The described program analyzes and isolates equipment faults concurrently with regular processing.

If necessary, the program replaces system elements by realigning communication and control paths.

Dependence of the program's replacement decisions upon the recording of extensive error statistics is also discussed.

An application-oriented multiprocessing system

IV The operational error analysis program

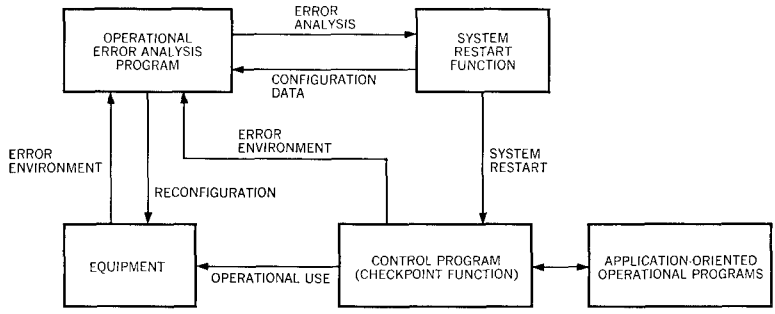
by D. C. Lancto and R. L. Rockefeller

The use of multiprocessing in the Federal Aviation Administration's (FAA's) air traffic control application has necessitated the development of a new type of program—the Operational Error Analysis Program (OEAP). Such a program performs on-line analyses of equipment failure indications concurrently with regular processing, and assesses the source and seriousness of each potential malfunction. If necessary, the program realigns the communication and control paths in the multi-element system to functionally replace failing elements by redundant elements.

Historically, major repair and preventive-maintenance activities have required dedication of the system to these particular jobs. In multiprocessing systems, this is expensive; and in the case of real-time applications, such as air traffic control, it may be prohibited by the application requirements for system availability. The 9020 system (see Part II of this paper) is potentially able to monitor its own errors, isolate failures to an element of the system, and functionally replace failing elements by redundant elements.

In response to application requirements, the equipment design of the 9020 system includes special provisions for configuration control, inter-element error indications, error-checking and logout facilities, and address translation. The system-program design includes the OEAP, system checkpoints in the application programs, and unit and system diagnostic procedures for off-line maintenance. All of these features contribute to the high availability of the 9020 system design. This Part of the paper concentrates on a description of OEAP, mentioning other 9020 control features only as they relate to OEAP.

Figure 1 Equipment and programming relationship



Background

Figure 1 suggests the broad relationship between 9020 elements and the main system programs. The basic provisions for error checking and identification, as well as the prerequisites for program control of system configuration, are designed into the equipment. For the sake of efficiency, the control program was given responsibility for checkpoint and restart functions; hence OEAP is called into use only when an equipment error is reported. OEAP then analyzes the error environment; if the error persists, OEAP isolates the malfunctioning element and removes it from the operational system. After reporting its findings and actions to the control program, OEAP makes any other configuration changes directed by that program. Data that may help to pinpoint the error within the removed element are printed for the maintenance personnel.

The 9020 system consists of seven types of equipment elements: Computing Element, Storage Element, Input/Output Control Element, Tape Control Unit, Tape Drive, Systems Console, and Peripheral Adapter Module. Each of these elements has been designed so that a stored program executed in any Computing Element can control system operation, monitor any error situations, and permit system reconfigurations. There is no built-in master-slave relationship; any relationships between Computing Elements must exist under the surveillance of the particular Computing Element that is running the applicable sections of the control program.

configuration control

The configuration-control feature allows program control over system configurations. Element states, data paths, and control paths are dynamically specified and activated by OEAP. Manual requests for system reconfigurations can be transmitted to the program via a typewriter. Configuration control allows the formation of complete and separate subsystems; these subsystems can be used in program debugging, miscellaneous production work, and scheduled or unscheduled maintenance—as well as in the main application.

error indications

The various system elements contain a variety of checks on data paths, control paths, and environmental conditions (e.g., temperature sensors). The error-handling design philosophy divides

such checks into two categories: (1) those that can be handled with normal I/O techniques by the Computing Element that initiated the operation, and (2) those that must be indicated to the system (in this case, "system" is defined as all Computing Elements that are set up to "listen" to such errors). The first category embraces errors for which the pertinent error environment can be obtained through the normal I/O sense commands. A Peripheral Adapter Module data check, for example, would fall in this category.

