Cost-driven Design for Archival Repositories *

Arturo Crespo and Hector Garcia-Molina
Computer Science Department
Stanford University
Stanford, CA 94305-2140, USA
{crespo,hector}@db.stanford.edu

ABSTRACT
Designing an archival repository is a complex task because there are many alternative configurations, each with different reliability levels and costs. In this paper we study the costs involved in an Archival Repository and we introduce a design framework for evaluating alternatives and choosing the best configuration in terms of reliability and cost. We also present a new version of our simulation tool, ArchSim/C that aids in the decision process. The design framework and the usage of ArchSim/C are illustrated with a case study of a hypothetical (yet realistic) archival repository shared between two universities.

1. INTRODUCTION
Digital information can be lost for a variety of reasons: magnetic decay, format and device obsolescence, human or system error, among many others. A solution is to build an *archival repository* (AR), a system capable of storing and preserving digital objects (e.g., movies, technical reports) as technologies and organizations evolve [1].

Designing an AR is difficult, as there are many configuration options and uncertainties about the future. For example, one must decide how many sites to use, what types of disks or tape units to use, what and how many formats to use to store documents, how frequently to check existing documents for errors, what strategy to use for error recovery, how often to migrate documents to a more modern format, and so on. On top to this, the designer needs to predict future events such as the reliability of sites and disks, survivability of formats, how many resources will be consumed by the recovery algorithms, how frequently the recovery algorithms will be invoked, how many user accesses will be made to the documents, and many other uncertainties.

Two important factors must be considered in AR design: the *level of assurance* (e.g., on average a document will not be lost for 1000 years) and the *cost* (e.g., an initial investment of 1 million dollars and yearly expenses of 100 thousand dollars). There has been some research on predicting the level of assurance of a given AR [4], but there has been little or no work on predicting the cost of an AR.

Predicting the cost of an AR is difficult task. First, we need to estimate the cost for each "event" such as the AR creation, the failure and repair of a disk, etc. For many of these events, we may also have to predict when they will happen. For example, since we do not know when a disk will fail, we cannot deterministically predict when and how often we will pay for its repair. Second, we may not know for certain future costs, so we may have to represent them with probability distributions (e.g., the price of a disk may be between \$100 and \$150). As we will see in this paper, deriving cost estimates and likelihoods for a given AR requires a lot of "guess work." However, the alternative of ignoring costs altogether can easily lead to systems that are overdesigned and overpriced, or that do not meet user expectations.

In this paper we show how AR costs (and failures) can be modeled, albeit in a rough way, so that rational decisions can be made. In particular, we present a complete design framework for making cost-driven decisions about ARs, and a powerful simulation tool, *ArchSim/C* that aids in the process. Our design framework is based on Decision Analysis (DA) theory [9] and we believe that it is a good way of structuring the design of archival repositories. ArchSim/C can model important configuration options, such as multiple formats, preventive maintenance, and failure distribution functions. By using specialized techniques, ArchSim/C is able to provide cost and reliability information for a configuration in a time frame that allows the exploration and testing of different policies. To illustrate the framework, we use as a running example a case study based on a hypothetical AR of MIT/Stanford technical reports.

The contributions of this paper are:

- An in-depth study of the costs involved in an AR.
- A comprehensive design framework for making AR cost decisions.
- A new version of our simulation tool that can predict the reliability *and* the cost of an AR.
- A demonstration of the framework in a case study.

*This work was partially supported by NSF.

2. ARCHIVAL REPOSITORIES

We define an *Archival Repository* (AR) as a repository that guarantees long-term data survivability. In this section we study the elements of a typical archival repository (AR), so we can later evaluate their reliability and cost. A typical AR is formed by a *data store* that can fail and an *archival system* (AS) that ensures long-term survivability of its documents. The AS provides fault tolerance by managing multiple *materializations* for each document. A materialization is the set of all the *components* necessary to provide some sort of human access to a document. For example, a materialization may include the bits, disks, and format interpreters necessary to display a technical report. Figure 1 shows the AS main functions (in solid-line boxes), the non-fault-tolerant store (in a dashed-line box), and the archival documents. The arrows represent the runtime interactions between the elements. This representation can model many existing archival systems including the Computing Research Repository [8], the Archival Intermemory Project [7, 2], and the Stanford Archival Vault [3].

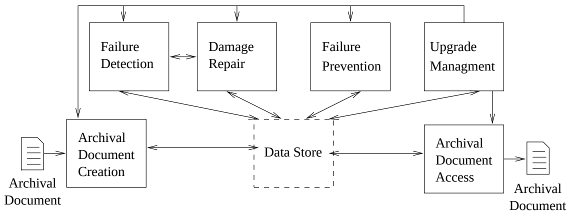


Figure 1: Archival Repository Model

The data store encompasses the set of components, such as sites, disks, or format interpreters that make materializations accessible. Because the store is not fault tolerant, materializations may be lost. A materialization is considered lost when *any* of its components has failed. If *all* of the materializations of a document are lost, then the document is considered lost.

The AS monitors materializations, and when a failure is detected, attempts to repair it. Further, the preventive maintenance module take actions to avoid failures. For example, the AS may copy components that are stored on a disk that is close to the end of its expected life, onto a newer disk. The AS may also initiate system upgrades to take advantage of newer technologies. In summary, the main functions of the AS are: document creation, document retrieval, failure detection, failure repair, preventive maintenance, and upgrade management.

