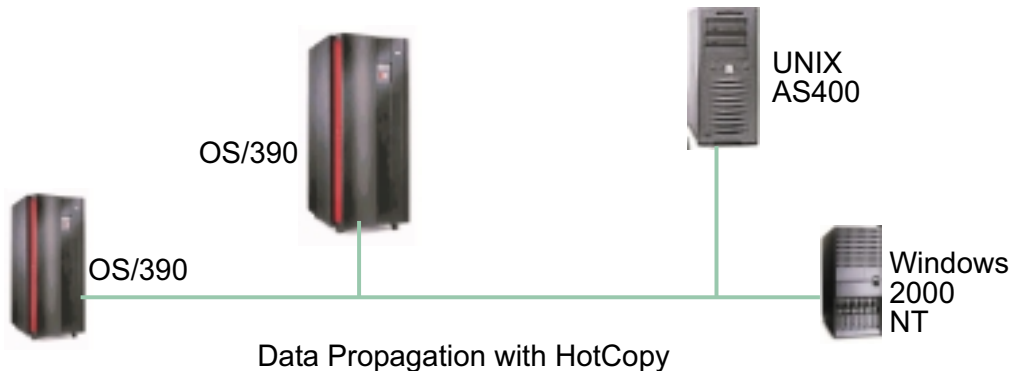


HotCopy



HotCopy is a system designed to automatically supply heterogeneous target environments with data from central sources. This way of supplying data we call *propagation*.

The basic concept of HotCopy is that data only need to be loaded once into a target database and thereafter only the incremental changes (*deltas*) need be added in chosen intervals.

In the following we refer to the application where the data are created as the *source application* and the data it creates as *source data*. The receiving application is referred to as the *target application* and its data are the *target data*. The process of propagation means diverting a copy of selected data from the source application and directing it to the target application for processing.

There are currently various methods to transfer data from one application to another and each has its advantages and disadvantages. Unlike most ETL tools HotCopy does not extract source data after processing but instead observes the change process and notifies the change to the target database.

Methods currently available

Currently available methods for distributed data can be described as follows:

1. A regular **copy of the complete source data** and recreation of the target data, for instance, every night everything is copied.
 - Depending on the size of the source database a considerable amount of data will have to be moved. This disadvantage increases proportionately to the distance of the target application.
 - The target data will not be right up-to-date since the amount will restrict frequent transfers.
 - If the transfer is interrupted the complete process must be repeated (lack of recovery).
 - Without a suitable extraction tool the complete database will have to be transferred, even if only certain fields are necessary.
2. Regular **transfer of the changed data** and update of the target data after an initial load.
 - The amount of data to be transferred will be minimized.
 - How up-to-date the data in the target environment are, depends on the frequency of transfers.
 - One problem is, how to recognize changes. Usually this will mean the introduction of update flags into the source data.
 - Without a suitable extraction tool reprogramming will be required to support the update selection logic.

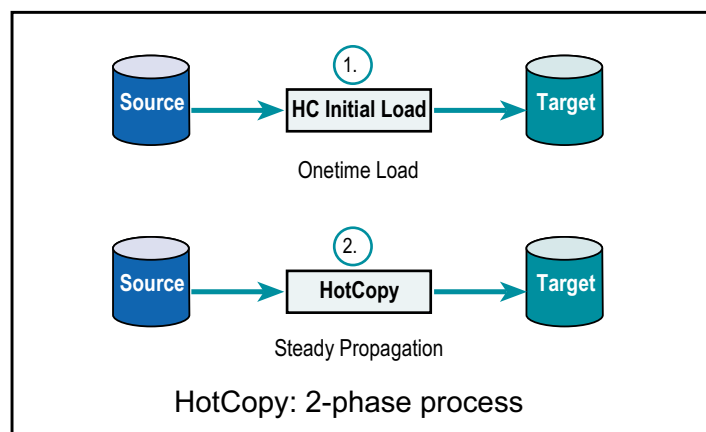
3. **Reprogramming the source application to transfer the updates** directly to the target.
 - Data selection can be fitted match requirements exactly.
 - The transfer of data to the target is almost real-time.
 - Demands significant programming effort: risks when changing an existing application and, of course, constant maintenance.
 - Performance penalties in the source application can hardly be avoided since during a synchronous transfer the source application has to wait on confirmation from the target. (Commit may be made only when the transaction is complete.)
4. **Using so called "triggers".**
 - Triggers are available only in certain environments (DB2, Oracle)
 - A trigger is a form of synchronous transfer and therefore slows down the source application. Triggers between two DB2 systems, for instance, will have a significant negative impact on the performance of both systems.

How does HotCopy work?

The HotCopy process guarantees current status of target data and avoids the above mentioned disadvantages of poor performance, extensive reprogramming, constant maintenance and repeated, redundant transfers of mass-data.

