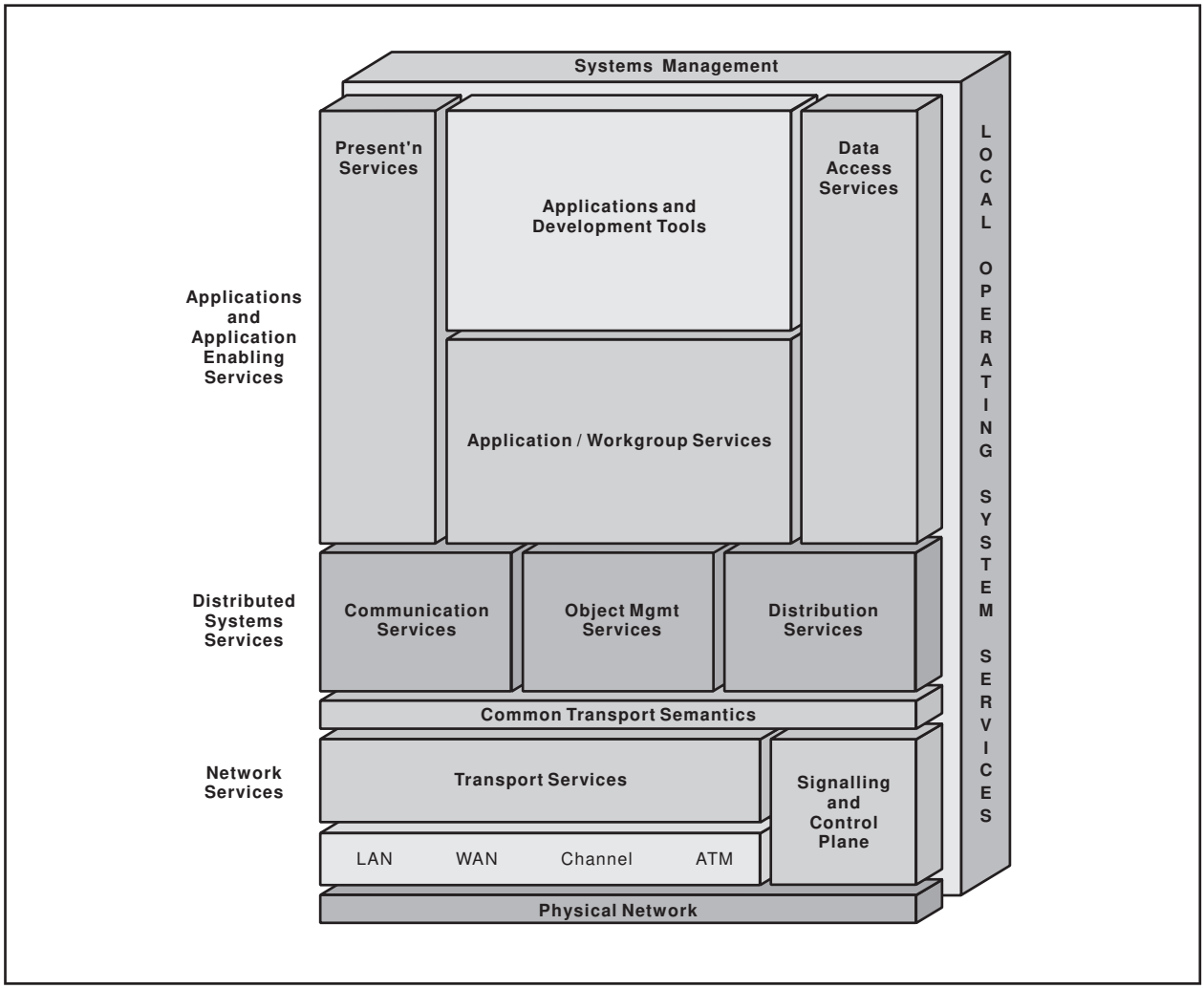
Open Blueprint

Time Resource Manager



GC23-3929-01

Open Blueprint

Time Resource Manager

GC23-3929-01

About This Paper

Open, distributed computing of all forms, including client/server and network computing, is the model that is driving the rapid evolution of information technology today. The Open Blueprint structure is IBM's industry-leading architectural framework for distributed computing in a multivendor, heterogeneous environment. This paper describes the Time resource manager component of the Open Blueprint and its relationships with other Open Blueprint components.

The Open Blueprint structure continues to accommodate advances in technology and incorporate emerging standards and protocols as information technology needs and capabilities evolve. For example, the structure now incorporates digital library, object-oriented and mobile technologies, and support for internet-enabled applications. Thus, this document is a snapshot at a particular point in time. The Open Blueprint structure will continue to evolve as new technologies emerge.

This paper is one in a series of papers available in the *Open Blueprint Technical Reference Library* collection, SBOF-8702 (hardcopy) or SK2T-2478 (CD-ROM). The intent of this technical library is to provide detailed information about each Open Blueprint component. The authors of these papers are the developers and designers directly responsible for the components, so you might observe differences in style, scope, and format between this paper and others.

Readers who are less familiar with a particular component can refer to the referenced materials to gain basic background knowledge not included in the papers. For a general technical overview of the Open Blueprint, see the *Open Blueprint Technical Overview*, GC23-3808.

Who Should Read This Paper

This paper is intended for audiences requiring technical detail about the Time Resource Manager in the Open Blueprint. These include:

- Customers who are planning technology or architecture investments
- Software vendors who are developing products to interoperate with other products that support the Open Blueprint
- Consultants and service providers who offer integration services to customers

© **Copyright International Business Machines Corporation 1995, 1996. All rights reserved.**
Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Introduction 1
Distributed Time Service 1
Functions of DTS 1
DTS Services and Features 2

DTS Structure 3
Time Clerk 3
Time Servers 3
Time Measurement 5
Programming Support 9

Appendix A. Referenced Publications 17

Appendix B. Notices 19
Trademarks 19

Appendix C. Communicating Your Comments to IBM 21

Figures

1. DTS Time Clerks and Servers 3
2. Local, Courier, and Global Time Servers 4
3. Computing the Correct Time 8
4. ISO Format for Time Display 10
5. Absolute Time Structures 10
6. Relative Time Structures 11
7. DTS Server/External Time-Provider Communication 13

© Copyright IBM Corp. 1995, 1996

Introduction

Distributed Time Service

A distributed computing system has many advantages, but also brings with it new challenges. One challenge is keeping the clocks synchronized on different systems. In a single system, one clock provides the time of day to all programs. Computer hardware clocks are not completely accurate, but there is always one consistent notion of what time it is for all processes running on the system.

In a distributed system, however, each system has its own clock. Even if it were possible to set all of the clocks in the distributed system to one consistent time at some point, those clocks would drift apart in time at different rates. As a result, each system has a different idea of what time it is. This is a problem, for example, for distributed programs that have dependencies on the ordering of events. It is difficult to determine whether Event A on System X occurred before Event B on System Y because System X and System Y may have different notions of the current time.

