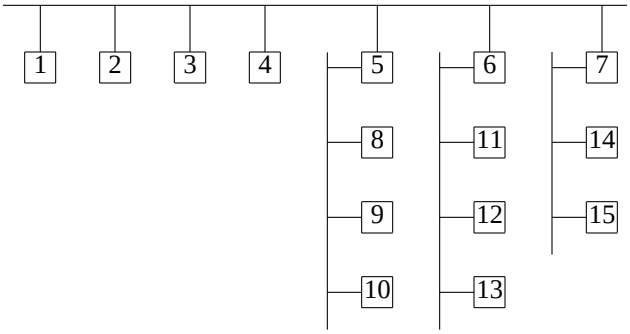


**Figure 1: Network Configuration**

at any time the status of one of the sites checked is not operational, the routine terminates with failure. The entire process is repeated for the second site that was passed to the communications routine. If this process reaches the root node for both sites without encountering an inoperative site, then the two sites can communicate.

## 4. Simulation results

Instead of simulating a specific network topology as in [14], we decided instead to consider a generic configuration that was likely to be encountered in many local area networks. This network configuration is shown in Figure 1. It consists of a backbone segment with seven sites to which three smaller segments are linked through gateways.

To simplify the analysis of our simulation results, we decided to run all our simulations assuming that all sites had identical mean times to fail and mean times to repair. If individual site attributes had been considered, it would have been difficult to determine if the availability was being affected by partitioning or by some other factor such as a site failing frequently for short periods of time. We assumed therefore that all sites have a MTTF of 19 days and a MTTR of one day which effects a site availability of 95%. This is consistent with measurements of site reliability made using the Internet [9].

The nine configurations considered in our study are enumerated in Table 1. The first three configurations consist of three replicas. Configuration A does not allow for partitions, B allows for three different partitions and C allows for two partitions. The following three configurations consist of four replicas. Configuration D does not allow for partitioning, E has four partitions, and F allows for three partitions. The remaining configurations are for five replicas, and allow for the same amount of partitioning as for the preceding configuration. For MCV with four voting entities, one of the replicas was given two votes in order to avoid ties.

Previous studies of replication control protocols that regenerate replicas or witnesses [12, 19, 10, 15] only considered environments where network partitions are excluded. As a result, these studies did not treat in depth the issue of spare site selection. In a study of regenerating full replicas [10], one of the authors suggested that the site with the best fault tolerance characteristics should be selected. While that will enhance availability, it does not address fully the effects of further partitioning. The newly regenerated replica may soon be isolated due to the failure of a gateway site, causing the data to become unavailable. We decided on a simple heuristic that attempts to address this problem: newly regenerated witnesses are to be placed on the LAN segment that contains the largest number of replicas in the current majority partition. For ODV, we add the following tie-breaking rule: in the event of a tie, newly regenerated witnesses are to be placed on the LAN segment with the highest ordered replica.

These two rules attempt to place the new witness in a location where it is least likely to be separated from the majority of replicas by network partition. The second rule is required because of the manner in which ODV determines quorums. In a quorum call, if exactly one-half of the replicas respond, the quorum is successful if the highest ordered replica is present. By placing a witness in that partition, a quorum will still be possible even if less than half the replicas are available. Since the witnesses are volatile, they can be placed on diskless sites. Also the witnesses are never actually accessed by the user, so their location is not as dependent on data usage patterns as are the actual replicas.

Since all sites were assumed to have identical failure characteristics, we did not consider placing witnesses on sites with the best fault tolerance characteristics. If an available site cannot be found on the LAN segment with the most replicas, then an attempt is made on the LAN segment containing the next most populated group of replicas. If a site cannot be found on any LAN segments with replicas, then a site is chosen randomly. In this manner, the heuristic attempts to predict the occurrence of a future current partition. This concentrates the voting entities in an environment where they cannot be separated as easily by partitioning.