3. AR COSTS

We can see the life of an AR as a sequence of events such as the failure of a disk, user access to a document, and making a copy of a document. A *cost event* is an event that has an economic impact. The definition of what has an economic impact will vary from organization to organization. For example, an organization may define economic impact as anything that has an impact on the accounting books (e.g., expenses and depreciation of capital equipment). Another organization may extend the definition to include expenses incurred by the users (e.g., expenses because of unavailability of the system). Cost events may or may not be triggered by a physical event. For example, an organization may buy a maintenance contract for disks under which an annual fee is paid in exchange for free repairs of all disks that may fail during the year. In this case, the failure of a disk (a physical event) will not have any economic impact (and thus it is not a cost event), while the annual payment for the maintenance contract (which is not an AR physical event) will be a cost event.

How can we compare the total cost of two ARs? Ideally, we would like to assign a monetary value (e.g., dollars) to each event in the sequence, and then, aggregate those costs into a single value. Having done that, we can simply choose the system with the lowest cost. If we know the sequence of costs events and each future cost can be deterministically computed (e.g., disk prices will decreased by 5% annually from current prices), this is a feasible task. In this case, we compute the monetary value of each event and we aggregate them by computing the average annual system cost, or ASC (e.g., the AR will cost \$100,000 annually).

However, as we explained in the introduction, we may not know the exact sequence of cost events. In addition, we may not know deterministically future costs and we may have to represent costs as probability distributions. In this case, the system is characterized by a probability distribution of ASCs (e.g., with probability 0.3, the annual cost will be \$100,000; with probability 0.7 it will be \$150,000). Although in simple cases the ASC distribution can found analytically, in general, we have to rely on simulations to obtain an approximation of the distribution.

With probabilistic ASCs, choosing the best AR is not straightforward. The general problem of choosing between two probability distributions of costs has been studied in [11]. In the extended version of the paper [5], we describe several ways of choosing between distributions, but in this paper, we will use the simplest way of selecting the best of two cost distributions: namely, we will choose the one with the lowest mean (average). Given this, throughout the paper, we will frequently talk about the mean annual system cost (MASC) as representative indicator for the distribution of the average annual system cost.

3.1 A Taxonomy of Cost Sources

In this section, we classify the cost sources in an Archival Repository. Our goal is to understand those sources, so we can use them as building blocks for cost events. The problem of classifying cost sources for computer systems has been studied in [6], but we are not aware of any studies for the specific case of Archival Repositories. The most common cost sources in an Archival Repository can be broken down into the following categories:

Hardware and Software: This category includes all the expenses (including lease fees) for servers, clients, disks, software, the network, and peripherals. Although, this is the most obvious source of cost for a computer system, it only represents about 20% of the total cost for the system [6]. Usually, it is easy to estimate the cost of the initial hardware and software, as we can just use market prices. However, for replacement hardware and software, this is a more complicated process as we need to predict future prices. Moreover, this prediction is often obtained in the form of a probability distribution, based on current trends, as well as possible future technological developments. For example, when predicting disk costs ten years from now, we may conclude that with 60% probability a terabyte will cost $10 or less, with 80% probability it will cost $15 or less, and with 99% probability it will cost $50 or less.

Non-labor Operational costs: This category includes all the costs (different than labor) necessary to maintain the AR operational. For example, these cost will include the electricity consumed by the system, air conditioning, and physical space. As with the Hardware-and-Software category, it is easy to estimate the initial cost of non-labor operational costs. For future costs, the major challenge in estimating theses costs is trying to predict the future needs of the components of the AR. For example, technological improvements may reduce the need for physical space, but they may increase the need for air conditioning.

Labor costs: This category includes all the human-related costs necessary for the AR. In particular, this will include management (e.g., system administrator), support (e.g., help desk), and development (e.g., application developers).

Information acquisition: Information is sometimes free (e.g., technical reports, thesis), but in general, libraries need to pay for information (e.g., journals). This payment may be a one-time fixed cost, periodic payments (subscriptions), or, more infrequently, pay per use. In some context, we may choose to ignore this cost and considered it "the cost of doing business" (i.e., the library will have to provide access to information even if they do not have an AR). However, we should consider this cost if the creation of the AR will change the way the library pays for information (for example, moving from a paper-based library to a digital library with publisher charging different amounts for paper journals than for digital journals).

Insurance: We define insurance as any agreement where an outside party takes the risk of a specific failure in an AR component in exchange of a fixed payment. An example of insurance is a maintenance contract where the library pays a fixed amount to a company that replaces failed disks. Insurance is important not only because of its direct cost, but also because it can reduce the *variance* of the AR cost. If we are able to "insure" all uncertain events, then we will have a deterministic ASC.

Unavailability: If the system is not available, there may be an economic impact for the organization. Unavailability may be caused by a system failure, but it can also occur when system resources are diverted to maintenance or repair tasks. For example, a user may be blocked because the system is checking the storage device that holds the requested document for errors. Similarly, the system may only be able to handle a fraction of the normal users when it is migrating documents to a new format.

Measuring the cost of unavailability is a difficult task. If users pay for access, we may be able to assign a direct cost corresponding to the lost income. If users to do not pay directly, we still want to penalize the system for unavailability, lest we end up with a design that disregards user needs. One way is to assume that users will access an alternate system (even if the content is not available elsewhere). We could, for instance, assume that the alternate system is equivalent to our AR. Thus, if it costs $500 per day to operate the AR, the cost of unavailability will be $500. We could also consider a commercially available alternate system. For instance, an average search on Dialog (SciSearch database) costs $6, so if we cannot satisfy say 1500 requests while doing preventive maintenance, then the additional cost will be $9000.

Cost of losing a document: Even though our objective is to preserve all documents in the repository, in some circumstances one can put a price on document loss. For example, an organization may choose between archiving certain documents or recreating them. In this case, the cost of losing the document would be the cost of recreating it. Of course, there are cases when we cannot put a dollar value on losing a document (e.g., the diary of a famous person), so we can use an arbitrarily large cost.

3.2 A Taxonomy of Cost Events

In the previous section we studied cost sources. In this section, we use those sources as building blocks for the most common AR cost events. Cost events can be broken down into the following categories.