On the other hand, a power-failure check from a Peripheral Adapter Module would belong in the second category. System checks, which can occur at any time (as contrasted with data checks, which always occur during operational use of a unit), are transmitted to the diagnose-accessible register in each Computing Element. Because its register is maskable, a Computing Element can selectively accept or ignore indications of system errors.

logouts

A process that preserves element information is said to *log* the information, and the information logged is called a *logout*. The ability of the equipment to log the system environment whenever a malfunction occurs is essential to the OEAP error-analysis function. Each Computing Element has the ability to log many of its own registers, as well as all important registers in Storage Elements, I/O Control Elements, and other Computing Elements; data from Computing Elements and I/O Control Elements go into the preferential-storage area that is controlled by the Computing Element. Whenever an I/O Control Element error condition is detected, the I/O Control Element automatically logs after "permission" is received from a Computing Element. A Computing Element logs automatically when its error logic detects a malfunction, whereas a Storage Element logs under control of the stored program in a Computing Element. For detailed error information from a Peripheral Adapter Module or Tape Control Unit, the Computing Element depends on sense and status data obtained through the normal I/O means. Thus, complete system environment data are available to OEAP for error-analysis purposes.

address translation

In the 9020 system, the address translation feature controls the logical assignment of addresses in each Storage Element. Address translation registers are loaded by the SET ADDRESS TRANSLATOR instruction. Address bands of 32,768 words may be assigned to any Storage Element using the address translation feature. Thus, when the system loses a Storage Element, the reconfiguration process can fill an address gap with no program relocations beyond those needed to correctly load the new Storage Element.

checkpoint

The checkpoint subprogram, part of the operational address-translation program, operates every thirty seconds and records about 56,000 words on magnetic tape. All dynamic tables, as well as the address translation registers, are recorded. Whenever it gains control, the checkpoint subprogram "locks up" each table; no table is recorded until all tables are locked, thus guaranteeing that the latest information is recorded. As each table is recorded (the most-used tables are recorded first), it is unlocked so that operational

processing may resume. The time required to fully complete a checkpoint is about three seconds.

off-line diagnosis

Because configuration control allows the setup of isolated subsystems within the 9020 system, maintenance functions can employ the full complement of unit and system off-line diagnostics prepared for factory and acceptance-test checkouts. The need for separately designed on-line diagnostic programs has been mitigated by this maintenance approach. The multi-element nature of the system, combined with the need for choosing maintenance subsystems from among many possible choices, dictated an approach that places OEAP within the control program framework.

The bulk of the unit diagnostic code was obtained from standard SYSTEM/360 modules. A comprehensive multiprocessing diagnostic control program that operates single or multiple Computing Elements in concurrent fashion was designed and written. A system evaluation program has been provided to check out system paths in the multiprocessor environment. Unit diagnostics for the Peripheral Adapter Module and diagnostics for distinctive 9020 features were generated by the FAA project group.

Program objectives

The main functional objectives of the Operational Error Analysis Program are:

- Error-check analysis and fault isolation
- Maintenance of error statistics
- Error environment reporting
- Reconfiguration

The bulk of OEAP is devoted to analyzing the error-check environment and to isolating, if possible, the malfunctioning element or *interface* (an interface being defined as the equipment between the points at which error checking stops in one element and begins in another element that is communicating with the first element). Malfunctions give rise to abnormal-condition signals of three possible kinds: element checks (ELC's), out-of-tolerance checks (OTC's), and on-battery signals (OBS's). ELC signals can be presented to the diagnose-accessible registers of the Computing Elements in two forms. A *pulsed* ELC is of short duration and is presented once per appearance of a check condition; after issuing a pulsed ELC, the issuing element attempts to continue operation. A *level* ELC, on the other hand, is continuously presented to the diagnose-accessible registers until the check condition is cleared in the issuing element. A level ELC from an element indicates that the element can proceed no further without external help.