The Distributed Time Service (DTS) solves this problem in two ways:

- Automatically, periodically synchronizes the clocks on each system in a distributed system.
- Keeps the synchronized notion of time close to the correct time. (In DTS, correct time is considered to be Coordinated Universal Time [UTC], an international standard.)

These services together allow cooperating systems to have the same notion of what time it is, and also to have that time be meaningful in the rest of the world.

Time in a distributed system is inherently more complex than time originating from a single source. Because clocks cannot be continuously synchronizing, there is always some discrepancy in their calculations of the current time as they drift between synchronizations. Also, indeterminacy is introduced in the communications necessary for synchronization. Clocks synchronize by sending messages about the time back and forth, but that message passing itself takes a certain (unpredictable) amount of time. So besides being able to express the time of day, a distributed notion of time, must also include an inaccuracy factor (how close the timestamp is to the real time). As a result, keeping time in a distributed environment requires not only new synchronization mechanisms, but also a new form of expression of time — one that includes the inaccuracy of the given time. In DTS, distributed time is therefore expressed as a range, or interval, rather than as a single value.

The Open Blueprint Time Service is based on the X/Open Distributed Computing Environment (DCE) Time Services.

Functions of DTS

DTS is a software-based service that provides precise, fault-tolerant clock synchronization for systems in Local Area Networks (LANs) and Wide Area Networks (WANs). The clock synchronization that is provided by DTS enables distributed computing programs to determine event sequencing, duration, and scheduling.

DTS consists of software components on a group of cooperating systems; it conforms to the client/server model that is used in distributed systems. In the DTS implementation, each server supplies the time to many client systems and programs through intermediaries called clerks. Clerks reside on their client systems. (The term *entity* refers to the server or clerk process when they have the same functions.)

© Copyright IBM Corp. 1995, 1996

Most systems have a DTS clerk that adjusts the clock on its client system; clerks use Open Blueprint Remote Procedure Calls (RPCs) to obtain time values from one or several servers in the network. The systems that do not have DTS clerks have DTS servers. Besides providing time values to clerks, servers also adjust the system clocks on their systems. DTS servers may obtain reference time values from sources of standardized time that are outside of the network. DTS servers communicate with each other to compute the time interval if an external provider is not on the system.

Because no device can measure the exact time at a particular instant, DTS expresses the time as an interval that contains the correct time. In the DTS model, clerks obtain time intervals from several servers, and compute the intersection where the intervals overlap. Clerks then gradually adjust the system clocks of their client systems to the midpoint of the computed intersection. When clerks receive a time interval that does not intersect with the majority, the clerks declare the non-intersecting value to be faulty. Clerks ignore faulty values when computing new times, thereby ensuring that defective server clocks do not affect clients.

DTS also permits time values from outside sources such as the U.S. National Institute for Standards and Technology (NIST). DTS uses the Coordinated Universal Time (UTC) standard that has largely replaced Greenwich Mean Time (GMT) as a reference. Many standards bodies disseminate UTC by radio, telephone, and satellite; commercial devices (time-providers) are available to receive and interpret these signals. DTS offers a Time-Provider Interface (TPI) that describes how a time-provider process can pass UTC time values to a DTS server and propagate the time values in the network. The TPI also permits other distributed time services to interoperate with DTS.

DTS Services and Features

DTS provides many other valuable services for computer networks that run distributed programs. The major features and benefits of DTS are:

- Correctness - DTS maximizes the probability that a client receives the correct time. DTS uses Coordinated Universal Time (UTC) as a base reference and defines any time interval containing UTC as correct.
- Fault Tolerance - DTS reports faulty servers, and does not use their time values during clock synchronizations.
- Management Capability - Control programs control and monitor the DTS software.
- Application Programming Interface (API) - DTS provides an interface that allows programs to obtain, compare, and calculate UTC time values, and to convert between UTC and other commonly-used time formats.
- Local Time Translation - When displaying time values, DTS translates the UTC times that it uses internally into local time values.
- Monotonicity - DTS (with the operating system) usually provides uni-directional gradual clock adjustment. The control program, though, can be used for non-monotonic abrupt clock adjustment.
- Automatic Configuration - DTS entities use RPC profiles (search tables) to obtain the locations of servers in a local area or cell.

DTS Structure

Time Clerk

The Time Clerk is the client side of DTS. It runs on a client machine such as a workstation, and keeps the machine's local time synchronized by asking Time Servers for the correct time and adjusting the local time to it.

The Time Clerk is configured to know the limit of the local system's hardware clock. When enough time has passed that the system's time is above a certain inaccuracy threshold (that is, the clock may have drifted far enough away from the correct time), the Time Clerk issues a synchronization. It queries various Time Servers for their opinion of the correct time of day, calculates the probable correct time and its inaccuracy based on the answers it receives, and updates the local system's time.

The update can be gradual or abrupt. If an abrupt update is made, the clock is modified to reflect the new time. Usually, the clock is updated gradually; in this case, the tick increment is modified until the correct time is reached. In other words, if a clock is normally incremented 10 milliseconds at each clock interrupt, and the clock is behind, then the clock will instead be incremented 11 milliseconds at each clock tick until it catches up. See “Adjusting System Clocks” on page 8 for more information on tick increments.

Figure 1 shows a LAN with two Time Clerks (C) and three Time Servers (S). Each Time Clerk queries three Time Servers when synchronizing; the Time Servers all query each other. (Although the default number of Time Servers that Time Clerks query is three, Figure 1 shows two being queried for simplification of the drawing.)