AR creation: Starting an archival repository involves a large number of expenses. Hardware and software need to be bought, infrastructure needs to be put in place, new personnel needs to be hired, and so on. For instance, the creation of an AR with 100 disks would involve a server (about $5000), the disks ($500 per disk for a total of $50000), installation costs (one consultant at $1000), renting and furnishing an office space ($800 for the realtor that finds the place and $2000 for furniture and other necessary improvements for the rented space), and loading of the documents (five days of work supervised by a system administrator, about $1200) for a total of about $60000. The AR creation cost can be amortized over time. Amortization can done by either charging a fraction of the startup cost over fixed periods of time (in which case it would be an operational cost) or over each usage of the system (in which case it would be a document access cost).

Document Access: When accessing a document, the AR may incur acquisition costs or labor costs (e.g., the cost of the operator who retrieves and mounts a tape).

AR operation: The total operational cost of the AR would include the office space taken by the repository, the necessary utilities (electricity, network, etc.), and the cost of the people in charge of keeping the system running. For example, in San Francisco, the average cost of office space is $380 per square meter per year, so if we assume the repository occupies a small office of $8m^2$, the annual space cost will be $3040 per year. Reasonable estimates for utilities are $4000 for electricity and $3000 for network connectivity. Finally a quarter-time system administrator and a 1/8th librarian would cost about $20000 per year. This results in a total operational cost of about $30000 per year.

Failure Detection: To enhance reliability, an AR needs to periodically check for failed components (e.g., corrupted tape). When performing failure detection, we should not only take into account the cost of the detection itself (e.g., moving a tape from storage, mounting the tape on the reader, checking the tape, and returning the tape to storage), but also the cost of unavailability that it may generate. In Section 5.4 we will see an specific example of how to compute these costs.

Repairs: When the AR fails and needs to be repaired, cost events may be generated (if we do not have a maintenance contract). For example, when a hard drive fails, we may need to buy a new hard drive (about \$500), remove and install the new one (\$100 for the time of the technician), and restore the content of the failed drive into the new disk (\$200 for the network cost and the unavailability caused by the transfer).

Preventive Maintenance: Before a component fails, we may want to transfer the information to a new component. For example, if we know that tapes can survive 20 years, we may decide to copy old tapes into new ones after 10 or 15 years. The cost associated with a preventive maintenance event, includes the cost of the new media, the transfer of the information, and the possible unavailability that this task may create in the AR.

Upgrades: Upgrades are similar to preventive maintenance, i.e., we transfer information from old components to new ones. However, the motivation and the cost implications of an upgrade are different. We perform upgrades to obtain some advantage from modern technology. These advantages may go beyond improved reliability (which is the reason for preventive maintenance) and may include reduced cost. For example, when upgrading to modern hard drives, we may gain reduced operational costs (e.g., if they require less administrator time, less power, or less physical space). Therefore, after an upgrade, we need to reconsider all other costs in the system and change the cost events appropriately.

4. AR RELIABILITY

The reliability of an AR gives the likelihood that the system will work for a given period of time. Formally, the reliability is the conditional probability that no "failures" have happened in the time interval $[0, t]$ given that the system was operational at $t = 0$. There are many ways we can define an AR failure. It could be the loss of a document, the loss of a certain fraction of the collection, or even the loss of some specific set of documents. In this paper we will take the most stringent criteria: loss of any single document.

As with costs, we summarize the probability distributions for time to failure by its mean. So, we use the mean time to failure (MTTF) as representative indicator for the distribution of the time to failure. For instance, an AR with a MTTF of 100 years is expected to survive 100 years, i.e., if we build say 10 identical ARs, and average the time when each fails, we get about 100 years.

It is important to note that MTTFs can be used to compare ARs, even if we expect their configuration to change relatively soon. For instance, say we compare two ARs, A and B, using a current hardware configuration, and find that A's MTTF is 50 years, while B's is 200 years. One may be tempted to think that because the current hardware will be replaced in say 15 years, then the longer MTTFs are meaningless. However, this reasoning is incorrect. System B, with its longer MTTF, is significantly less likely to fail during the first 15 years than A and is hence preferable. In 15 years, when we change the configuration of the AR, we can re-evaluate its MTTF, and again decide what are the best options based on the predicted MTTFs at that time. In summary, MTTFs can be used to compare systems even over short periods of time.

To estimate system reliability, we need to identify the undesired events, such as the failure of a disk or an operator error, that may lead to a failure. A document is lost if the bits that represent it are lost, and also if the necessary components that give meaning to those bits are lost. An undesired event does not necessarily cause information loss. In fact, we have seen that if the AR keeps two copies of a document, and the disk holding one of the copies fails, then the document is not lost. It would take a second undesired event affecting the second copy to cause information loss.

5. DESIGNING AN AR

Our goal is not simply to evaluate a given AR, but instead to design an AR that meets our cost and reliability targets. For instance, we need to decide how many document copies to keep, what formats to store them in, how frequently to check for errors, and so on, in order to attain some desired reliability and maximum cost. To aid the design, we use a framework based on Decision Analysis [10]. We show our design framework in Figure 2. The framework is a cycle where we first formulate our objectives. Then, we identify the uncertainties (e.g., when a disk will fail). A large number of uncertainties can make the system difficult to analyze, so we next identify and eliminate the uncertainties that do not have a critical influence on the overall performance of the system. Then, we assess the probability distribution of the uncertainties and predict the performance of the AR design. Finally, we perform a sensitivity analysis to appraise our design. If we find a problem with the recommended design (for example, we discover that one of our initial assumptions is incorrect), then we iterate over the cycle.