Errors are classified as *solid* or *intermittent* by the error-analysis programs. A distinction is possible in most cases because solid error conditions show up as level ELC's in the appropriate error register. Normally, the reading of an error register clears the register, but if the error is solid, the error bit persists in the "on" condition at

successive readings. Error conditions not readily classified as solid are typically classified as intermittent.

The main purpose of the error-analysis function is the identification of malfunctioning elements and interfaces. Although error indications show up classified by type and/or elements involved, error conditions tend to be reported in multiple. For instance, an I/O Control Element that has a problem with a Storage Element may report to the Computing Element that there is an I/O Control Element as well as a Storage Element problem. Finding the failing element or interface then becomes a logical exercise for OEAP. The program systematizes the techniques traditionally used by human beings.

The analysis routines report their findings to the error-control and statistical routine of OEAP. For the benefit of operational and maintenance personnel, OEAP dynamically reports the condition of an element when it reported an error check. A count of the number and frequency of intermittent errors for each element is maintained. Interface errors are recorded when errors occur in both of two elements communicating with each other. These error statistics are helpful in deciding whether one or more elements should be removed from the system. The error count is used by the system operator to decide whether one or both of two interfacing elements must be removed from the system.

Another of OEAP's primary functions is to promptly record error-environment information. When informed that a malfunction has been detected, the control program receives pertinent information about the failure. In case of a solid error, OEAP reports that an element has been deleted; in case of an intermittent error, it must be decided what further action (usually reconfiguration) should be undertaken by OEAP. Within a few seconds of the reported error, relevant information is reported via high-speed printer, typewriter, and magnetic tape.

OEAP has sole responsibility for maintaining the system configuration; except at initial program loading, it alone executes the reconfiguration instruction. Since the configuration control registers are not readily accessible to a program, OEAP simulates their contents in a table that is duplicated in different Storage Elements.

Whenever an error occurs during an I/O operation, the control program attempts to execute the operation again by ordinary retry procedures. OEAP records pertinent retry information, consisting mainly of sense and status data, as an aid to a more efficient maintenance of I/O channels and devices.

OEAP operates in the Supervisor mode and executes most of the privileged instructions. Normally resident in main memory, OEAP is directly called into use by every machine-check interruption. Moreover, when other interruption conditions indicate that an error condition exists, the control program lends control to OEAP. OEAP also controls all of the interruption program status words in the alternate Preferential Storage Area (PSA). (Because the alternate PSA is located precisely 32,768 words in the address range

above the primary PSA, it is located in another Storage Element. The alternate PSA is referred to automatically, via equipment, when the Storage Element containing the primary PSA becomes unavailable.) Some of the editing tasks performed by OEAP are executed in the problem-program mode after system processing is resumed, thus taking less time away from productive processing.

Unless specifically directed to ignore them, OEAP monitors the error conditions reported in non-operational subsystems. The motive here is to ensure that the redundant elements being held in readiness are in first-class operating condition. Whenever a redundant element reports a failure, it is reconfigured into an inactive state by OEAP.

When OEAP gains control to analyze a reported malfunction, other productive processing temporarily halts. Part of the OEAP philosophy is that the Computing Element taking the machine-check interruption will first attempt to execute OEAP. Other Computing Elements in the operational system are directed to begin "time-down" operations of various lengths, the shortest time-down operation being longer than that needed for a normal OEAP recovery. Since the majority of errors occurring in the system are intermittent errors, the Computing Element in which the machine check originated will probably be successful. But if a time-down is completed, the associated Computing Element assumes responsibility for the error analysis and recovery operation.