The performance of protocols using volatile witnesses is dependent on the access rate to the replicated data. As the access rate increases, failed witnesses are detected and regenerated more rapidly, and the performance of the protocols improve. To study this effect, we ran simulations for all configurations at four different access rates:

**Table 1: Replica Placement.**

| Configuration | Included Sites | | | |
|---|---|---|---|---|
| | 1st LAN | 2nd LAN | 3rd LAN | 4th LAN |
| A | 2, 5, 7 | | | |
| B | 2 | 8 | 11 | |
| C | 6, 7 | 8 | | |
| D | 1, 5, 6, 7 | | | |
| E | 2 | 8 | 11 | 14 |
| F | 2, 7 | 8 | 11 | |
| G | 1, 3, 5, 6, 7 | | | |
| H | 2, 4 | 8 | 11 | 14 |
| I | 2, 7 | 8, 9 | 11 | |

one access per week, one per day, one per hour, and at a continuous access rate. Tables 2 to 5 summarize the results of the simulations for all configurations. Data unavailabilities were measured instead of data availabilities as they show more clearly differences among the protocols. The column labeled ''Sites with Witnesses'' indicates the placement of the witnesses. Unavailabilities for optimistic dynamic voting and majority consensus voting are in the columns labeled ODV and MCV, respectively.

Without witnesses, ODV outperformed MCV in all cases for access rates of one per hour and constant accessing, often performing twice as well as MCV. This was expected since it had been predicted in prior studies [11]. For the slower access rates, ODV performed about as well as MCV for configurations using three replicas, and performed better than MCV when using more than three replicas. The reason for this is that system state information used by ODV is updated during an access operation, and when a replica recovers from a failure. At lower access rates, the updating of this information becomes more dependent on recovering sites than at high access rates. As the number of replicas are increased, updates due to recovering sites also are increased and the protocol operates more efficiently.

To evaluate the performance of ODV using witnesses, we first consider its performance at a constant access rate. For all cases ODV outperformed MCV. When one replica was replaced by a witness, the unavailability provided by ODV was decreased in each case. In some cases it performed twice as well as ODV without witnesses. This demonstrates that storage costs can be decreased and availability improved by the replacement of a replica with a regenerable witness. Alternately, the availability provided by a given number of replicas can be improved by the addition of a witness.

When two replicas were replaced by witnesses, the availability provided was less than that without witnesses, or with one witness. The reason for this is that a quorum cannot be gathered if all replicas fail within the same period of time. The likelihood of this increases as the number of replicas decrease and the number of witnesses increase. A threshold is reached, at which point the probability of all replicas failing simultaneously is greater than the availability provided by the protocol when only replicas are used. That threshold is reached at two witnesses with ODV for the configurations we considered.

Availability can be increased by the addition of witnesses without replacing replicas, but a similar threshold is reached at which increasing the number of witnesses does not increase availability. For instance, configuration D with one witness can be considered as three replicas, with one witness added to increase availability. This can be compared to configuration A, which utilizes three replicas and no witnesses. In this case, configuration D with one witness provides 20 times less unavailability than does configuration A. Similarly, configuration G with two witnesses is equivalent to using three replicas, with the addition of two witnesses to increase availability. However, when compared to D with one witness, no increase in availability is observed. This is because the optimum availability that can be achieved with three replicas for this protocol, is attained by the addition of one witness.

For MCV at a constant access rate, replacing one replica with a witness resulted in availabilities about the same or better than MCV with only replicas. When two replicas were replaced, the availability was the same or better than when just one replica was replaced. For five replicas, replacing one or two replicas resulted in improved availability for each case. The addition of two witnesses to a given number of replicas provided better