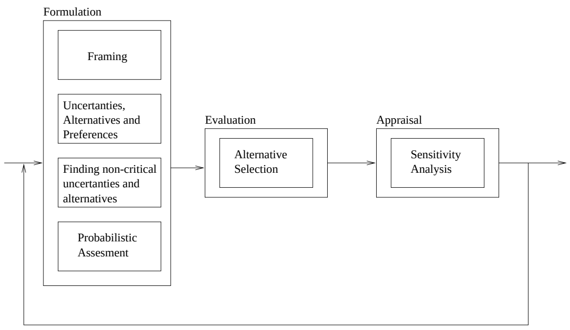


Figure 2: AR Design Cycle

To illustrate the design process, we will use a case study of a Stanford-MIT technical reports AR. At each step, we will discuss the assumptions or decisions made in this sample design. We will also present simulation results for this case study to show the types of conclusions that can be reached.

5.1 Framing the problem

The first is to clearly define the success criteria for the design. Typically, the criteria will include the archival guarantees (MTTF) and the cost of the AR (MASC). Possible goals can be: (i) Maximize the MTTF of an AR such that the MASC is less than a given amount. (ii) Minimize the MASC of an AR for a minimum MTTF. (iii) Maximize some combination of the MTTF and MASC. In other words, we want to transform the MTTF and MASC to a common metric (let us say dollars) and maximize its combination. For example, if documents can be recreated, the organization may be able to assign a dollar value to losing a document as we discussed in Section 3.1.

In our case study, the goal will be to have a repository with a MTTF at least equivalent to that of standard paper (100 years) with the minimum MASC possible. A failure is defined as the loss of one or more documents.

When designing an AR, some decisions are taken before starting the design process (policies), others are delayed until the implementation of the system (tactics), while the rest are the focus of the design (strategies). For example, in our case study we assume that the AR will cover Stanford and MIT technical reports (a policy) and that the decision on the specific brand of the hard drives that the AR will use can be defered until implementation time (a tactic). It is important for the design team to agree on which decisions are policies or tactics, as no time should be spent studying them during the design process.

5.2 Identifying Uncertainties, Alternatives, and Preferences

Uncertainties are probabilistic factors that affect the AR (e.g., the time when the disk will fail). Despite their name, we might actually have some control over an uncertainty. For instance, even though we do not know when a disk will fail, we may be able to choose between disk with different MTTFs. When we control the value of an uncertainty completely, we will call it a *variable*. For example, if we assume that disk prices will decrease exactly 5% per year (and we know the current price), then the cost of a replacement disk becomes a variable.

Alternatives are the different designs that we have available. For example, in our case study, we may consider:

- ARs with disks with MTTF of either 3, 5, 10, or 20 years.
- ARs with failure detection intervals of 30, 60, 120, or 720 days.

The combination of these different values results in 16 possible configurations that we need to evaluate.

5.3 Modeling an AR

An important decision is the level of granularity in the model. If we have too little granularity, then we will have complex uncertainties that are difficult to analyze. If we have too much granularity, the number of variables will be high, making the analysis of the model difficult and even impossible. For example, in our case study we decide to use disks, sites, and formats as the lowest level of detail (in contrast to choosing documents, files, or even bits). Thus, we only need to quantify how much money disks, sites, and formats will cost and how will they affect MTTF. This is much simpler than trying to find the MASC and MTTF of the AR as a whole (not enough granularity), or the MASC and MTTF of every single file (too much granularity).

To model and evaluate a particular AR configuration, we propose an extension of the model presented in [4]. In particular, our extension adds cost events and their associated cost distributions. Recall that our model of an AR has two major elements: a non-fault-tolerance data store and an archival system (AS) that ensures long-term survivability of the information. To model the store, we need to define:

- **AR Description**
  - Initial collection: 200,000 documents. No documents created after startup. Each document, d, will have materializations:
    - $\langle d, MIT, disk_i \rangle$,
    - $\langle d, MIT, disk_k \rangle$,
    - $\langle d, Stanford, disk_x \rangle$,
    - $\langle d, Stanford, disk_y \rangle$.

    Where MIT and $Stanford$ are the two sites; and $disk_i$, $disk_k$, $disk_x$, and $disk_y$ are different storage devices.
  - Number of components and types: 100 storage devices in each site, 2 sites.
  - Failure dependency graph: $site \rightarrow disk$, when the disk is in the given site.
- **Policies**
  - Document Creation policy: for each document, two materializations are created, one in each site.
  - Document to Materialization: read from any materialization.
  - Failure detection algorithm: complete scan of all disk. Site failure detection is instantaneous.
  - Damage Repair algorithm: discard bad component and replace with new component instantaneously.
  - Failure prevention algorithm: none
  - Upgrade policy: none
- **Distributions (unknown for now)**
  - Disk Failure dist. during access (time)
  - Disk Failure dist. during archival (time)
  - Disk Failure Detection success dist. (probability)
  - Disk Repair success dist. (probability)
  - Disk Failure Detection interval dist(time)
  - AR Creation Cost. (dollars)
  - AR Operational cost dist. (dollars)
  - Disk Failure Detection cost dist. (dollars)
  - Disk Repair cost dist. (dollars)

Figure 3: Archival Repository Model Parameters

Variable	low	base	high
Disk MTTF during access (years)	20×0.9	20	20×1.2
Disk MTTF during archival (years)	20×0.9	20	20×1.2
Success of a Failure Detection (probability)	1	1	1
Success of a Failure Repair (probability)	1	1	1
Failure Detection Interval (days)	120×1.5	120	120×0.9
AR Creation Cost (dollars)	55000	60000	70000
AR Operational Cost (dollars/year)	200	300	400
Failure Detection Cost (dollars/run)	$1000 + 1200 * 6$	$1200 + 1500 * 6$	$1400 + 1800 * 6$
Repair Cost (per replaced disk)	$450 + 100 + 164$	$500 + 100 + 204$	$600 + 100 + 244$