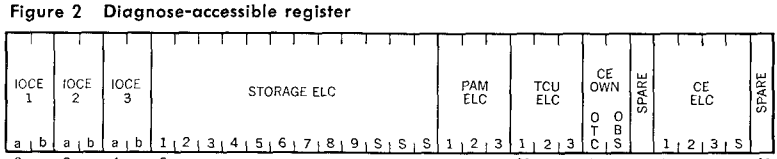
OEAP requires about 52,000 bytes of main storage, as well as a system tape that can be used to restore the OEAP program in case of failure in the Storage Element that contains the OEAP.

Program design

error reporting

The Operational Error Analysis Program is designed to take advantage of the 9020 error-reporting facilities. The diagnose-accessible register illustrates the type of information OEAP has to work with. The format of this register is shown in Figure 2. The diagnose-accessible register provides detailed interruption source information. A bit in the register is unconditionally set on the receipt of an abnormal condition signal. Each bit in the register is individually maskable by a corresponding bit in the select register, except for

Figure 2 Diagnose-accessible register

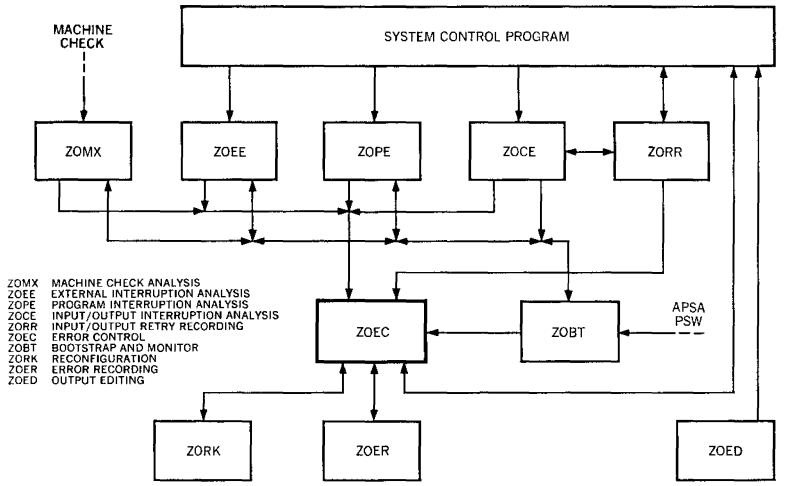


IOCE BITS ARE ENCODED AS FOLLOWS:

a	b	
0	0	NO CHECK SIGNALS
0	1	OBS (PULSE: CCR PARITY; LEVEL: OBS)
1	0	OTC
1	1	ELC

IOCE	I/O CONTROL ELEMENT
PAM	PERIPHERAL ADAPTER MODULE
TCU	TAPE CONTROL UNIT
CE	COMPUTING ELEMENT
ELC	ELEMENT CHECK
OTC	OUT OF TOLERANCE
OBS	ON BATTERY SUPPLY

Figure 3 Relationship of OEAP and system control program



the bits indicating I/O Control Element information. Since such information is encoded, one bit in the select register will mask both I/O Control Element bits. To allow an interruption, a corresponding bit must be "on" in the select register. The control program becomes aware of the fact that a bit in the diagnose-accessible register is set to 1 when both an external interruption occurs and bit 31 in the external-interruption program status word is 1. OEAP is then called to analyze the error environment.

The main error-reporting vehicle is the machine-check interruption, which causes immediate activation of OEAP. Machine checks interrupt a Computing Element when error conditions are detected within this element, in the I/O Control Element controlled by this element, or in the Storage Element being accessed when a failure occurs. In the machine-check case, the error environment consists of logout data and Computing Element check registers.