For performance HotCopy works asynchronously but ensures that only consistent (committed) logical data are propagated.

The first step is a **HotCopy initial load**. Here the desired attributes/fields of the source data are carried over to the target. After this one-time initial load the permanently active HotCopy propagation process, the **HotCopy** task, is started. HotCopy now ensures that relevant changes in the source data which occur after the initial load will be passed on to the target database. The user specifies the relevant fields of a record (VSAM) or of a segment (IMS) or columns/attributes of a table (DB2) for the initial load and for HotCopy as so called **HotCopy objects**.



Technically HotCopy consists of:

- Various data Recording facilities to recognize a change of data in its respective environment, IMS, CICS or Batch, and to report it to the HotCopy started task (HCP).
- The HotCopy Started Task (HCP) which collects the data from the data Recording facilities in Dataspaces, selects where appropriate and transfers them to the targets.
- A data transfer facility which transfers consolidated change data to the target application and updates it.
- An ISPF user interface to control HotCopy and maintain the HotCopy database.
- The HotCopy database (CDB) which contains all data structures (HotCopy objects) to be propagated by HotCopy.

The following source and target database formats are supported.

Source formats:

- DB2 databases on OS/390.
- IMS databases
- VSAM datasets.

- Sequential files.
- The above under IMS, CICS and in batch.

Targets:

- DB2 databases for OS/390.
- Any SQL database system, e.g. Oracle, UDB, MySQLm DB2/400.

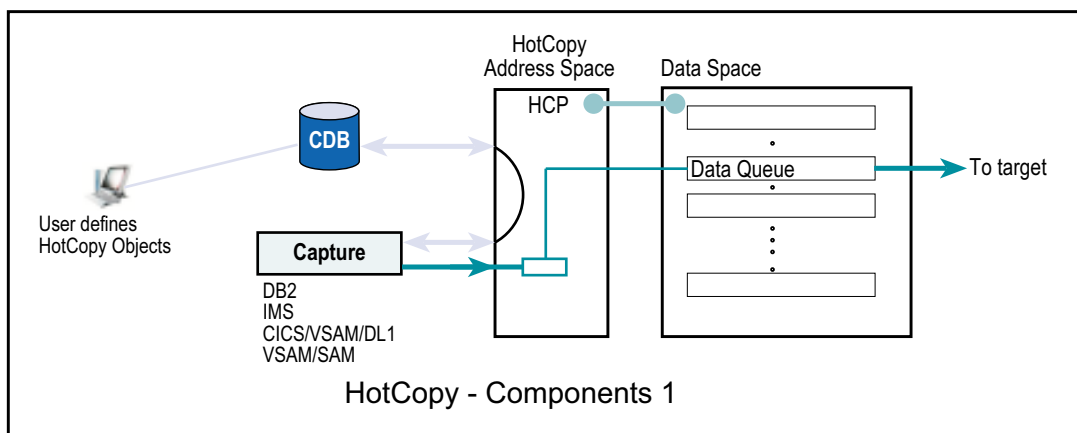
HotCopy Objects

With the help of an ISPF menu-controlled the user defines the tables, database segments and files to be controlled and the data to be propagated. These definitions are known as **HotCopy objects**. A HotCopy object consists of:

- A unique name,
- A description of the source data, in this case the complete record or segment layout of the database concerned. This description can have a different form, dependent on the original database. Thus in the case of DB2 it will be taken from SYSTABLES. In the case of IMS we would use the IMS internal DBDs and PSBs or alternatively the COBOL/Assembler copy books or PL/1 includes.
- A definition of the fields/attributes to be selected. The user selects these from a panel which shows all fields of the database.
- A specific key for the target table in which the user marks the corresponding data fields as key fields.
- Definition of the conditions which trigger propagation – it is possible make the propagation dependent on complex conditions based on the contents of multiple fields.
- Clauses to control handling of non-relational originating elements – in COBOL or PL/1 there are often data structures which are simply not relational: REDEFINE arrays, OCCURS DEPENDING and the like. HotCopy is capable of propagating such structures into chained relational tables.
- Definition of the target system for propagation: TCP/IP address or name, port, database, table name, etc.

HotCopy objects are maintained in the **HotCopy Database (CDB)**. Before HotCopy can start propagation of a HotCopy object, a HotCopy initial load must be run. The HotCopy initial load uses the information stored in the affected object to determine source, target and transformation required to create a one-time copy. This will also be required if the HotCopy propagation process is interrupted.

Note: It must be ensured that the source database is not active during the initial load process: Stop Database (IMS, DB2), CEDA LOCK (CICS), etc.



Finally the HotCopy initial load sets a LOADED flag in the relevant HotCopy object. This signals the start of the propagation process to HotCopy. When started, the HotCopy task (HCP) will commence propagation for all objects in the CDB which have the LOADED flag set.