Figure 4: Base values

- How many component instances and types are present in the system: that is, how many disks, formats, etc., are available.
- Time distributions for component failures. Many components have two different failure distributions, one during archival and another during access. For example, a tape is more likely to fail when it is being manipulated and mounted on a reader than when it is stored. Therefore, each component may have two failure distributions: during archival and during access. For some components, such as disks or sites, the access and archival distributions will be the same.
- Time for performing a component check. This distribution describes how long it takes to discover a failure (or to determine that a component is good), from the time the check process starts. For example, consider checking a tape. This may involve getting the tape from the shelf, mounting the tape, and scanning the tape for errors.
- Time for repairing a component failure. This distribution describes how long it takes to repair a component. This distribution may be deterministic (if the component can be repaired in a fixed amount of time). Repair time may be "infinite" if the component cannot be fixed.

In addition, there is an important interdependency between components. Specifically, the failure of one component may cause the failure of another component. For example, if a site fails (e.g., because it was destroyed by a fire), then all the disks at the site will also fail. This failure dependency is captured by a directed graph. For example, an arrow between "Site A" and "Disk 1" in the interdependency graph means that if "Site A" fails, then "Disk 1" will also fail.

To model the AS we need to define:

- Document Creation algorithms and their associated cost distributions: When a new document is added to the AR, the AS uses the document creation algorithm to create enough materializations to ensure survivability of the document. This action may create one or more cost events, each with a different cost distribution.
- Document Access algorithms and their associated cost distributions: how a document request is transformed into requests for the appropriate components, and the associated costs of that operation.
- Failure Detection algorithm and their associated cost distributions: As explained earlier, the AS scans the store looking for damaged or lost materializations. When a damaged materialization is found, a damage repair algorithm is started (as described below).
- Damage Repair algorithms and their associated cost distributions: After a failure has been detected, the AS attempts to repair damaged components. There are many strategies to repair a damaged document that are discussed in [4].
- Failure Prevention policies and their associated cost distributions: The AS scans the store and takes preventive measures so materializations are less likely to be damaged. For example, the AS may copy components that are stored on a disk that is close to the end of its expected life, into a newer disk.
- Upgrade algorithms and their associated cost distributions: A technology upgrade may change the algorithms used by the AS as well as the cost distributions.

Figure 3 summarizes the AR model for our case study. The failure and cost distributions for the model and described in the next subsections. Note that for simplicity the model assumes no format or site failures. (Our methodology can of course handle a more general model.)

5.4 Transforming Non-Critical Uncertainties into Variables

We can simplify the AR analysis by considering as variables the uncertainties that have little impact on MTTF and MASC. For example, if the distribution for disk prices introduces little variation on the total cost, we might as well replace it with its mean. Eliminating uncertainties can save substantial analysis and simulation effort. We call uncertainties that have a large impact on MTTF or MASC the *critical uncertainties*. The remaining ones are called *non-critical uncertainties* or, given that we are fixing them, just *variables*. In this subsection we will see how can we identify critical and non-critical uncertainties.

To determine the impact of an uncertainty, we need to find its distribution. Obtaining an exact probability distribution for each uncertainty may take a significant effort with a limited payoff, so instead we approximate the distributions by

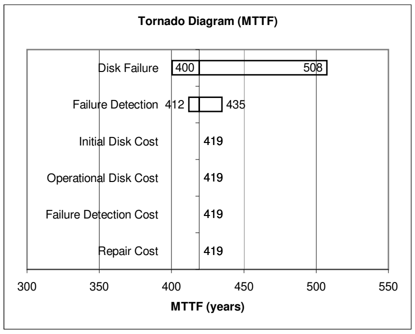


Figure 5: Tornado Diagram (MTTF)

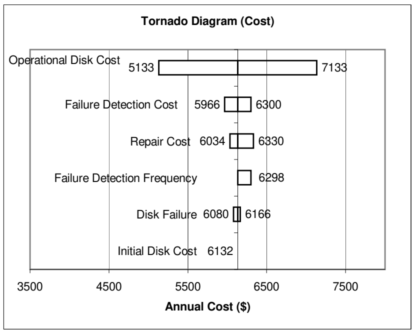


Figure 6: Tornado Diagram (Cost)

using just three values: low, base, and high which correspond to the distribution 10, 50, and 90 percentile. Finding the appropriate low, base, and high values for an uncertainty is more an art than a science. Only experience and a good understanding of the AR components allow one to make these predictions.

After approximating the distributions of the uncertainties, we assess their impact by using a Tornado Diagram. A Tornado Diagram shows the system performance (MTTF or MASC) for the low/base/high value of each uncertainty (while keeping all other uncertainties at their base values). An example of a tornado diagram can be found on Figure 5. We will explain this diagram in detail later in this section, but for now, we can see some uncertainties (such as the Disk Failure) impact MTTF significantly while others (such as Failure Detection Cost) have little or no impact.

Returning to our case study, let us consider the case where disks have a MTTF of 20 years and we scan the repository every 120 days. (In practice we would do a similar evaluation for each of the other 15 alternatives discussed in Section 5.2). First, we obtain a *rough* range for the values of the variables. These ranges are shown in Figure 4. The choice of these values is highly subjective, but, nevertheless, we will attempt to describe the rationale that an expert may have followed to reach these values.

Disk failure during access and archival: For these two uncertainties, we choose to have the same distributions, since disks do not fail significantly more when accessed. In this case, we use as base value the MTTF advertised by the manufacturer (20 years). We assume that there is little variation in MTTF, so we will assign a low value of 90% of the advertised MTTF and a high value of 120% of the advertised MTTF.