**Table 2: Data Unavailabilities for Continuous Access**

| Configuration | Sites with Witnesses | MCV (95% Confidence Intervals) | | ODV (95% Confidence Intervals) | |
|---|---|---|---|---|---|
| A | NONE | 0.007433 | ± 0.000253 | 0.007101 | ± 0.000337 |
| | 2 | 0.007537 | ± 0.000317 | 0.004628 | ± 0.000258 |
| B | NONE | 0.018050 | ± 0.000390 | 0.011311 | ± 0.000372 |
| | 2 | 0.012074 | ± 0.000394 | 0.007402 | ± 0.000352 |
| C | NONE | 0.011752 | ± 0.000322 | 0.009014 | ± 0.000364 |
| | 6 | 0.011979 | ± 0.000399 | 0.006988 | ± 0.000318 |
| D | NONE | 0.007239 | ± 0.000249 | 0.000635 | ± 0.000095 |
| | 1 | 0.007243 | ± 0.000283 | 0.000339 | ± 0.000069 |
| | 1,7 | 0.007219 | ± 0.000319 | 0.004921 | ± 0.000271 |
| E | NONE | 0.014139 | ± 0.000349 | 0.001857 | ± 0.000171 |
| | 11 | 0.011474 | ± 0.000344 | 0.000998 | ± 0.000128 |
| | 11,15 | 0.012518 | ± 0.000418 | 0.007856 | ± 0.000356 |
| F | NONE | 0.011784 | ± 0.000314 | 0.001536 | ± 0.000156 |
| | 2 | 0.011410 | ± 0.000340 | 0.001000 | ± 0.000130 |
| | 2,8 | 0.010601 | ± 0.000381 | 0.007149 | ± 0.000349 |
| G | NONE | 0.001207 | ± 0.000087 | 0.000055 | ± 0.000035 |
| | 3 | 0.000862 | ± 0.000082 | 0.000019 | ± 0.000019 |
| | 1,3 | 0.000545 | ± 0.000075 | 0.000337 | ± 0.000067 |
| H | NONE | 0.004056 | ± 0.000156 | 0.000192 | ± 0.000052 |
| | 2 | 0.003491 | ± 0.000161 | 0.000088 | ± 0.000038 |
| | 2,11 | 0.002819 | ± 0.000148 | 0.001035 | ± 0.000125 |
| I | NONE | 0.011131 | ± 0.000311 | 0.000589 | ± 0.000099 |
| | 2 | 0.009581 | ± 0.000311 | 0.000284 | ± 0.000074 |
| | 2,11 | 0.009627 | ± 0.000357 | 0.001391 | ± 0.000141 |

availability than just adding one witness for the configurations studied. This can be seen by observing the unavailabilities provided by configurations A and D with one witness, and G with two witnesses.

With an even number of replicas, MCV provides about the same availability as it does with one less replica, even when one replica is given an extra vote to break ties. In this case, the availability can be greatly enhanced by the addition of one witness to provide an odd number of votes. For example, configuration G with one witness is equivalent to D with the addition of a witness and it provides more than eight times less unavailability.

At an access rate of one per hour, both ODV and MCV experienced a slight increase in unavailability, but in general the results were similar to those at a constant access rate. As was expected, at access rates of one per day and one per week, the performance of the protocols using witnesses were worse than when witnesses were not used. As was discussed earlier, the performance of protocols using regenerable volatile witnesses is dependent on the access rate to the data. When the time between accesses is greater than the MTTR of the failed site, the detection of failed witnesses is too slow to effect an increase in availability over that when witnesses are not used. It can also be seen that the difference in performance between ODV and MCV using witnesses starts to narrow as the access rate decreases. This is most apparent for the one per week access rate, although ODV still outperforms MCV in most cases.

## 5. Conclusions

We have presented in this paper a comparative simulation study of two voting protocols with volatile witnesses. We found that the dynamic voting protocol with regenerable volatile witnesses outperformed majority consensus voting and majority consensus voting with volatile witnesses in most cases. ODV with regenerable volatile witnesses can provide better data availability than MCV while using less replicas. We also found that the location of the witnesses had an impact on the protocol performance. More work is needed to get a better grasp of this impact and develop better topology-dependent witness placements heuristics. It might be that the performance of ODV with regenerable volatile witnesses can be