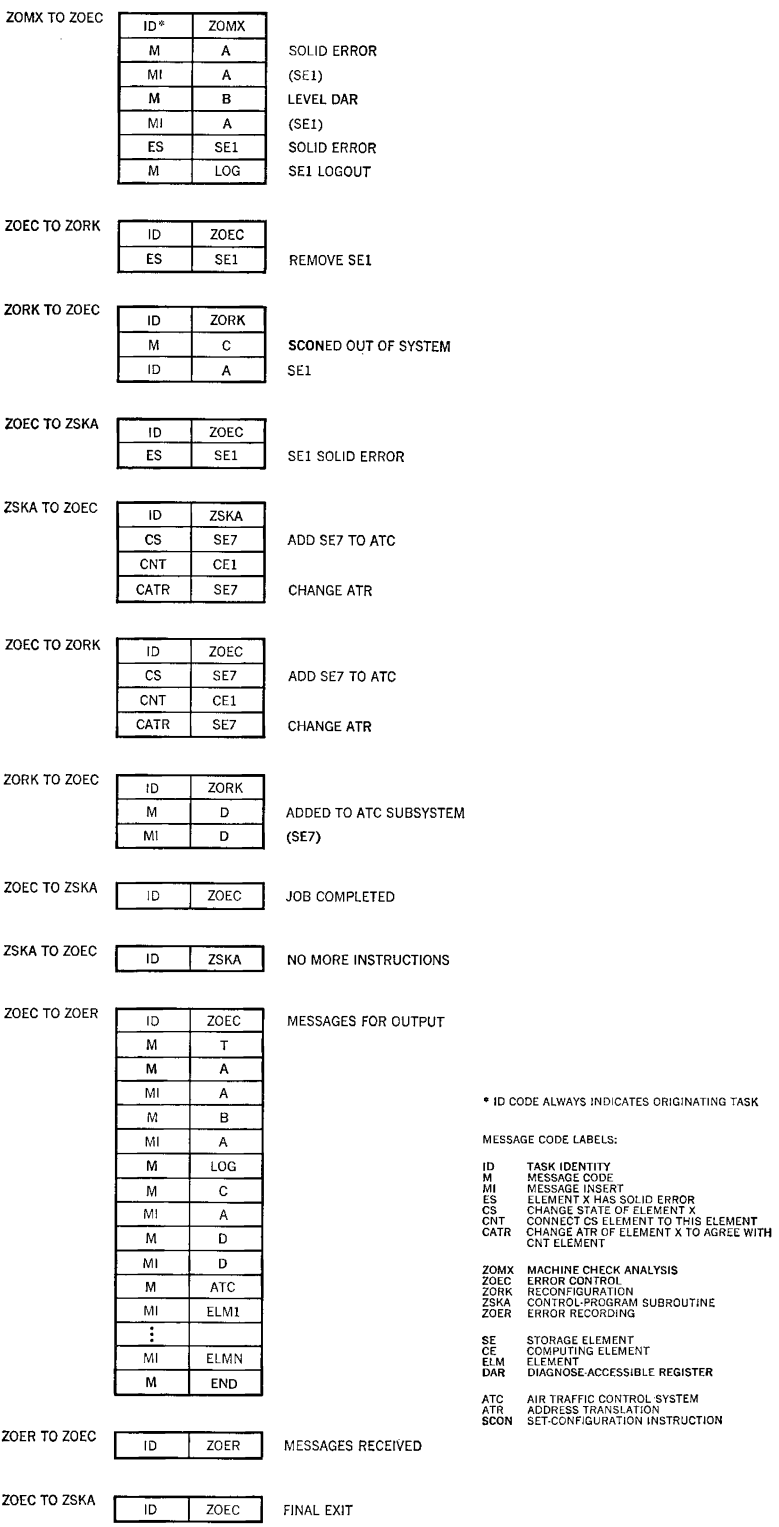
External interruptions are caused by (1) Peripheral Adapter Module and Tape Control Unit problems, (2) a Storage Element which detected a failure while not being accessed by a Computing Element (but being accessed by an I/O Computing Element or not being accessed at all), or (3) one Computing Element informing the other Computing Elements that it has taken a machine check. When an external interruption occurs, the control program activates OEAP.

