Description

The present invention relates to automatic equipment analysis, and particularly to the service support requirements of computer input/output (I/O) devices that have built-in error detection and error recovery capabilities.

"Predictive Support: Anticipating Computer Hardware Failures", Hewlett-Packard Journal, Vol. 37, No. 11, 1986, pages 30-33 discloses a system which periodically examines the soft error rate of various system components. When these rates approach uptime threatening levels, the Predictive Support system automatically notifies the appropriate person so that corrective action can be taken.

The present invention will be described in connection with a computer disk file subsystem for data storage. Such a storage device may be referred to as a Direct Access Storage (DAS) subsystem. Nevertheless, those skilled in the art will recognize that the invention described may be incorporated into other computer devices, and particularly into various I/O devices having built-in error detection and recovery capabilities.

The occurrence of a recovered I/O device error event may or may not indicate that a service action is required to restore normal operation. If I/O device performance is permanently degraded due to an error event, or if error events recur often enough to noticeably degrade I/O device performance, then a service action should be scheduled. However, a service action is typically not recommended if I/O performance is not noticeably degraded.

The decision whether or not a service action is required is typically made through a manual and empirical process. Detailed error symptom reports are called up and examined to determine some degree of problem severity and permanence effecting machine operation. Because of this complicated process, the decision to call or not call for service action is often based on an inaccurate understanding of the real problem. An incorrect decision can be costly in terms of either lost performance or the replacement of non-defective parts.

The service action decision process to isolate the cause of a possibly very intermittent or usage pattern sensitive fault is also typically conducted through a manual and empirical evaluation process. The preferred prior art fault isolation technique is to use maintenance diagnostic programs to recreate an error symptom. However, this technique is inefficient at recreating intermittent error symptoms. An alternative is to manually analyze error symptom history data to derive a fault syndrome that can be equated to a probable service action. The service action then is to systematically replace suspect parts until error events are no longer reported, as an indication that normal operation has been restored.

An input/output (I/O) device 11 of the prior art is shown with its error response mechanism 13 in FIG. 1. The I/O device receives an I/O operation command to read or write data. If the I/O device successfully completes the operation, it can respond that the operation has been completed without error (response (1)). If, however, an error is detected, the I/O device proceeds with its error symptom analysis and error recovery procedure, and either recovers successfully, or fails to recover. If the operation is successfully recovered, it can transmit a command response that the operation is complete, but reporting a recovered error symptom (response (2)). If the operation is not recovered, the I/O device issues a command response that the operation is incomplete, and includes a damage report and a report of the non-recovered error symptom (response (3)).

When the I/O device reports successful completion of the operation without error, that condition reflects normal machine operation. A report of operation complete, but with a recovered error symptom report indicates that the I/O device operations can continue, but that the error symptom data should be examined to determine if a service action is required. A report that the operation is incomplete, with the damage report and the non-recovered error symptom requires immediate attention to restore normal I/O device operation. A service action may or may not be required as a result of either of the error symptom reports. Any service action requirement must be determined external to the I/O device by manual analysis of error symptom data.

For example, in a DAS subsystem of a computer system, as the user performs write and read operations, transferring data to and from the subsystem, the disk file will periodically, in the manner known, generate a data check message or an equipment check message indicating that some other than routine event has occurred. Data checks are, in many disk files, somewhat expected events, and thus are counted and recorded in ordinary bookkeeping associated with the DAS subsystem, and may not indicate that a service action is required.

Usage data related to the DAS subsystem is also accumulated. Such data might include which I/O devices are being used, seek counts, and data transfer accounts. This is sometimes referred to as usage information.

Conventionally, a report is generated periodically (such as daily or weekly) noting all of the recovered exceptions or other potential error events that occurred since the last reporting took place. A person will analyze this report to determine if the events recorded represent a problem with the I/O device, and whether corrective action is needed, and what that corrective action might be. Nevertheless, under current systems, this analysis must be performed using only the printed reports to note exception events and detect trends in error events.

Accordingly, the present invention provides a method of automatically detecting and analyzing exception events occurring in a computer peripheral subsystem connected to a host computer system, the method being characterised by the steps of: maintaining a Subsystem Environment data base comprising information relating to the current configuration of said computer peripheral subsystem and usage and error information relating to said peripheral subsystem; conditionally invoking a Problem Description procedure when the data in said Subsystem Environment data base reflects an exception event that indicates either a clear or a probable need for service attention to said peripheral subsystem, said Problem Description procedure comprising: developing a current Problem Profile record describing said exception event; creating from stored rules a view of a Problem Profile data base relevant to said exception event; scanning a Problem Profile data base comprising Problem Profile records of previously occurring exception events to relate said current Problem Profile record to a Problem Profile record that already exists in said Problem Profile data base; if said current Problem Profile record matches an existing Problem Profile record in said Problem Profile data base, and said existing Problem Profile record indicates that a service action is pending for the covered problem, updating said existing Problem Profile record; if said current Problem Profile record matches an existing Problem Profile record in said Problem Profile data base, and said existing Problem Profile record indicates that a service action is not pending for the covered problem, updating said existing Problem Profile record and invoking a Problem Evaluation procedure; if said current Problem Profile record does not match an existing Problem Profile record in said Problem Profile data base, adding said current Problem Profile record to said Problem Profile data base and invoking a Problem Evaluation procedure; wherein said Problem Evaluation procedure comprises: examining said current Problem Profile record in view of stored rules relating to acceptable subsystem performance; and if operations of said peripheral subsystem have been disrupted or degraded beyond the limits of acceptable subsystem performance, generating a service alert message, and inserting said service alert message into said Problem Profile record, updating said existing Problem Profile record in said Problem Profile data base, and transmitting said service alert message to said host system.