**Table 3: Data Unavailabilities for One Access per Hour.**

| Configuration | Sites with Witnesses | MCV (95% Confidence Intervals) | | ODV (95% Confidence Intervals) | |
|---|---|---|---|---|---|
| A | NONE | 0.007433 | ± 0.000253 | 0.007119 | ± 0.000329 |
| | 2 | 0.008673 | ± 0.000353 | 0.004912 | ± 0.000272 |
| B | NONE | 0.018050 | ± 0.000390 | 0.011589 | ± 0.000419 |
| | 2 | 0.016083 | ± 0.000483 | 0.007680 | ± 0.000357 |
| C | NONE | 0.011752 | ± 0.000322 | 0.009564 | ± 0.000394 |
| | 6 | 0.016273 | ± 0.000493 | 0.007629 | ± 0.000339 |
| D | NONE | 0.007239 | ± 0.000249 | 0.000912 | ± 0.000102 |
| | 1 | 0.007158 | ± 0.000687 | 0.000681 | ± 0.000091 |
| | 1,7 | 0.007929 | ± 0.000319 | 0.005192 | ± 0.000282 |
| E | NONE | 0.014139 | ± 0.000349 | 0.002626 | ± 0.000186 |
| | 11 | 0.012058 | ± 0.000358 | 0.001895 | ± 0.000155 |
| | 11,15 | 0.013854 | ± 0.000454 | 0.008301 | ± 0.000361 |
| F | NONE | 0.011784 | ± 0.000314 | 0.001966 | ± 0.000166 |
| | 2 | 0.011820 | ± 0.000350 | 0.001738 | ± 0.000148 |
| | 2,8 | 0.013229 | ± 0.000439 | 0.007687 | ± 0.000337 |
| G | NONE | 0.001207 | ± 0.000087 | 0.000134 | ± 0.000044 |
| | 3 | 0.000906 | ± 0.000086 | 0.000098 | ± 0.000028 |
| | 1,3 | 0.000661 | ± 0.000081 | 0.000693 | ± 0.000093 |
| H | NONE | 0.004056 | ± 0.000156 | 0.000358 | ± 0.000058 |
| | 2 | 0.003633 | ± 0.000173 | 0.000439 | ± 0.000069 |
| | 2,11 | 0.003479 | ± 0.000199 | 0.001815 | ± 0.000145 |
| I | NONE | 0.011131 | ± 0.000311 | 0.001035 | ± 0.000105 |
| | 2 | 0.009523 | ± 0.000313 | 0.000813 | ± 0.000093 |
| | 2,11 | 0.010622 | ± 0.000392 | 0.001732 | ± 0.000152 |

improved by restricting regeneration to the LAN segment on which the witness was originally spawned.

We also found that static MCV with volatile witnesses can provide comparable availability to MCV while using less replicas. Here too, the choice of the location of the witnesses significantly affects data availability.

## Acknowledgements

Darrell D. E. Long was supported in part by the National Science Foundation under Grant NSF CCR-9111220, and by the Institute for Scientific Computing Research at Lawrence Livermore National Laboratory. The simulation results were obtained with the aid of SIMSCRIPT II.5, a simulation language developed and supported by CACI Products Company of La Jolla, CA. We are grateful to Robb Mills for providing us with the code used to compute batch averages.

## References

[1] M. G. Baker, J. H. Hartman, M. D. Kupfer, K. W. Shirriff and J. K. Ousterhout, ‘‘Measurements of a Distributed File System,’’ *Proc. 13th ACM Symposium on Operating System Principles*, (1991), pp. 198-212.

[2] D. Davcev and W. A. Burkhard, ‘‘Consistency and Recovery Control for Replicated Files,’’ *Proc. 10th ACM Symposium on Operating System Principles*, (1985) pp. 87-96.

[3] D. K. Gifford, ‘‘Weighted Voting for Replicated Data,’’ *Proc. 7th ACM Symposium on Operating System Principles*, (1979), pp. 150-161.

[4] A. Hisgen, A. Birrell, T. Mann, M. Schroeder, and G. Swart, ‘‘Availability and Consistency Tradeoffs in the Echo Distributed File System,’’ *Proc. 2nd Workshop on Workstation Operating Systems*, (1989), pp. 49-54.