Success of failure detection and a repair: For these two uncertainties, we assume that the probability of success is 1. In other words, we are assuming that there are no hidden failures (i.e., if a disk is defective, we can always tell) and that all repairs are successful (i.e., we can always replace a defective disk with a working one). Note that the later does not mean that we can always repair a document. It just means that we are always able to install a new disk and to copy the content of the failed disk from alternative sources *if it is available.*

Failure Detection Interval: This uncertainty shows that the assignment of low, base, and high valued need not be symmetrical. For instance, we assigned a low value of 1.5 times the targeted mean time to detection (120 days), while we assigned a high value of 0.9 times the detection time. In other words, it is more likely that detection will be slower rather than faster.

AR Creation Cost: Using the rationale presented in Section 3.2, we will estimate the initial AR cost to be $60000. To allow for error, we will choose a low value of $55000 and a high value of $70000. We will assume that this will be a one-time cost (i.e., no amortization will be done over time).

Operational Cost of a Disk: As illustrated in Section 3.2, we use a total operational cost of about $30000 per year with a low value of $20000 and a high value of $40000 per year to allow for errors. This total operational cost divided by the number of disks (100 per site) results in a operational cost per disk of $200 to $400.

Cost of the Disk Failure Detection Algorithm: This is probably the hardest uncertainty to estimate. We divide the cost of the detection algorithm in two components. The first component represents the direct cost of running the algorithm, while the second component reflect the cost of service unavailability. The direct cost of the failure detection algorithm includes the time required by the System Administrator to start the scan and correct any problems with the scan (assuming these tasks are not included in the administrator's salary already). If we assume that failure detection involves 5 days of part-time work, the cost will be about $1200 per run (see Section 3.2).

To estimate the unavailability cost, we will assume that users will use an alternate commercial service. Using the costs of Section 3.2 for 1500 missed user requests, we price unavailability at $9000. Therefore, the total cost of running the failure detection algorithm is about $10200. To allow for error, we will choose a low value of $8200 and a high value of $12200 per run.

Cost of the Disk Repair: The repair cost is equal to the cost of adding a new disk ($450 to $600) plus the cost of removing the disk ($100) and a fixed amount for the resources involved in copying the data from the alternate sources onto the new disk (equal to twice the cost of running the detec-

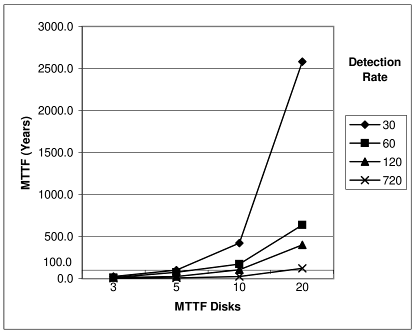


Figure 7: MTTF for base values

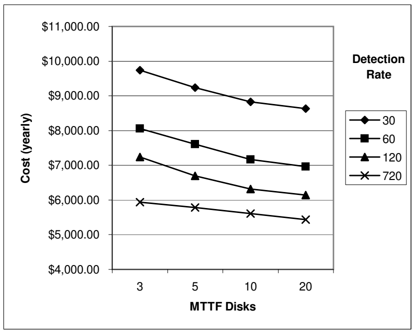


Figure 8: MASC for Base Values

tion algorithm on a single disk, this is, for the base cost, \$10200/100 $*$ 2 or \$204).

We are now ready to generate the Tornado Diagrams. We use ArchSim/C to simulate the performance of the system. At this stage, we do not want to run the full fledged simulations (which may take a significant amount of time). Instead, we run *fast* simulations with broad confidence intervals (requiring fewer repetitions) and considering all uncertainties, except disk failures, to be deterministically fixed at their base values. Fixing the value of the variables speeds up the simulation as we do not need to compute a random value for each event and allows us to group events. For instance, instead of generating a random value for each repair cost, we just count the number of repairs that were performed during the simulation and multiply by the fixed cost of making a repair. We treat disk failures differently because deterministic failure times would cause all disks to fail at the same time (and all data would be lost).

To generate the Tornado diagrams, we evaluate the AR reliability and cost for the proposed design with all the variables at their base values. Then, we modify each variable independently (while keeping all others at their base value) to its high and low value and evaluate performance again. Each tornado diagram summarizes $1 + 2 \times$ variables simulations (one simulation for the base case and two for each variable). In our case, this results in a total of 13 simulation per tornado diagram. We show the result of our simulations in Figure 5. We can see that most of the MTTF variation (95.7%) comes from the disk MTTF variation. Therefore, with respect to this metric, we can safely assume that the other uncertainties are noncritical and can be fixed at their base values.

Figure 6 shows the equivalent diagram for costs. In this case, 94% of the cost variation is produced by the operational cost of the disks. Therefore, with respect to this metric, we can safely assume that the other uncertainties are non critical and can be fixed at their base values.

In conclusion, we only need to consider disk failures and the disk operational cost as critical uncertainties, for the case of a design with disks having a MTTF of 20 years and failure detection interval of 120 days. To complete the analysis, we need to repeat the process with the other 15 configurations. Although we do not show the results for the other cases, the conclusion is the same: only disk MTTF and cost are critical. (In general, the conclusions could vary from scenario to scenario, but this does not occur in our case study.)

5.5 Eliminating Futile Alternatives

Let us turn our attention to the available alternatives. For that purpose, we used ArchSim/C again to run fast simulations. The results are in the graphs of Figure 7 and 8. From the graphs, we can see that a detection interval of 720 days never achieves our required minimum of a MTTF of 100 years (it barely achieves it for a disks with MTTF of 20 years, but the 90% confidence interval includes values below 100 years). Similarly, disks with MTTF of 3 years also never achieve our required minimum MTTF. Note that these results are based on fast simulation where all uncertainties, except the MTTF of disks, are fixed. If we are very aggressive and eliminate too many alternatives, we might eliminate the alternative that may happen to be the best when running the full simulation. On the other hand, by eliminating some alternatives, the time to run the full-fledged simulations later is reduced. For this case study, we not consider further disks with a MTTF of 3 years or detection time of 720 days. If we were more aggressive, we could have also eliminated disks with MTTF of 5 years (except when the detection interval is 30 days).