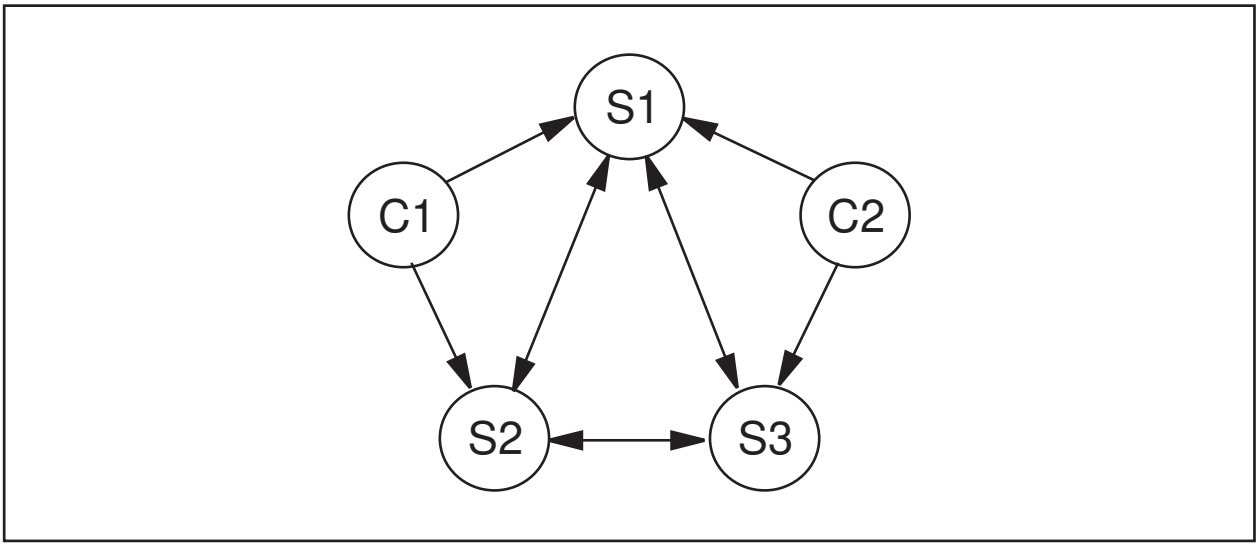


Figure 1. DTS Time Clerks and Servers

Time Servers

A Time Server answers queries about the time. The number of Time Servers in a DCE cell is configurable; the default for a LAN is 3 for servers and 1 for clerks. Time Clerks query these Time Servers for the time, and the Time Servers query each another, computing the new system time and adjusting their own clocks if necessary. One or more Time Servers can be attached to an External Time Provider (described in “The Time-Provider Interface” on page 12).

© Copyright IBM Corp. 1995, 1996

A distinction is made between Local Time Servers (Time Servers on a given LAN) and Global Time Servers (Time Servers outside a given LAN), because they are located differently by their clients. A client may need to contact a Global Time Server, for example, if the client wants to get time from three servers, but only two servers are available on the LAN. Also, a DTS system may be configured to have 2 LAN servers and 1 Global Time Server synchronizing with each other, rather than just having Time Servers within the LAN synchronizing with each other. Couriers are needed in this situation.

A Courier Time Server is a Time Server that synchronizes with a Global Time Server (a Time Server outside the Courier's LAN). It imports an outside time to the LAN by synchronizing with the Global Time Server. Other Time Servers in the LAN can be designated as Backup Courier Time Servers. If the Courier is not available, one of the Backup Couriers serves in its place.

Figure 2 shows two LANs (LAN A and LAN B) and their local Time Servers (S). In each LAN, one of the local Time Servers acts as a Courier Time Server (Co) by querying a Global Time Server (G) for the current time.

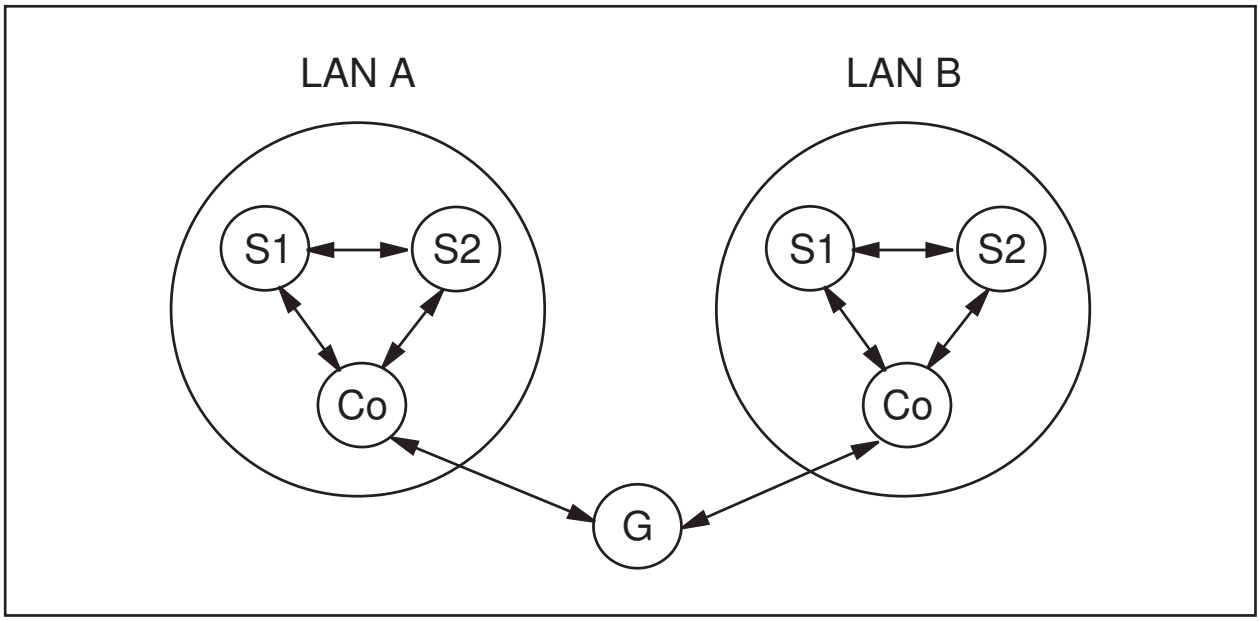


Figure 2. Local, Courier, and Global Time Servers

Local servers are available only to the servers and clerks in a single LAN, but global servers are available throughout a cell. Any server can be configured as either a local or a global server.

The number of global servers is usually small, but global servers have several important functions that enable DTS to synchronize every node in the network. Global servers are necessary in the following situations:

- When a network has multiple LANs, or an extended LAN
- When systems that are not on LANs have access to LANs through point-to-point links
- When clerks or local servers cannot access the required number of local servers

Local servers, called couriers, request time values from one randomly-selected global server at every synchronization.

Couriers maintain lists of global servers whose bindings they import from the cell profile. At every synchronization, couriers use the responses of all local servers and one global server when synchronizing their own clocks. Couriers provide network-wide synchronization by doing the following:

- Requesting time values from at least one global server in a remote area, and request the balance of values from local servers up to the required number.
- Using the global server times and local server times to synchronize the clocks in their respective systems.
- Relaying newly-computed clock times to other servers and clerks on the LAN during future synchronizations.

For a network containing multiple LANs or point-to-point links, one server on each LAN or segment must be configured as a courier. This configuration ensures that various portions of the network remain synchronized, and are not isolated from each other.

Locating Time Servers

To synchronize time, clerks need to be able to locate servers, and servers need to be able to locate servers automatically. Servers are located by using Remote Procedure Call (RPC) profiles.

Each DTS clerk node contains up to three profiles: a system profile, LAN profile, and cell profile. When it attempts to locate servers, a clerk first performs an RPC lookup of the entries in the system profile. The clerk then looks for the LAN profile entry. If the LAN profile entry is not found, the clerk searches for the default profile entry which may contain the LAN profile entry. When the clerk locates the LAN profile, it reads the server entries to build a list of local servers. This process is repeated at set intervals.

