To: Distribution

From: D. R. Vinograd

Date: 06/26/75

Subject: Attributes of a Good Backup System

The purpose of this MTB is to define the attributes of a well designed backup system and to point out what qualities or functions should be optimized. To do this, one must first define the role of a backup system in a system such as Multics. After that, one can specify its desirable qualities.

THE ROLE OF BACKUP

The backup system functions as an insurance policy to protect the data that the users place into the storage system hierarchy. It provides this insurance by creating copies of data objects, either as they are modified, referred to as incremental backup, or as part of a large logical set , referred to as complete or static backup. In the event of data loss, backup provides a means to recover copies of the orignal data objects which may have been damaged or destroyed. If the system were never to lose data because of a failure, or a user never deleted or destroyed information, or the hardware never failed, then backup would be a waste of system resources. But all of the above do happen, and more then once, so unless a site wants to run a risk of unrecoverable data loss (a strange form of self-insurance) then some form of backup must operate.

FORMS OF BACKUP

Backup has taken many forms in other computer systems. One approach is that of physical copy of the storage system taken on some scheduled basis, usually once per day. Should an accident happen, the lost data can be recovered from the copy, although modifications and additions made after the copy will be lost. Another approach is for the operating system to do nothing. If individual users desire to backup their data then they must create and manage their own copies.

To forestall user mistakes some systems have implemented delete commands which do not really delete a data object, but place it into "limbo". If the user realizes soon enough that he has made a mistake it then can be "undeleted".

To protect against the failure of a physical storage unit, some systems allow the user to specify that two copies of his data objects exist on different physical storage units. Modifications to the primary data object are automaticaly made by the system to the copy.

To protect against program failure some systems allow a user to specify that a data object will have 'shadow' copies which are not updated with changes made to the primary copy until a specific command is given.

To protect large valuable data bases all of the above techniques, plus the recording of each modification request and/or result, are used.

BACKUP ATTRIBUTES

The following paragraphs describe in reasonable detail some of the attributes that a well designed backup system should have.

Backup is a system function. It is the system's responsibility to specify and control how the backup system should operate, the data formats that will be used, and the interface that should be presented to the user and to site operations. The specification should be general enough such that users are not prompted to construct their own backup mechanisms, yet not so general that the backup system can be misused for other purposes that are not related to data preservation and recovery. If this is not done, and users were allowed to control or specify the operation of backup, then system recovery after a failure might be impossible because some data necessary to the recovery sequence had not been dumped or some new data format had been introduced for dumping that the recovery procedures could not understand.

Backup should operate invisibly to the normal user. A user should not have to perform any operation or command sequence to enable the backup of his data objects. Nor should the user be able to inadvertently disable the backing up of his data objects. This does not preclude user control of whether a data object should be dumped. This aspect will be discussed later.

The media on which backup writes should be physically removeable from the computer site. If user data objects are to be protected against all possible accidents, the media they are contained on must be removable to a place of safety, such as a vault.

Backup should be reasonably independent of the structure, or operation, of the hardware or media used to create and store the copies it produces. The types of removable media and the hardware operational modes used today can and will change with technological improvments. If backup is intergrated too closely with a specific type or kind of media operation then its maintainability or performance may suffer.

The logical structure of the backup media must be error resistant to both logical and physical errors. It must be less complicated then the structure being protected, else it would require its own backup system. The logical media structure should be designed for quick and easy resynchronization with minimum information loss if a portion of the media is unreadable because of physical error. If the recovery sequence is dependent on an ordered series of events then the media structure should provide information so that checks can be made to insure that the order is correct.

Backup must operate at a reasonable cost to the system. Its cost should be measured in terms of CPU and I/O resources and the media it uses. Site operations must have the means to control the level of system resources allocated to the production of data object copies. Exercising this control should not cause future recovery problems. A very high percentage of the copies created will never be used. The resource cost to produce these copies must be kept very low else the cost to use one of the copies will be astrononical. Backup produces multiple copies of a data object as the data object is modified over time. If recovery is necessary, the user is normally interested in the most recent copy. The older copies can be disregarded. This requires a sort-merge operation which will produce a set of the most recent copies of all data objects that were previously produced. This media compaction operation should be independent of backup. Backup should incrementally dump only those data objects which have been modified. It should not cause excessive I/O traffic nor expose system data bases to unnecessary risk by heavy paging.

Backup is a system function and its operation must be observable. Its operational modes should be heavily enough metered so that system administrators can determine how the resources used are distributed. The metering should be per invocation as well as cumulative. Post processing tools should be provided as needed.

Backup should only dump consistent data objects. If a data object is not consistent then it is a waste of time to dump it since its recovery will be of no benefit to the user and may be more confusing and time consuming then no recovery at all. The user, who is the best judge of the matter, must be able to inform backup of the consistency state of a data object. Backup must also recover only consistent data objects. If the data object was consistent when dumped then the dump image must be recovered without any media errors. If this cannot be done then, in most cases, the data object should not be recovered. In certain cases

it may be necessary to recover whatever can be recovered regardless of data loss. In these cases the data object(s) recovered should be flagged as inconsistent.

Backup should flag the data objects it recovers so that when an attempt is made to use the recovered data object the user is informed. The user can then determine if the recovered data object is useful before proceding further. If this is not done the user may find out later, to his dismay, that the data object recovered, while consistent, is out of date and has produced or influenced the production of some erroneous results.