Regarding MASCs, the preliminary analysis shows a surprising result. The MASC of an AR with costly, but more reliable, disks ends up lower than that of an AR with the cheap, less reliable, disks. This is because of the cost of buying a new disk (when the cheap disk fails) and transferring the information to it. Therefore, we drop disks with a MTTF of 3 years, and detection intervals of 720 days, and reduce our alternatives from the original 16 to just 9.

5.6 Probabilistic Assessment of Uncertainties

For our final analysis we may need to assess the probability distributions of uncertainties more precisely. In practice, we will rely on experts to produce these distributions. Techniques for probability distribution elicitation are described in [12].

To illustrate, in our case study, we model disk failures with an "infant mortality" distribution. This kind of distribution, typical for electronic devices, has two phases. First, when most manufacturing defects will cause a failure, the probability of failure is high, but drops sharply over time. In

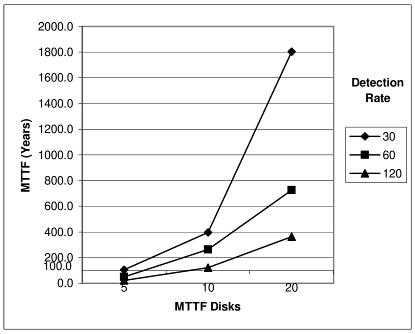


Figure 9: MTTF Evaluation

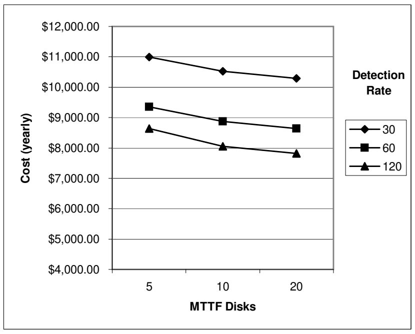


Figure 10: Cost Evaluation

the second phase, the probability of failure is constant. To model this distribution we use three parameters: time span for the first phase, percentage of devices failing in the first phase, and the probability of failure in the second phase. For our disks, we use a distribution where 10% of the disks fail within the first 30 days (i.e., an exponential distribution with mean 285 days) and, after that disks fail following an exponential distribution with mean 20 years.

To model the operational cost of disks, we assume that the library will sign one-year maintenance contracts. Although the price is fixed for one year, from year to year, the price specified in the contract may change due to market conditions. We will use an uniform distribution between $200 and $400 per disk to capture those market fluctuations. Using these more complex distributions makes our predictions more accurate, but also makes evaluation much harder. Fortunately, ArchSim/C can handle such general distributions.

5.7 Evaluating an AR Design

ArchSim/C receives as input an AR model (including costs), a stop condition (e.g., stop when the first document), a simulation time unit (minutes, hours, days, etc.), and the number of repetitions. ArchSim/C outputs the mean time to failure (mean time to stop condition), a cost metric, and a confidence interval for both the MTTF and the MASC.

ArchSim/C follows the structure of a traditional simulation tool. Each component of the AR model registers future events in a timeline. For example, when a disk is created, the simulation uses the disk failure distribution to compute when the disk will fail; then, it registers the future failure event in the timeline. The simulation engine advances time by calling the module that registered the first event. This module may change the state of the repository and register more events in the timeline. Additionally, the module may contact the Cost Manager and record some cost involved with its operation. After the module returns, the simulation engine checks for the stop condition and, if it has not happened, it advances to the next event, in chronological order. If the stop condition has occurred, the simulation stops and records the point on the timeline when this happened and the total cost incurred up to that time. The engine keeps re-running the simulation until the number of repetitions requested by the user is reached. At that moment, ArchSim/C computes the MTTF and the MASC by averaging the recorded time to failure and costs at the end of each repetition. ArchSim/C also compute the confidence interval for those values. Further details on ArchSim/C, including techniques and features that speed up significantly the simulation can be found in [4].

In the previous sections we concluded that the most promising alternatives were the ones with disks with a MTTF of 5 to 20 years and a detection/repair interval of 30 to 120 days. We also concluded that we would consider the MTTF of the disks and the yearly operational cost of the disks as critical uncertainties and the rest, as variables. Using this setup, we use ArchSim/C to fully simulate the AR and obtain its reliability and cost.

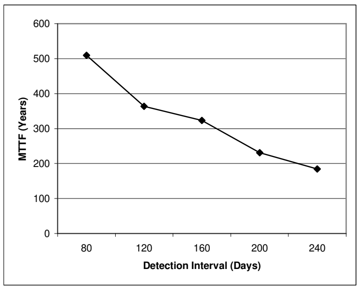
In Figure 9 we see the MTTF of the AR for different configurations. From the figure, we conclude that we need disks with a MTTF of either 10 or 20 years and detection intervals of 30 to 120 days to achieve our target MTTF of 100 years.

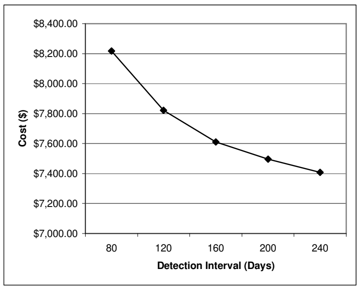
In Figure 10 we see that the least expensive alternative is the one with a detection interval of 120 days. Consistently with the preliminary simulation, here again the cost decreases when using more reliable disks. Therefore, the best alternative is one that uses disks with MTTF of 20 years and has a detection interval of 120 days. Such an AR will have a MASC of $7,822 and a MTTF of 364 years. Note the critical role that costs played in reaching this decision: if we had ignored costs we could have easily selected a design that achieves the desired MTTF in a much more expensive way!