Certain equipment error conditions manifest themselves in the form of program or I/O interruptions. Such interruptions are interpreted by the control program; if the control program deems that an interruption is associated with an error condition, OEAP is activated.

control program links

The relationship between OEAP and the control program is suggested by Figure 3, which also introduces the OEAP task names and flow concepts. From the diagram, it can be seen that there are six

Figure 4 Typical set of OEAP internal interface codes: SE1 has a solid error and is replaced by SE7



connecting paths between OEAP and the control program, the major path being the one that links the error control task (called ZOEC) with the control program. As is obvious from the diagram, ZOEC is the focal point of OEAP. Once entered, ZOEC controls all subsequent OEAP activity.

The OEAP design includes a set of "interface" codes. Whenever one task transfers control to another, or whenever ZOEC returns control to the control program, a list of the interface codes is passed. These codes can indicate analytical findings and they can specify operations to be performed by the program gaining control. Each interface code occupies two bytes. The design also makes provision for passing data independently of the interface codes. Shown in Figure 4 is a typical sequence of codes that might be involved when the machine-check analysis task (ZOMX) is entered. Note that OEAP knows which task is responsible for each list.

error environment analysis

Four error analysis tasks are shown in Figure 3: machine-check analysis (ZOMX), external-interruption analysis (ZOEE), I/O-interruption analysis (ZOCE), and program interruption analysis (ZOPE). Each of these tasks analyzes the environment surrounding a reported error, but only one operates at a given time. Controls are built into the program to prevent more than one Computing Element from trying to simultaneously execute the error-analysis tasks.

ZOMX is entered when one of the active Computing Elements takes a machine-check interruption (the Computing Element logs its vital registers and control triggers before yielding control). ZOMX immediately locks up OEAP by turning on the OEAP "active" switch and by masking off interruptions that can be masked. The "active" switch is a control that prevents more than one Computing Element from executing the error analysis tasks at the same time.

A Computing Element obtains error environment data from three main sources: a machine check from itself, an I/O Control Element, or a Storage Element. To begin its analysis, the Computing Element looks at the check-register data in its own logout. A Computing Element logs itself automatically when it experiences a machine check. From these data, the element can determine the source or sources of the error condition. There are 24 check-register indicators for various logic checks made on the Computing Element. Another 14 indicators are used to indicate conditions in a Computing Element or in Computing-to-Storage Element interface areas specifically designed for the 9020 system, and four bits serve to indicate which Storage Element is having problems from the Computing Element's standpoint. ZOMX examines each of these indicators and determines the general flow of its subsequent analysis.

If ZOMX determines from the interruption code that an I/O Computing Element malfunction has been reported, the I/O Computing Element's check registers (which have been automatically logged by the element) are examined. The I/O Control Element has 36 internal logic check indicators, of which 12 are included for special 9020 system logic. From either the Computing Element or

I/O Control Element logout, ZOMX may determine that a Storage Element ought to be logged also. ZOMX can request a logout from the appropriate Storage Element. The seven words in a Storage Element logout reflect the status of ten check indicators as well as pertinent registers.

From the data in logouts, ZOMX attempts to trace error indications back to the primary source of trouble. In designing ZOMX and ZOEE, it was anticipated that faults will normally be reported in multiple. For instance, if the Storage Element develops trouble in an I/O Control Element to Storage Element fetch or store, both I/O Control Element and Storage Element report a difficulty. In this case, the Storage Element logout suffices to indicate that the I/O Control Element request reached the Storage Element and that the Storage Element detected its own logic problem. This example is, of course, a relatively simple one for ZOMX to interpret.

After a fault has been isolated to an element, the next step is to determine whether the error is intermittent or solid. ZOMX twice reads the diagnose-accessible register. If the bit that indicates a possible source of error is cleared after the first reading, the error is considered intermittent; if the bit still indicates an error at the second reading, the error is classified as solid. Hence intermittent errors occurring in rapid succession may be considered by OEAP as a solid failure.