**Table 4: Data Unavailabilities for One Access per Day.**

| Configuration | Sites with Witnesses | MCV (95% Confidence Intervals) | | ODV (95% Confidence Intervals) | |
|---|---|---|---|---|---|
| A | NONE | 0.007433 | ± 0.000253 | 0.007478 | ± 0.000308 |
| | 2 | 0.011743 | ± 0.000413 | 0.009863 | ± 0.000373 |
| B | NONE | 0.018050 | ± 0.000390 | 0.015712 | ± 0.000432 |
| | 2 | 0.019857 | ± 0.000547 | 0.015994 | ± 0.000484 |
| C | NONE | 0.011752 | ± 0.000322 | 0.011651 | ± 0.000381 |
| | 6 | 0.019885 | ± 0.000535 | 0.016379 | ± 0.000489 |
| D | NONE | 0.007239 | ± 0.000249 | 0.003610 | ± 0.000160 |
| | 1 | 0.009098 | ± 0.000328 | 0.003840 | ± 0.000190 |
| | 1,7 | 0.011344 | ± 0.000384 | 0.009914 | ± 0.000394 |
| E | NONE | 0.014139 | ± 0.000349 | 0.008370 | ± 0.000270 |
| | 11 | 0.013881 | ± 0.000401 | 0.010253 | ± 0.000313 |
| | 11,15 | 0.015924 | ± 0.000474 | 0.016462 | ± 0.000492 |
| F | NONE | 0.011784 | ± 0.000314 | 0.006471 | ± 0.000241 |
| | 2 | 0.013774 | ± 0.000384 | 0.010593 | ± 0.000323 |
| | 2,8 | 0.015741 | ± 0.000471 | 0.016594 | ± 0.000494 |
| G | NONE | 0.001207 | ± 0.000087 | 0.000646 | ± 0.000066 |
| | 3 | 0.001511 | ± 0.000111 | 0.000878 | ± 0.000078 |
| | 1,3 | 0.002429 | ± 0.000179 | 0.003837 | ± 0.000187 |
| H | NONE | 0.004056 | ± 0.000156 | 0.002359 | ± 0.000129 |
| | 2 | 0.005016 | ± 0.000206 | 0.003943 | ± 0.000183 |
| | 2,11 | 0.006194 | ± 0.000274 | 0.010019 | ± 0.000309 |
| I | NONE | 0.011131 | ± 0.000311 | 0.005424 | ± 0.000214 |
| | 2 | 0.011873 | ± 0.000363 | 0.007142 | ± 0.000282 |
| | 2,11 | 0.012734 | ± 0.000414 | 0.006317 | ± 0.000237 |


[5] S. Jajodia and D. Mutchler, ‘‘Dynamic Voting Algorithms for Maintaining the Consistency of a Replicated Database,’’ *ACM Transactions on Database Systems*, Vol. 15, No. 2 (1990), pp. 230-280.

[6] A. Law and J. Carson, ‘‘A Sequential Procedure for Determining the Length of a Steady-State Simulation,’’ *Operations Research*, Vol. 27 (1979), pp. 1011-1125.

[7] A. Law and R. Mills, *Statistical Analysis of Simulation Output Data Using SIMSCRIPT II.5*, Los Angeles: CACI, Inc., (1988).

[8] B. Liskov, S. Ghemawat, R. Gruber, P. Johnson, L. Shrira and M. Williams, ‘‘Replication in the Harp File System,’’ *Proc. 13th ACM Symposium on Operating Systems Principles*, (1991), pp. 226-238.

[9] D.D.E. Long, J.L. Carroll and C.J. Park, ‘‘A Study of the Reliability of Internet Sites,’’ *Proc. 10th Symposium on Reliable Distributed Systems*, (1991), pp. 177-186.

[10] D.D.E. Long, ‘‘The Management of Replicated Data in a Distributed System,’’ Ph.D. dissertation, Department of CSE, University of California, San Diego, (1988).

[11] D.D.E. Long and J.-F. Pâris, ‘‘A Realistic Evaluation of Optimistic Dynamic Voting,’’ *Proc. 7th Symposium on Reliable Distributed Systems*, (1988), pp. 77-83.

[12] J.D. Noe and A. Andreassian. ‘‘Effectiveness of Replication in Distributed Computing Networks.’’ *Proc. 7th International Conference on Data Engineering*, (1987), pp. 508-513.

[13] J. Ousterhout, H. Da Costa, D. Harrison, J. Kunze, M. Kupfer and J. Thompson", ‘‘A Trace-Driven Analysis of the UNIX 4.2 BSD File System,’’ *Proc. 10th Symposium on Operating System Principles*, (1985), pp. 15-24.