If a clerk does not obtain enough server entries (a minimum specified via administration), it attempts to locate additional servers, usually those outside the LAN. To locate these servers, a clerk locates the cell profile, which has a well-known Cell Directory Service (CDS) name. The cell profile contains global server entries; that is, servers that are normally found outside the LAN.

Servers import information about other servers from LAN and cell profiles in the same way.

Time Measurement

The following factors affect time measurement:

Clock error: All system clocks have common properties that contribute to clock error and interfere with the synchronization process. System clock error tends to increase over time; the error rate of change is known as *drift*. If each system clock in a network started at the same time and ran at the same rate, the clocks would remain synchronized. But, because each system clock drifts at a different rate, however, the system clocks throughout a network become unsynchronized.

The difference between any two clock readings is known as the *skew* between the clocks. The clocks that are used in many computer systems have a specified maximum drift of a few seconds per day. If uncorrected for several days, the skew between networked system clocks can inhibit the performance of distributed applications.

The DTS server or clerk on each system tracks the drift of its client's system clock, and periodically synchronizes with other DTS systems to reduce the skew between its client's time value and those of the other DTS systems. The DTS server or clerk adjusts the system clock on its client system as the final step of this repeating synchronization process.

Communications and processing uncertainties: Communications delays also inhibit the synchronization process, especially when two systems communicate over a WAN or low-speed link. DTS can adjust for the known processing delays that are required to send and receive messages between systems. Because of the varying quality of communications links, however, the time required to send, receive, and acknowledge messages varies from one message to the next. These delays cannot be known exactly, because transit over the network and the time required to read an incoming timestamp both vary.

Rather than using estimates of communications and processing delays, DTS records all known error factors that accompany a time measurement sent over the network. This measurement enables DTS to determine the relative quality of a time source regardless of its geographic location or changing conditions on communications links.

Determining Accuracy

To synchronize system clocks to the most accurate settings, DTS needs a way to determine the accuracy of time sources relative to each other and to UTC. This section describes how DTS determines the relative accuracy of any time source available in the network.

DTS uses an *inaccuracy value*, or *tolerance*, to determine the relative precision of time values that it obtains from system clocks and external time-providers. This DTS feature effectively transforms each time value into an interval, or range, rather than a point on a continuum.

Inaccuracy values are determined by the following three factors:

- **Drift**

  When reading a clock, DTS calculates the amount of time that the clock may have drifted since DTS last read the clock. Drift is the largest component of most inaccuracy values.

- **Communications delay**

  The inaccuracy also contains the uncertain portions of the communications delays between systems. Although DTS compensates for processing delays, it cannot predict or directly measure the varying delays that occur on network links. The inaccuracy values that a clerk or server obtains from co-located systems on a LAN tend to be much lower than those obtained from servers outside the LAN.

- **Leap seconds**

  UTC time is measured by atomic clocks, which are extremely stable. The standard, however, keeps time based on the earth's position. Because of the slowing of the earth's rotation, it occasionally becomes necessary to advance UTC time by 1 second. These events are known as leap seconds. Leap seconds may occur in the final second of any month, and usually occur about once every 18 months. At the end of each month, DTS accounts for leap seconds by increasing all inaccuracy measurements by 1 second. DTS later adjusts the clocks to remove the extra second of inaccuracy if an external time-provider determines that a leap second did not actually occur.

Without DTS corrections, a system clock's inaccuracy is always increasing. For example, suppose that a clock starts with a UTC time of 0:00:00.00 (midnight) and zero inaccuracy. Because of drift, when the clock next shows a time of 0:00:00.00, the inaccuracy is 8 seconds. UTC time may be 23:59:52.00 or 0:00:08.00, but it is probably somewhere in between. Therefore, the system time is an interval that contains UTC time and is bound by the inaccuracy.

Synchronizing System Clocks

To maintain uniform system times, DTS servers and clerks periodically synchronize the clocks in all network systems. The DTS entity that is on each system performs these synchronizations by requesting that servers send their combined clock and inaccuracy values (time intervals) to the originating system. The entity then uses these values to compute a new system time.

An *epoch* is an identifier that a time server appends to the time values it sends to other servers. Servers only use time values from other servers with whom they share the same epoch number for synchronization.

DTS servers and clerks have slightly different synchronization procedures. Before attempting to synchronize with other systems, DTS servers always check that an external time-provider is present on the server system. A given server requests times from other servers if no time-provider is available. When no time-provider is available and a server synchronizes with its peer servers, the server uses its own system time as one input value when it computes a new system time.

Clerks, on the other hand, cannot have time-providers, and they do not use the system time of their client systems to compute new times. When a clerk is synchronizing its client system's clock, the clerk uses only the time values that it obtains from servers to compute a new system time. Most network systems run the DTS clerk process.

When a DTS clerk requests time intervals from several servers, it uses them to calculate a new time that is correct (that is, contains UTC), and that minimizes inaccuracy. When the servers respond and the DTS clerk calculates network communications uncertainties and drift for each time value, the clerk has a set of intervals (t1 through t4 in Figure 3 on page 8). Because each interval contains UTC, the intersection is the smallest interval the clerk can choose that also contains UTC. This intersection is the computed time. The DTS entity uses the computed time interval to adjust the clock on the system that receives the server values.

Besides eliminating large inaccuracy values during synchronization, DTS also discards intervals that are received from faulty clocks (t2 in Figure 3 on page 8). DTS detects and rejects clock intervals that do not intersect with the majority of the intervals. When DTS detects a faulty interval, it notifies the system manager by displaying an event message, identifying the server that sent the faulty value.

A server that has a high-drift clock, or that is far away in the network, submits its time to the DTS entity (t1 in Figure 3 on page 8), but the large time interval is ignored because more accurate times are available. Note that, in Figure 3 on page 8, the endpoints of correct time (t1) are farther from the computed time midpoint than those of the interval that is declared faulty (t2).