5.8 Appraising Cost Decisions

In this final phase, we revisit our assumptions by running sensitivity analysis of the critical and non-critical uncertainties. We again illustrate the process via our case study. Recall that our proposed design was an AR with disk with a MTTF of 20 years and a detection interval of 120 days. Using ArchSim/C we ran sensitivity analyses for all uncertainties, but due to space limitations, we only present the results for two: the Detection Interval (a critical uncertainty) and the cost of detecting failures (a non critical uncertainty).

In Figure 11 we perform a sensitivity analysis for the detection interval. We want to find out the impact of a small change in the suggested 120-day interval. In the figure, we can see that the smaller the detection interval, the higher the MTTF of the AR. In particular, an AR with a detection interval of 240 days will have a MTTF of 184 years. So, we





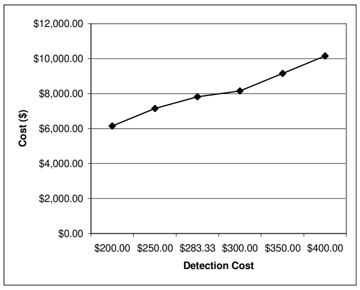


Figure 11: MTTF Sensitivity Analysis (Detection Interval)

Figure 12: MASC Sensitivity Analysis (Detection Interval)

Figure 13: MASC Sensitivity Analysis (Detection Cost)

can double the value of the detection interval and we still achieve the target archival guarantees. If doubling the value of the detection interval had caused an important decrease in cost, then we would need to reassess our recommendation.

In Figure 12 we see the AR cost for different detection intervals. As expected, larger detection intervals decrease costs. For instance, increasing the interval to 240 days, causes a reduction of cost of $400 or about 6%. We now have to decide if it is worth considering new alternatives given a potential saving of $400. If this is the case, we should return to the formulation phase and add alternatives with detection intervals in the 120 to 720 days range. Notice that we cannot make a new recommendation based only on the sensitivity analysis, because the variable we want to change may interact in unexpected ways with the reliability and cost metrics. Concretely, in this case, the cost associated with the detection interval might not be a continuous function, so we may need to revisit our cost estimates and re-run the simulations.

Let us turn now our attention to the sensitivity analysis of the cost of detecting failures. This variable has a different nature than the detection interval, as it does not affect the MTTF of the AR. Additionally, we may not be able to change the value of this variable (the cost may be fixed by the market), so a sensitivity analysis here rather than validating or invalidating our proposal, gives us an idea of how much cost may increase (or decrease) if our estimate of the value of this variable was erroneous.

In Figure 13 shows AR cost for different detection costs. As expected, the figure shows higher costs when the detection cost increases. The important observation here is that costs are increasing almost linearly with a very small slope. An increase of 100% in the detection cost (from 200 to 400), only results in an increase of 33% in the AR cost. This means that a small error in the estimate of the detection cost will not affect the AR much. Unfortunately, it also means that efforts in reducing the detection cost will have small payoffs.

6. CONCLUSIONS

In this paper we have studied how to make cost-driven decisions about archival repositories. We presented a framework that improves the efficiency and effectiveness of the AR design process. We described a powerful simulation tool, ArchSim/C, for evaluating the reliability and cost of ARs and the available archival strategies. We described how ArchSim/C can efficiently perform large simulations involving many components and very long simulated periods. We believe our design framework and ArchSim/C can help librarians and computer scientists make rational and economical decisions about preservation, and help achieve better archival repositories.

7. REFERENCES

[1] C. Borgman, S. Chen, H. Garcia-Monlina, K. Thibodeau, , and G. Wiederhold. *NSF Workshop on Data Archival and Information Preservation.* National Science Foundation, March 1999. At http://cecssrv1.cecs.missouri.edu/NSFWorkshop/.

[2] Y. Chen, J. Edler, A. Goldberg, A. Gottlieb, S. Sobti, and P. Yianilos. A prototype implementation of archival intermemory. In *Proceedings of the Fourth ACM International Conference on Digital Libraries*, 1999.

[3] B. Cooper, A. Crespo, and H. Garcia-Molina. Implementing a reliable digital object archive, 1999. Submitted for publication to ACM DL 2000.

[4] A. Crespo and H. Garcia-Molina. Modeling archival repositories for digital libraries. In *Proceedings of the Fourth European Conference on Research and Advanced Technology for Digital Libaries (ECDL)*, 2000.

[5] A. Crespo and H. Garcia-Molina. Taking cost-driven decisions about archival repositories. Technical report, Stanford University. At http://www-db.stanford.edu/ crespo/papers, 2001.

[6] Gartner Consulting. *TCO Analyst. A White Paperon GartnerGroupś Next Generation Total Cost of Ownership Methodology*, 1997.

[7] A. Goldberg and P. Yianilos. Towards an archival intermemory. In *Advances in Digital Libraries*, 1998.

[8] J. Halpern and C. Lagoze. The Computing Research Repository: Promoting the rapid dissemination and archiving of computer science research. In *Proceedings of the Fourth ACM International Conference on Digital Libraries*, August 1999.

[9] R. Howard. The science of decision-making. In *Readings on the Principles and Applications of Decision Analysis*, volume 1, 1964.

[10] R. Howard. Decision analysis: Practice and promise. In *Management Science*, volume 34, 1988.

[11] H. Levy. *Stochastic Dominance: Investment Decision Making under Uncertainty.* Kluwer Academic Publishers, 1998.

[12] C. S. von Holstein. Assessment and evaluation of subjective probability distributions. Technical report, Economic Research Institute, Stockholm School of Economics, 1970.