**Table 5: Data Unavailabilities for One Access per Week**

| Configuration | Sites with Witnesses | MCV (95% Confidence Intervals) | | ODV (95% Confidence Intervals) | |
|---|---|---|---|---|---|
| A | NONE | 0.007433 | ± 0.000253 | 0.007285 | ± 0.000265 |
| | 2 | 0.022966 | ± 0.000556 | 0.022690 | ± 0.000550 |
| B | NONE | 0.018050 | ± 0.000390 | 0.018874 | ± 0.000444 |
| | 2 | 0.036012 | ± 0.000732 | 0.034360 | ± 0.000710 |
| C | NONE | 0.011752 | ± 0.000322 | 0.012749 | ± 0.000359 |
| | 6 | 0.034989 | ± 0.000719 | 0.034424 | ± 0.000714 |
| D | NONE | 0.007239 | ± 0.000249 | 0.005432 | ± 0.000202 |
| | 1 | 0.013636 | ± 0.000180 | 0.005897 | ± 0.000217 |
| | 1,7 | 0.022922 | ± 0.000562 | 0.022162 | ± 0.000562 |
| E | NONE | 0.014139 | ± 0.000349 | 0.011931 | ± 0.000321 |
| | 11 | 0.018663 | ± 0.000463 | 0.016411 | ± 0.000401 |
| | 11,1 | 0.028583 | ± 0.000633 | 0.035018 | ± 0.000738 |
| F | NONE | 0.011784 | ± 0.000314 | 0.009328 | ± 0.000278 |
| | 2 | 0.018813 | ± 0.000473 | 0.016087 | ± 0.000387 |
| | 2,8 | 0.028911 | ± 0.000631 | 0.034475 | ± 0.000715 |
| G | NONE | 0.001207 | ± 0.000087 | 0.000925 | ± 0.000075 |
| | 3 | 0.002560 | ± 0.000140 | 0.002576 | ± 0.000146 |
| | 1,3 | 0.009047 | ± 0.000357 | 0.006136 | ± 0.000226 |
| H | NONE | 0.004056 | ± 0.000156 | 0.003833 | ± 0.000163 |
| | 2 | 0.008340 | ± 0.000260 | 0.008184 | ± 0.000264 |
| | 2,11 | 0.018543 | ± 0.000543 | 0.015984 | ± 0.000396 |
| I | NONE | 0.011131 | ± 0.000311 | 0.008443 | ± 0.000263 |
| | 2 | 0.017245 | ± 0.000455 | 0.014831 | ± 0.000421 |
| | 2,11 | 0.024329 | ± 0.000599 | 0.010564 | ± 0.000314 |


[14] J.-F. Pâris, D.D.E. Long and A. Glockner, ‘‘A Realistic Evaluation of Consistency Algorithms for Replicated Files,’’ *Proceedings 21st Annual Simulation Symposium*, (1988), pp. 121-130.

[15] J.-F. Pâris and D.D.E. Long, ‘‘Voting with Regenerable Volatile Witnesses,’’ *Proc. 7th International Conference on Data Engineering*, (1991), pp. 112-119.

[16] J.-F. Pâris, ‘‘Voting with Witnesses: A Consistency Scheme for Replicated Files,’’ *Proc. 6th International Conference on Distributed Computing Systems*, (1986), pp. 606-612.

[17] J.-F. Pâris, ‘‘Efficient Management of Replicated Data,’’ *Proc. 2nd International Conference on Database Theory*, Lecture Notes in Computer Science # 326, Springer Verlag (1988), pp. 386-409.

[18] J.-F. Pâris, ‘‘Efficient Voting Protocols with Witnesses,’’ *Proc. 3rd International Conference on Database Theory*, Lecture Notes in Computer Science, Springer Verlag (1990), pp. 305-317.

[19] C. Pu, J.D. Noe and A.B. Proudfoot. ‘‘Regeneration of Replicated Objects, A Technique and its Eden Implementation,’’ *IEEE Transactions on Software Engineering*, Vol. SE-14, No. 7 (1988), pp. 936-945.

[20] R. D. Schlichting and F. B. Schneider, ‘‘Fail Stop Processors: An Approach to Designing Fault-Tolerant Computing Systems,’’ *ACM Transactions on Computer Systems*, Vol. 3 (1983), 222-238.

[21] P. K. Sloope, ‘‘A Realistic Evaluation of the Effect of Network Partitioning on the Management of Replicated Data,’’ M.S. Thesis, Department of Computer Science, University of Houston, (1991).