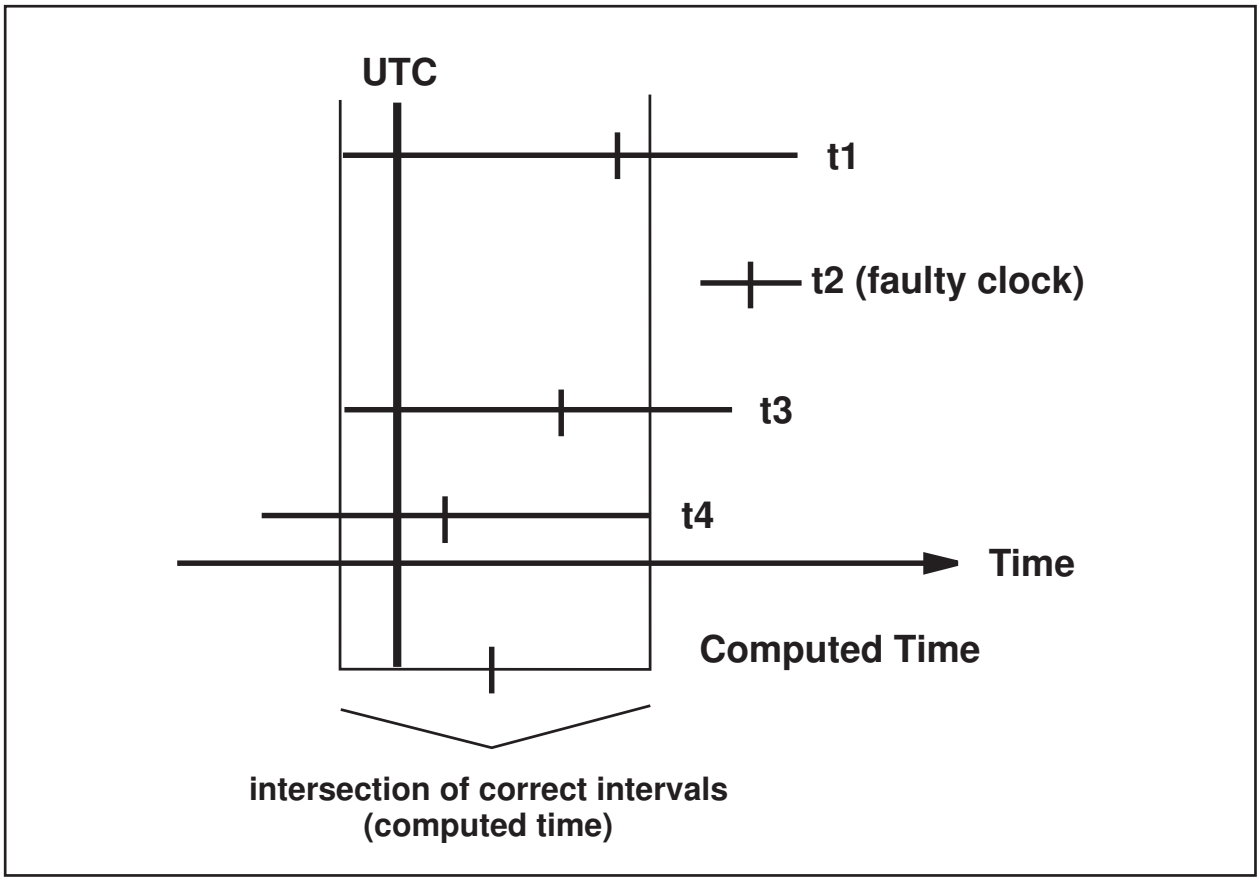


Figure 3. Computing the Correct Time

Computing Time

During the synchronization process, servers with the greatest accuracy have the most influence in determining new system times throughout the network. In Figure 3, the server that submits time value t3 has the smallest correct interval, and is therefore closest to the computed time. Server systems with external time-providers (TPs) are usually the servers with the most accurate times. Beyond TP servers, those servers with the highest quality clocks and best communications links tend to influence the time on other systems to the greatest degree.

The synchronization process also reduces the skew between systems. The computed time interval is often smaller than the interval supplied by any single clock. Note that the computed time in Figure 3 is a smaller interval than any of the source intervals. As the synchronization procedure is constantly repeated on each network system, the skew between systems is reduced, and they are more closely synchronized. However, if a time-provider is absent from the network, the clocks may collectively drift away from UTC.

Adjusting System Clocks

Many system clocks are based on an oscillator, and operate with a combination of hardware and software. The hardware for each clock contains a timer that sends interrupts to the operating system at fixed intervals; each interrupt is a single *tick*. A software variable that contains the current value of the time is incremented by a fixed amount (for example, 10 milliseconds) at each tick. DTS adjusts the rate of the clock by changing only the incremental value that is added to the software variable. It does not directly affect the ticks of the hardware clock.

DTS adjusts system clocks at the rate of 100 to 1; that is, it requires 100 time units to adjust 1 time unit of error. For example, it takes 1 minute and 40 seconds to correct a 1-second error. This rate of adjustment exceeds the normal rate of drift, so that synchronization is carried out without further significant interference from the clock.

In some cases, the system clock should be set immediately rather than gradually. DTS provides this option for the following situations:

- During system start-up to set the initial system time.
- If it has been a long time since the last synchronization, and the skews between system clocks are too large to wait for a gradual adjustment.
- When a network has catastrophic hardware problems that cause a large number of the clocks to become faulty.
- When the time interval for a given clock does not intersect with the intervals of other clocks, and the error exceeds a predetermined tolerance.

Programming Support

Time Representation

The international time standard Coordinated Universal Time (UTC) has largely replaced Greenwich Mean Time (GMT). The standard is administered by the International Time Bureau (BIH) and is widely used. DTS uses opaque binary timestamps that represent UTC. A DTS binary timestamp cannot be read or disassembled; the DTS API allows programs to convert or manipulate these binary timestamps, but they cannot be displayed. DTS also translates the binary timestamps into text strings which can be viewed and printed.

Absolute time is a point on a time scale. For DTS, absolute times reference the UTC time scale; absolute time measurements are derived from system clocks or external time-providers. When DTS reads a system clock time, it records the time in an opaque binary timestamp that also includes the inaccuracy and other information.

DTS uses a standard time format: Coordinated Universal Time (UTC), which notes the time since October 15, 1582, the beginning of the Gregorian calendar. This time is interpreted using the Time Differential Factor (TDF), the time difference in hours and minutes from Greenwich Mean Time. This provides the information needed to display the time in local time (time for a given time zone). For example, the TDF in New York City is -5 hours. The TDF for Greenwich, England is 0. Selecting a time-zone rule (generally at installation) determines the TDF and any seasonal changes to the TDF. All output time then reflects the local time.

DTS displays all times in a format that complies with the International Organization for Standardization (ISO) 8601 (1988) standard. The inaccuracy portion of the time is not defined in the ISO standard. Figure 4 on page 10 explains the ISO format for the time presentation string:

1994-12-22-13:30:25.785-06:00I000.071

In this timestamp example, the year is 1994, the day is December 22, and the time is 13 hours, 30 minutes, and 25.785 milliseconds. A negative time differential factor of 6 hours indicates local time is the U.S. Central Time time zone. The inaccuracy is 71 milliseconds.

In Figure 4 on page 10, the relative time preceded by the + or - indicates the hours and minutes that the calendar date and inaccuracy are offset from UTC. The presence of one of these characters in the string

also indicates that the calendar date and time are the local time of the system, not UTC. The delineator "I" indicates the beginning of the inaccuracy component that is associated with the time.

Relative time is a discrete time interval that is usually added to or subtracted from another time. The TDF that is associated with absolute times is an example of a relative time.