HotCopy – the Propagation Process

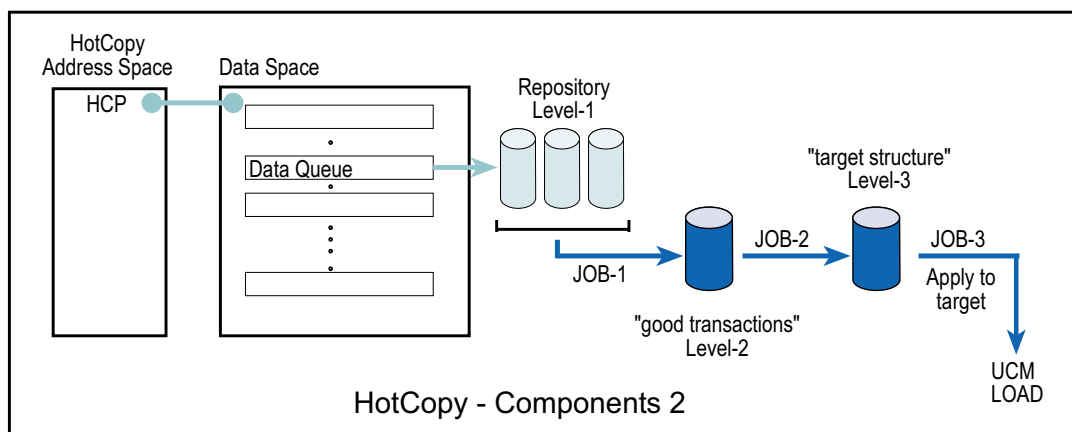
After creation of the target database HotCopy will ensure that all relevant changes which are made in the source data are also mirrored in the target. The HotCopy components work together as follows:

- HotCopy's Recording facility sends messages of changes (INSERT, DELETE, UPDATE, CHECKPOINT, COMMIT) by way of Cross Memory Call (XMS) to the address space of the HotCopy Started Task (HCP). Depending on the environment the Recording facility agents are placed in various interfaces:
 - ✓ DB2 Logger Exit or Archived Log
 - ✓ IMS Language Interface
 - ✓ VSAM/SAM Subsystem Interface
 - ✓ CICS File Control Interface (VSAM), DL/I Interface
- The HotCopy Started Task (HCP) manages the delta changes reported by the Recording agents in a data space. For each active HotCopy object there is a data queue reserved therein. The data space is persistent, that is, it will survive a restart of the HotCopy address space or a restart of the operating system.
- When a data queue reaches its threshold at the latest, its contents will be output from the data space into a file (Level-1 repository). This migration happens asynchronously to the other HotCopy processes. This file, whose contents we refer to as **raw change data**, is a HotCopy object and therefore belongs to a unique data queue. Its name, generated automatically, includes the HotCopy object name along with the system name and a timestamp. The raw change data contains all changes made to the monitored object during a space of time. For performance reasons we do not take account of logical integrity at this stage but only record a 1:1 snapshot of the data movement.
- After creation of the Level-1 repository a consolidation into a Level-2 repository is made. This occurs when the Level-1 repository has achieved a predefined size. In this case a unique name consisting of HotCopy object, consolidation level, system name and timestamp is also generated. The contents of the second level repository are known as **consolidated change data**. This file contains only committed transaction data. Transactions which are not yet committed will be output to the raw change data.
- The next step, during the transfer from Level 2 to Level 3 is the restructuring of the data to match the target structure based on the details in the HotCopy object. While the record structure in Level-2 still corresponds to the source record, that of Level-3 corresponds to the target structure(s).

Further processing depends on the target application.

Supporting the Target System

HCP saves data by HotCopy object first in its data queues within a data space. After a predefined interval



or on an operator/administrator command, a CLOSE event may be triggered for any one of the data queues. This causes the system to seek a logical termination point, that is, only data from currently running transactions will be taken into the Level-1 repository, data from newly started transactions will remain in the data space for the time being.

As soon as this activity has completed, a job will be started which turns the Level-1 repository into a level-2 repository, by sorting out the irrelevant transactions. The data are still raw changes. In Level-1

there may be more than one file since a new one is set up every time a data queue reaches a defined threshold. In this case the job would process several, concatenated input files.

After successful completion of this process the data will be handled by a second batch job. "job 2", which is triggered automatically on completion of the predecessor and converts the data to the corresponding format required by the target system. This job creates a Level-3 repository and triggers a third batch job.

The third batch job handles the transfer to the target system and the update of the target data. For transfers to workstations and other non-MVS systems the UCM-UBS Command Manager is used. For mainframe DB2 a modified load utility is utilized.

Each of these jobs is monitored during its run by the HCP and its history recorded. Abended jobs are flagged and can be restarted individually (from the HCP panels). Besides that, no new Level-3 job will be submitted for the same repository thread when a predecessor has failed.



**UNTERNEHMENSBERATUNG
SOFTWARE SERVICE GMBH**