Even in the absence of machine-check interruptions, the control program may decide that an error condition has occurred. Because one of the error analysis tasks will be involved in this event, a control switch is used to indicate whether OEAP is already being executed. For most external interruptions, which would require ZOEE, OEAP will already be in use. In that case, the busy Computing Element takes on an OEAP monitoring role, a function to be explained later. ZOEE's function is to analyze the error environment created when a Peripheral Adapter Module or Tape Control Unit reports a problem directly to a Computing Element, when certain error conditions exist for a Storage Element, or when a Computing Element cannot recover from its analysis of a machine check. In each of these cases, ZOEE can look at existing logouts, obtain new data, or merely utilize information from the diagnose-accessible register.

All analysis tasks proceed somewhat similarly once an error has been isolated, i.e., they reveal their identity and generate appropriate interface codes, thus communicating to ZOEC the cause and source of the error. If necessary, the code passed to ZOEC indicates that the source of the error could not be determined.

When ZOCE is invoked because of failure in an I/O operation, the error-environment information consists of a table of thirteen words containing sense and status data, device address, number of retries attempted, etc. When, for instance, a flight-strip printer is addressed as an output device, another useful datum is the character that was being transmitted when the failure occurred plus the three previous characters. These characters are printed to assist the

maintenance man. ZOCE does not attempt any diagnostic I/O operations but relies solely on the information presented to it by the control program.

ZOCE develops and maintains an error history table in which it stores statistical information concerning each device, control unit, and interface. Using this data, ZOCE determines when the control units (Peripheral Adapter Modules and Tape Control Units) and the I/O Control Elements have generated enough intermittent failures to deserve replacement. ZOCE uses parametric algorithms to control these decisions.

ZOPE, the last error-analysis task to be discussed, examines two main error conditions: an I/O Control Element that cannot access its preferential storage, and a Computing Element that has found a Storage Element in a logout-stopped condition. After the control program refers such program interruptions to the OEAP for investigation, ZOPE attempts to restart stopped Storage Elements by logging them. ZOPE reports its actions and conditions to ZOEC.

error statistics

Error statistics are kept by three of the OEAP tasks: ZOCE, ZORR (I/O retry recording), and ZOEC. As mentioned above, ZOCE uses error statistics in determining when a control unit or I/O Control Element should be removed from the system. For example, when one of the adapters (as many as 160 adapters may exist) fails solidly, the Peripheral Adapter Module must still be allowed to remain in the system. ZOCE reports the failure by the proper interface codes, but the Peripheral Adapter Module is charged by ZOEC with an intermittent error. Particular combinations of adapter failures may prompt ZOEC to make an independent recommendation that the Peripheral Adapter Module be removed.

ZORR is a statistical data-gathering task that can be entered by the control program or by ZOCE. Its function is to record sense and status data, either for a particular device address on which an I/O retry is performed or for an I/O operation that was successfully or unsuccessfully retried. ZORR keeps track of the number of retries made in each retry sequence, as well as of the number of times each retry was made with the same response from the device. The information from ZORR's table for a device is reported when a retry succeeds or when the control program abandons its retry attempts.

Most of the error statistics are kept and most of the decisions concerning those statistics are made by ZOEC. In its error table, ZOEC counts the intermittent errors in all active system elements. Whenever an intermittent error is reported, ZOEC not only updates the count for the appropriate element or interface but also compares the new total to a pair of thresholds. The first threshold indicates the point at which the element must be considered as marginally serviceable. At this point, it must be decided whether the element should be removed from the system. If a redundant element can be brought in and a number of other conditions are favorable, the marginal element is replaced. When the second threshold is reached by the error count, ZOEC considers the element to have solidly failed and sets up the appropriate interface codes by which

ZORK (the reconfiguration task) can reconfigure the element from the system. When OEAP has completed its work, error counts and thresholds are printed for review by the operator and the maintenance personnel.

reconfiguration

In the 9020 system, reconfiguration functions are controlled, almost entirely, by OEAP. The main exception occurs at system load time, when an initial program load must be performed before OEAP is resident in main storage. As soon as OEAP is loaded and supplied with configuration information, it reconfigures the system in accordance with its tables. Since the configuration control registers in the 9020 system elements are not readily available to the programmer, OEAP maintains configuration tables (in fact, it stores them in duplicate) to assure that reconfiguration operations are correctly specified and performed.