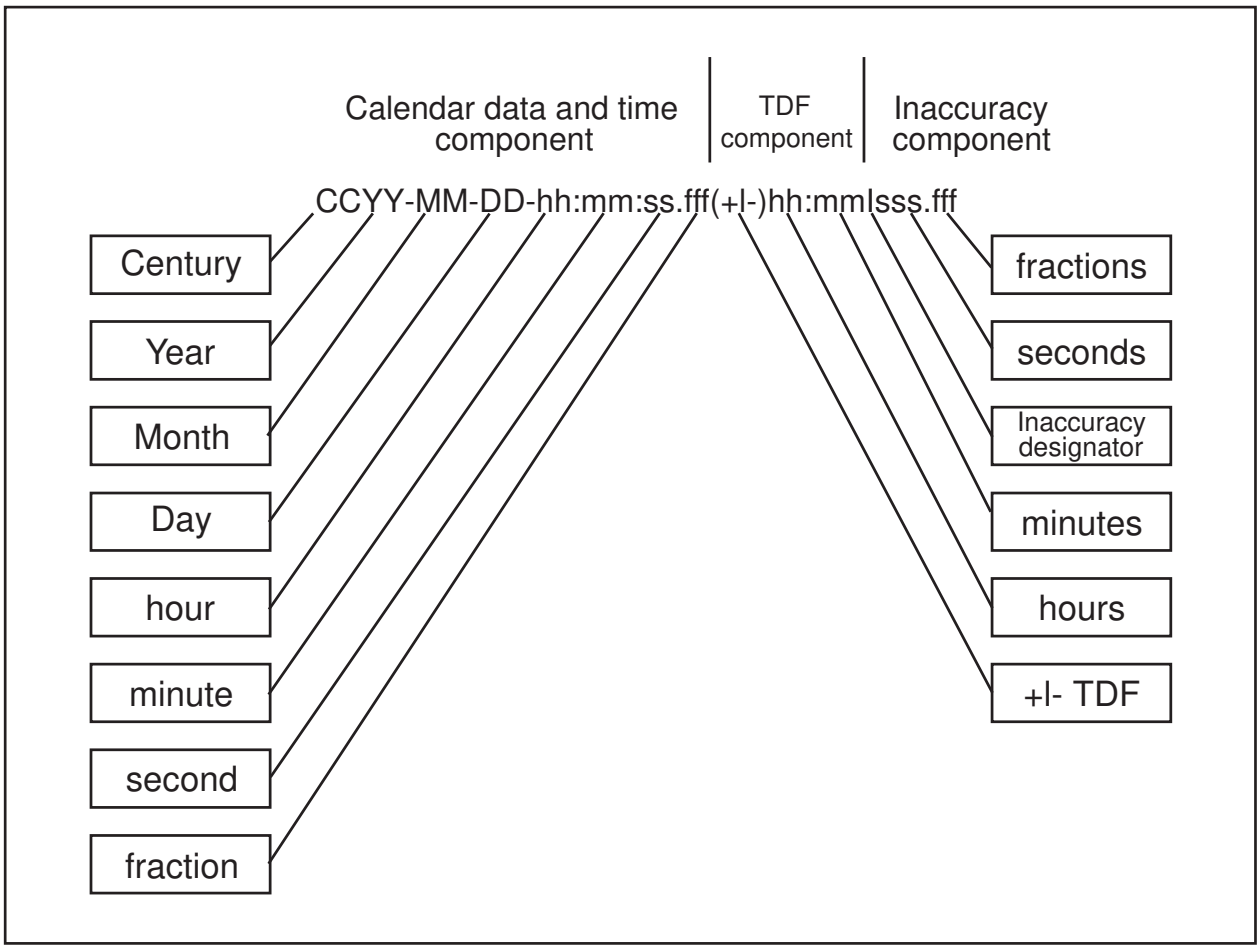


Figure 4. ISO Format for Time Display

Time Structures

DTS can convert among several types of binary time structures that are based on different base dates and time unit measurements. DTS uses UTC-based time structures, and can convert other types of time structures to its own presentation of UTC-based time. The DTS API routines perform these conversions for programs on your system.

Figure 5 lists the absolute time structures that the DTS API uses to modify binary times for programs.

Figure 5. Absolute Time Structures

Structure	Time Units	Base Date	Approximate Range
utc	100-nanosecond	15 October 1582	A.D. 1 to A.D. 30,000
tm	second	1 January 1900	A.D. 1 to A.D. 30,000
timespec	nanosecond	1 January 1970	A.D. 1970 to A.D. 2106

Figure 6 lists the relative time structures that the DTS API uses to modify binary times for programs.

Figure 6. Relative Time Structures

Structure	Time Units	Approximate Range
utc	100-nanosecond	+/- 30,000 years
tm	second	+/- 30,000 years
reltimespec	nanosecond	+/- 68 years

UTC is useful for measuring time across local time zones, and for avoiding seasonal changes (daylight savings time) that can affect the local time. DTS uses 128-bit binary numbers to represent time values internally. Binary timestamps that are based on the utc structure contain:

- Count of 100-nanosecond units since 15 October 1582
- Count of 100-nanosecond units of inaccuracy applied to the preceding item
- TDF expressed as a signed quantity
- DTS version number

The tm structure is based on the time in years, months, hours, minutes, and seconds since 1 January 1900 Greenwich Mean Time. The timespec structure is normally used in combination with, or in place of, the tm structure to provide finer resolution for binary times. The timespec structure specifies the number of seconds (unsigned) and nanoseconds since 1 January 1970 Greenwich Mean Time. The reltimespec structure represents relative time. It is similar to the timespec structure except that the seconds field is signed.

The DTS Application Programming Interface (API)

The Distributed Time Service (DTS) programming routines can be used to obtain timestamps that are based on Coordinated Universal Time (UTC), to translate among different timestamp formats, and to perform calculations on timestamps. Programs can use the timestamps that DTS supplies to determine event sequencing, duration, and scheduling. Programs can call the DTS routines from server or clerk systems.

The Distributed Time Service routines are invoked by C programming language APIs. The DTS API routines offer the following basic functions:

- Retrieving timestamp information
- Converting between binary timestamps that use different time structures
- Converting between binary timestamps and ASCII representations
- Converting between UTC time and local time
- Manipulating binary timestamps
- Comparing 2 binary time values
- Calculating binary time values
- Obtaining time zone information

Distributed resource managers use the DTS APIs and services to generate and manipulate timestamps.

Although DTS provides for the display of timestamps in a culturally-neutral format (ISO format) and uses this format in its administration, this format is not particularly useful for human understanding. Instead, resource managers use the X/Open XPG4 interfaces[10] to display timestamps in a culturally-dependent format. DTS provides APIs that convert the DTS binary timestamp into a format acceptable to the XPG4 API.

Object-oriented interfaces for distributed time services are under specification by the Object Management Group (OMG). IBM intends to support an object-oriented interface for distributed time services when such a standard exists.