Backup should provide an easy to use interface to the user who wants to recover replacements for a small number of damaged or destroyed data objects. The user should not have to know how the backup system operates nor should he have to search through summary records of all the data object processed to determine the media identifier where his data objects are stored. This would be a security violation since he would see the names of data objects that he has no right to know about. The recovery should not affect nor be affected by any work the user may have done after the data loss but before the start of recovery. The recovery time should be minimal so that user inconvenience is low. If the user cannot do productive work then, as far as he is concerned, the system is not available. As a result, his revenue contribution is lost and his antagonism to the system will increase. If he encounters this situation too many times he may discontinue his use of the system.

Backup should maintain a record of the copies it creates. These records should be compact so the online storage costs will be minimal. There should be a way for any user, with the correct permission, to interrogate these records in a controlled manner and use the information to start a recovery sequence in much the same way he might queue a print request. The records should be complete enough to produce reports such as:

a daily summary of all data objects dumped so that a site administrator could derive a picture of backup's activity and resource utilization,

a table of contents of a specified media unit so that if a unit is lost or damaged the loss can be determined,

a table of media identifiers of all copies which belong to some user or are part of some defined set so that a complete recovery of that set of data objects can be quickly accomplished.

Backup should be logically independent of the internal structure of the copies that it produces. The copies of data objects should be physical copies not encoded information which can later be use to recreate the data object. This will allow changes to

the internal structure of any data object that must be copied without requiring any changes to backup. If backup were to encode the copies of data objects then any changes to the data objects would have to be reflected in the encoding and decoding operations of the backup system.

The recovery of data objects should have a minimal impact on the system operation. Even in the extreme case where the hierarchy itself must be rebuilt, once this is done the system should be immediately available. Users should not have to wait until the entire storage system is intact.

Backup is a system service and it must be able to copy data objects from all parts of the hierarchy when requested to do so. The system must provide backup with a well defined means to access the data objects that must be copied. This method should be invisible to the user and should not be thwartable by inadvertent user action.

The organization and structure of the system that is to be backed up is very important. If the system and user data objects are randomly distributed throughout the physical storage system then the loss of any physical unit will damage or destroy a random collection of system and user data. This type of loss will cause non-predictable results and usually will result in the reconstruction of the entire hierarchy. This is because in this random type of organization it is usually impossible to tell which data objects have been lost or damaged. If system and user data objects are completely separated and the user data objects are organized in a manner which provides a convenient mapping between some logical organization and a physical storage unit, then the loss of a single storage unit will only destroy a specific set of data objects. Since there will usually be more user than system data objects, and thus more user than system storage units, the random failure of a storage unit will more likely affect a small group of users rather then the entire system.

OPTIMAL ATTRIBUTES

All of the above attributes are important to the design of a good backup system but some are more important then others. If we can identify this smaller set and structure the design of the backup system to optimize these attributes, then the resultant backup system will be that much more useful. To do this let us go back to the premise stated earlier that backup is a data insurance policy for the system and as such must protect it. The question is, what kind of protection is desired.

A very good measure of the usefulness of a system is the amount of user usable computational time available. This is true both

from the users viewpoint of accomplishing some task and from the systems viewpoint of selling enough resources to economically justify its own existence. If the amount of useful computation time is low, then the system loses revenue and the users accomplish little useful work. Another way to state this is that all parties concerned want to maximize useful system up time. This line of reasoning leads to the conclusion that the most important attribute of a good backup design is to minimize the service interruption time for the greatest number of people whenever it is necessary to invoke a data recovery sequence. Restated this means that the primary design goal of backup is to make the system available for useful work to the greatest number of users in the shortest time period, should data loss occur. The achievement of this goal may require some different design approaches in the system and special system interfaces for the exclusive use of backup. In fact, if this attribute is not recognized early in the design and specification of the system, backup may not be able to achieve this goal. Care must be exercised as to what the phrase "available for useful work" means. If, for example, the primary reason for the existence of the system is to maintain some data base and that data base is destroyed then no useful work can be accomplished by the user community until the data base is recovered even if the system is operational. If the information loss affects only a small portion of the user community then the majority of the users can accomplish useful work. This is a trade-off situation since if the system is made available to users then the real-time recovery of the remaining lost data may be slower, since backup may have to compete with other users for resources. This decision must be left to the site management.

A less important objective than minimal recovery time is to minimize the cost of the insurance protection. If the cost of protection is too resource-expensive, sites will skimp on its use. Then when recovery is necessary the results will be below their expectations. The goal is to keep the per-copy cost of backup very low. The copies produced must be directly usable in the recovery sequence without slowing it down. There is a trade-off between the cost of creating incremental copies of data objects and the cost of lost information because the data object copies lag behind the real world objects. For some applications this time can be fairly long; for example a relatively static data base where the updates can be repeated. For others, such as an airline reservation system, the time must be very short. Being able to recover close to the time of the failure may also increase the recovery time.

If these two aspects of a backup system design are emphasized, the minimization of recovery time and the minimization of system resources to produce copies of data objects, then the resultant system will provide the protection that the system requires at a price it can afford. Its interface to a single user or operations can be crude, its record keeping non-existent, but its primary

function as an inexpensive data insurance policy will be fulfilled.