Reconfiguration operations in OEAP are performed by ZORK, which follows the directions forwarded to it by ZOEC. The commands given to ZORK can require that an element's configuration register and address translation register be changed in various ways. ZORK can delete an element from the system, connect an element to other elements, or change the element's state. ZORK checks whether the configuration register in an element is set by monitoring the response when the element actually sets its configuration control register. If the response is not returned, ZORK tries a second time to set the element's configuration control register. If a second failure is noted, ZORK removes that element from the system and attempts to clear the faulty element's configuration control register into "state zero" for maintenance purposes.

ZORK ensures that a Computing Element does not delete itself or the Storage Element containing OEAP (and thus ZORK itself) from the system.

Whenever a subsystem is to be set up for maintenance, data analysis, or other uses, ZORK sets up the desired configuration at manual request. However, any redundant element needed immediately by the operational system can be recalled by ZORK.

error recording

One of OEAP's basic functions is to document the error environment through its ZOER and ZOED tasks. The normal recording media are magnetic tape, high-speed printer, and 1052 typewriter. Data in raw form are recorded on magnetic tape as backup for the high-speed printer output or for subsequent off-line computer analysis. The high-speed printer is used to present the formatted results of OEAP's analysis and data gathering efforts. Header data is followed by the formatted logout and OEAP's current system configuration data.

Through ZOER and ZOED, OEAP records other information useful to the system operator and maintenance man. This information includes various tables that are normally printed at request, I/O retry data recorded by ZORR, and the results of system reconfigurations made at manual request.

The design goals of OEAP assign an important role to the error analysis monitoring function that is accomplished by ZOBT. It is

interesting to note, though, that a monitoring operation is not essential for successful recovery. Since the majority of errors (perhaps nine out of ten) are expected to be intermittent, the Computing Element taking a machine check should normally make a successful recovery.

error analysis monitoring

When a Computing Element receives a machine check, as explained earlier, the OEAP busy switch is set and other active Computing Elements are forced to take an external interruption. Recognizing that an error condition exists, the control program branches to the ZOEE task of OEAP. Because the OEAP busy switch is set, each Computing Element is forced by the external interruption to execute ZOBT. If the Computing Element actually executing the main portion of OEAP cannot recover, the first Computing Element that starts executing the error monitoring portion of ZOBT takes responsibility for recovery. This decision is based on elapsed time; the monitoring Computing Element allows the prime-recovery Computing Element approximately 350 milliseconds.

ZOBT contains code to overcome the loss of the Storage Element, called the PSA SE, that stores OEAP. If the PSA SE contents must be retrieved from magnetic tape, the monitoring Computing Element allows the prime recovery Computing Element seven seconds for recovery. When time-down is completed, the monitoring Computing Element places the prime recovery Computing Element into the wait state and takes over recovery. If there are other Computing Elements executing ZOBT, each one in turn can become the monitoring Computing Element for the new prime recovery computer.

This explanation of the error analysis monitoring function assumes that only one Computing Element receives a machine check. Actually, whenever any Computing Element begins to execute one of the error analysis tasks because of an error, it sets the OEAP busy switch and directs all other active Computing Elements to execute the error analysis monitoring function.

Summary comment

The Operational Error Analysis Program implements the dynamic on-line error analysis essential to a high-availability multiprocessing system. Although this OEAP discussion emphasizes the programming aspect, the authors realize that adequate error-checking equipment is a prerequisite for advances in programming design.

OEAP's design heavily depends on the convention that the Computing Element receiving a machine-check interruption should attempt recovery. This rule is fine if OEAP is resident in main storage. For applications with severely limited main storage, however, OEAP may have to reside in part on disk or drum; then the design philosophy becomes less desirable because a malfunctioning Computing Element necessitates an I/O operation before the work of analysis can start. Depending on the number of Computing Elements, the nature of the error, and the amount of main storage, trade-offs are obviously involved.