The Time-Provider Interface

The distributed system supports the synchronization of its clocks, but there must also be a way to ensure they are synchronized to the correct time. The notion of the correct time must come from an outside source — the External Time Provider. This may be a hardware device, such as one that receives time from radio or telephone sources. This external time is given to a Time Server, which then communicates it to other servers (a Time Server does not notify other Time Servers that the communicated time originated from an External Time Provider). Such an External Time Provider can be very accurate. If no such device is available, the external time source can be the system administrator, who consults a trustworthy time source and enters it into the system. This cannot, of course, be as accurate as an automatic time source, but it may be accurate enough in some cases. DTS supports the ability to interface with an External Time Provider through the Time Provider Interface.

When a time-provider is used with a system running DTS, the external time-provider is implemented as an independent process that communicates with a DTS server process through Remote Procedure Calls (RPCs). The DTS server is the client. The Time-Provider process (TP process) is the server. Both the RPC-client (the DTS server) and the server (the TP process) must be running on the same system.

Figure 7 on page 13 illustrates the RPC calling sequence between DTS and the TP process. Note that darker solid lines represent the path followed by input parameters. The lighter solid lines represent the path followed by output parameters and return values.

The TP process offers two procedures that DTS calls to obtain time values, the ContactProvider and ServerRequestProviderTime. The ContactProvider procedure is called to verify that the TP process is running, to obtain a control message that DTS uses for subsequent communications with the TP process, and for synchronization after it receives the timestamps. The ServerRequestProviderTime procedure is called to obtain the timestamps from the external time-provider. The DTS server synchronizes time using the timestamps returned by the external time-provider (TP process).

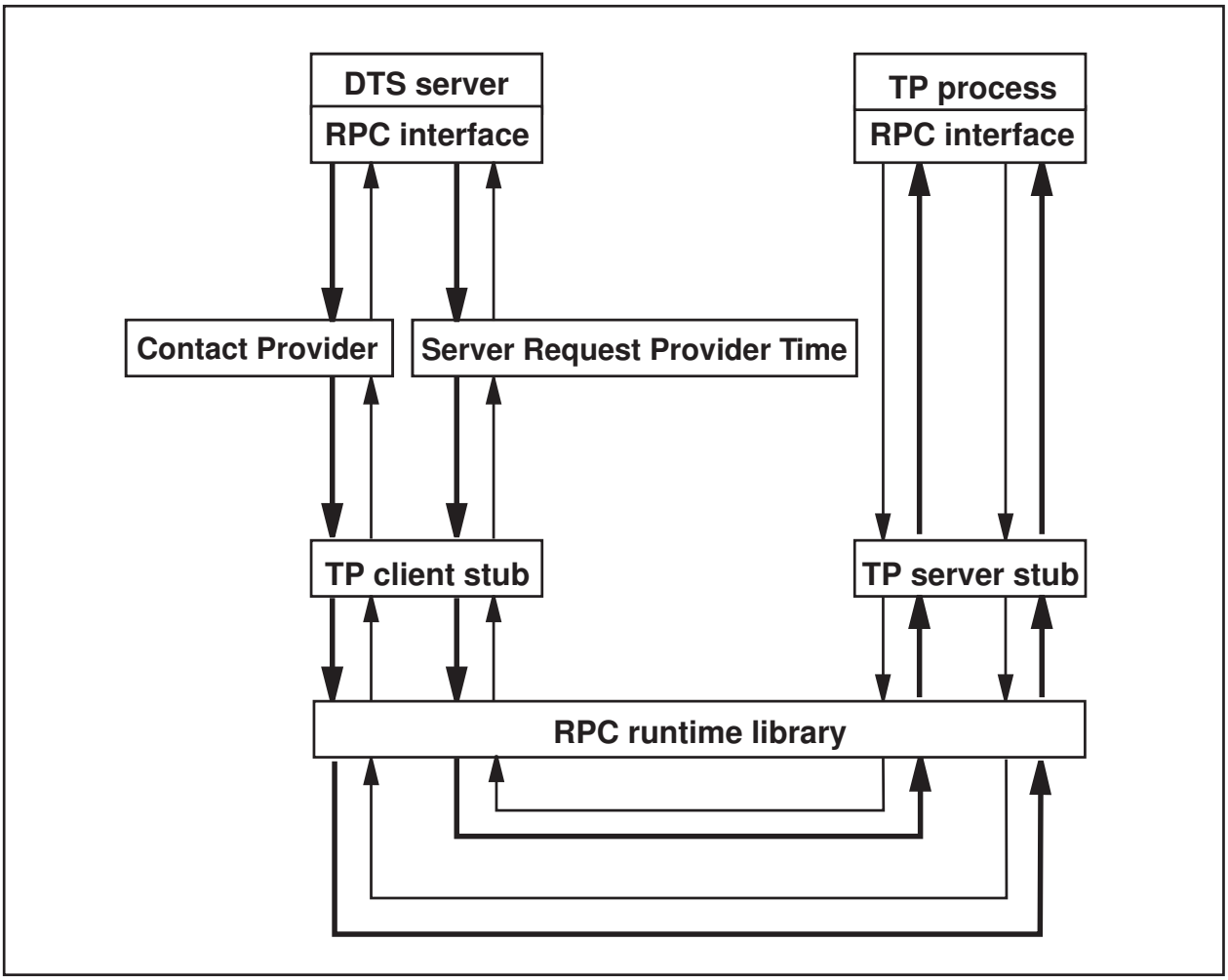


Figure 7. DTS Server/External Time-Provider Communication

Users can write programs that conform to the DTS Server/External Time-Provider Communication model in Figure 7 to introduce external time-providers into their distributed system. An atomic clock or radio clock are examples of external time sources that can be accessed by the distributed time service by using the external time-provider interface and model.

Scalability

The number of servers from whom a clerk requests the time during each synchronization is a configurable parameter. The distributed time service should be configured so that there are enough servers to satisfy the need of every clerk. Servers should also be located close to the clerks to whom they provide the time. This minimizes communication delay which contributes to inaccuracy. Small cells can satisfy these criteria by using a single set of servers serving all clerks. Large cells should be configured with many more servers than are required for any one clerk. In very small cells, a single time server may be enough. All clerks obtain time from that one time server. As time drifts on this single time server, clerk systems will also drift.

Very small systems that don't want the overhead of executing a clerk to locate time servers (even 1) may choose to execute a command to set the local clock by specifying which DTS server from which to obtain time. This sets the local clock abruptly. To keep this very small system time-synchronized requires that this command be executed periodically.

Performance

How often to synchronize a clock within the distributed system is a tradeoff of clock drift versus memory size to run the distributed time service versus impact to network performance. The distributed time service protocols are designed to use the network efficiently. Many parameters associated with each time entity (clerk or server) can reflect or affect operational behavior. For the distributed system's other services, such as security and directory, to continue functioning properly, clocks must be within 5 minutes of each other. This does not imply that the time service must always be run to synchronize clocks. You may synchronize only at system start-up time, or this function can be run on a periodic basis, such as daily. But if accurate time is needed to do event sequencing or scheduling, the distributed time service should be in operation.

Interoperability with Network Time Protocol

Network Time Protocol (NTP) provides the mechanisms to synchronize time and coordinate time distribution in an unmanaged, global-internet environment[5] . NTP uses a returnable-time design, in which a distributed subnet of time servers operating in a self-organizing, hierarchical-master-slave configuration synchronizes local clocks within the subnet and to national time standards using wire or radio. The servers can also redistribute reference time using local routing algorithms and time daemons.

DTS and NTP can be run simultaneously on the same LAN. The following sections describe how to give time to, and get time from, local and remote NTP time sources. DTS provides the following two programs to allow interoperability between DTS and NTP:

- ntp time-provider program which takes time from an NTP server
- null time-provider program which is used on a DTS server whose clock is already synchronized by an external agent such as NTP

To get time from local NTP sources, the DTS server is run on a system that is running an NTP clock driver with a clock and the null time-provider. Other DTS servers take time from this time source.

To get time from remote NTP sources, the DTS server is run with the NTP time-provider on a system with access to an NTP server.

To give time to NTP sources, any DTS server or clerk that runs the nptd server or the xntpd server with the -s option and a special configuration file (ntp.conf) can be configured as an NTP server. In this configuration, NTP never sets the clock. NTP can, however, give the time to other NTP clients.

In your configuration, be careful that loops do not form, such as: `NTP->DTS->NTP`.

Administering System Clocks

The time control program[3] allows you to synchronize, adjust, and maintain the system clocks in a distributed network. The following are the time control program functions:

- Configure the DTS server as a global server
- Modify the epoch and set the local time to a new time
- Establish a DTS entity (a clerk or server)
- Cause DTS to exit on the local system
- Suspend a DTS entity
- Start a DTS entity

- Access help service
- Modify characteristics of a DTS entity
- Show the characteristics of a DTS entity
- Synchronize the system clock with the time obtained from DTS servers in the network
- Remove the global server entry
- Adjust the system clock to a new time (gradually)

Security

The Distributed Time Service has been instrumented with audit events[9] . These auditable events are an interpretation of the Class C2 requirements. Class C2 is the lowest security class that includes any audit requirements. It is widely accepted as the minimum security class for use by commercial and government organizations. The requirements are provided in the Department of Defense Trusted Computer System Evaluation Criteria (commonly known as the Orange Book), the Trusted Network Interpretation (commonly known as the Red Book), and the NCSC audit guideline[11, 12, 13] . The classes of events audited are the time server RPC interfaces for managing the time service and requesting/providing time. These audit events may be displayed through the distributed system audit facility.

Distributed programs can use timestamps to determine event sequencing, duration, and scheduling. An example might be a program that displays log entries, such as audit events or errors, from all systems in the distributed system's administrative domain in a time-sequenced order.

Relationship to Other Open Blueprint Resource Managers

DTS uses RPC resource manager in the communications between DTS clients and DTS servers. It also uses the RPC resource manager in the protocol between a Time Server and a Time Provider.

DTS uses the Directory resource manager (Cell Directory Service [CDS], in particular) to locate time servers. The LAN profile in the CDS contains entries for the DTS local time servers. Global time server entries are kept in the CDS's cell profile. Global directory services (Domain Name System or X.500) are used indirectly if a global time server is registered in a foreign cell.

The DTS principal that represents the server on a given system is the primary access control object for DTS. This principal has controlled access by human users and clerk or server processes. The Access Control List (ACL) for the DTS server can contain any type of ACL entry that is valid for a principal (human or process) or authorization group to which this principal belongs.

Each server executes as the system's principal, which is a member of a unique security group. Entities obtaining time from servers perform authenticated RPCs and verify that the server satisfying their request is an authentic member of the security group.

DTS uses the Audit resource manager to audit its security-critical events.

Relationship to Local System Time Services

Because every system has a clock, the distributed time synchronization routines are layered on top of the local system services, to provide only one path for updating the clock. Local system services also provide APIs for getting and setting time and manipulating and displaying timestamps. Because the systems that comprise the distributed system act as a whole, local time services should not be used to get and manipulate timestamps if the events are to be sequenced with events from other systems within the distributed system.

Appendix A. Referenced Publications

1. *OSF DCE Application Development Reference Version 1.1*
2. *OSF DCE Application Development Guide Version 1.1*
3. *OSF DCE Administration Guide Version 1.1*
4. *Introduction to OSF DCE Version 1.1*
5. *Internet RFC 1305: "Network Time Protocol (v3)"*
6. X/Open, *DCE: Time Services*, Document Number C310, ISBN 1-85912-067-9
7. X/Open, *DCE: Remote Procedure Call*, Document Number C309, ISBN 1-85912-041-5
8. X/Open, *DCE: Directory Services*, Document Number C312, ISBN 1-85912-078-4
9. *OSF DCE RFC 28.1: "DCE Server Auditable-Event Identification and a Proposed Audit Logging API"*
10. X/Open, *Portability Guide/4*, Document Number T905, ISBN 1-85912-053-9
11. *National Security Computer Center, "Department of Defense Trusted Computer System Evaluation Criteria,"* DOD 5200.28-STD, December 1985
12. *National Security Computer Center, "Trusted Network Interpretation of the Trusted Computer System Evaluation Criteria,"* NCSC-TG-005, July 31, 1987
13. *National Computer Security Center, "A Guide to Understanding Audit in Trusted Systems,"* NCSC-001, June 1, 1988

© Copyright IBM Corp. 1995, 1996

Appendix B. Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
USA

Trademarks

The following terms are trademarks of the IBM Corporation in the United States or other countries or both:

IBM
IBMLink
Open Blueprint

The following terms are trademarks of other companies:

DCE	The Open Software Foundation
X/Open	X/Open Company Limited

© Copyright IBM Corp. 1995, 1996

Appendix C. Communicating Your Comments to IBM

If you especially like or dislike anything about this paper, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply. Feel free to comment on specific error or omissions, accuracy, organization, subject matter, or completeness of this paper.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

- If you prefer to send comments by FAX, use this number:

  United States and Canada: 1-800-227-5088.

- If you prefer to send comments electronically, use one of these ID's:
  - Internet: **USIB2HPD@VNET.IBM.COM**
  - IBM Mail Exchange: **USIB2HPD at IBMMAIL**
  - IBMLink: **CIBMORCF at RALVM13**

Make sure to include the following in your note:

- Title of this paper
- Page number or topic to which your comment applies

© Copyright IBM Corp. 1995, 1996

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

GC23-3929-01