The present invention also provides a system for automatically detecting and analyzing exception events occurring in a computer peripheral subsystem as defined in Claim 6.

Embodiments of the present invention eliminate the need for an external manual process to determine service action requirements. The external manual process is replaced by a service alert function that is built into the I/O device. The service alert function operates concurrent with other I/O device functions. The internal service alert function of the invention monitors device I/O operations and records exception event records to detect a fault syndrome that requires a service action, determine service action requirements, and initiate a service action via an asynchronous service alert message to the I/O device attachment.

An embodiment of the present invention provides a method of automatically detecting and analyzing exception or error events occurring in a computer peripheral subsystem connected to a host computer system. The method of the invention includes maintaining a Subsystem Environment data base comprising information relating to the current configuration of the computer peripheral subsystem, and usage and error information relating to the peripheral subsystem.

A Problem Description procedure is conditionally invoked when the data in the Subsystem Environment data base reflects an exception event that indicates either a clear or a probable need for service attention to the peripheral subsystem. The Problem Description procedure includes creating from stored rules a view of the Subsystem Environment data base relevant to the exception event, and developing a current Problem Profile record describing the exception event.

A Problem Profile data base comprising Problem Profile records of previously occurring exception events is scanned to relate the current Problem Profile record to a Problem Profile record that already exists in the Problem Profile data base. If the current Problem Profile record matches an existing Problem Profile record in the Problem Profile data base, and the existing Problem Profile record indicates that a service action is pending for the covered problem, the existing Problem Profile record is updated. If the current Problem Profile record matches an existing Problem Profile record in the Problem Profile data base, and the existing Problem Profile record indicates that a service action is not pending for the covered problem, the existing Problem Profile record is updated and a Problem Evaluation procedure is invoked. If the current Problem Profile record does not match an existing Problem Profile record in the Problem Profile data base, the current Problem Profile record is added to the Problem Profile data base.

In the Problem Evaluation procedure, the current Problem Profile record is examined in view of stored rules relating to acceptable subsystem performance. If operations of the peripheral subsystem have been disrupted or degraded beyond the limits of acceptable subsystem performance, a service alert message is generated. The service alert message is inserted into the Problem Profile record in the Problem Profile data base. The existing Problem Profile record in the Problem Profile data base is updated, and the service alert message is transmitted to the host system.

Below a detailed description of a preferred embodiment of the present invention will be given with reference to the drawings, in which

FIG. 1 is a block diagram of an I/O device having an error recovery mechanism, illustrating the types of status responses that can be made to an I/O operation command,

FIG. 2 is a block diagram of an I/O device including the service alert function of the invention,

FIG. 3 is a conceptual diagram of the generation of the Subsystem Environment data base and the application of the Problem Detection Procedure,

FIG. 4 is a block diagram conceptually illustrating the application of the Problem Description Procedure, and

FIG. 5 is a conceptual block diagram conceptually illustrating the application of the Problem Evaluation Procedure.

As illustrated in FIG. 2, a computer peripheral subsystem 21 includes one or more input/output (I/O) devices 23 for receiving operation commands and data from the host computer system. The maximum is typically 64 devices per 7 a subsystem. The subsystem additionally includes an error detection and recovery mechanism 25. As in the prior art, if the operation is successfully completed, a response that the operation was completed without error is generated (response (1)). If, however, an error is detected, the I/O device internal error detection, determination, and recovery mechanism 25 is invoked. The error detection, determination, and recovery mechanism 25 is typically common for all the devices in the subsystem. If the error is successfully recovered, an operation complete message is transmitted, but without recovered error symptom message (response (2)). Thus, to the host computer system, responses (1) and (2) are one message. Any difference is only seen as a difference in command response time. When an operation is incomplete because the error was not successfully recovered, a message indicating incomplete operation, and a damage report are provided by the I/O device (response (3)). In both of the cases in which an error is detected, error symptom data is used only internally by the I/O device service alert function 27. The service alert function conditionally produces an asynchronous service alert message that contains all of the service action requirements necessary to restore normal I/O device operation. A subsystem memory 29 stores data representing the subsystem physical configuration, a subsystem usage log, an exception event log, fault criteria, problem procedures, and a Problem Profile (record) data base. Together, the subsystem physical configuration, the subsystem usage log, and the exception event log form a Subsystem Environment data base.

Service Alert Message Generate Process

The service alert function 27 is a programmed facility that interfaces with I/O devices in the subsystem to receive detailed input data from the subsystem. The Service Alert Message Generate process is carried out by the service alert function 27 and is partitioned into three principal procedures:

1. Problem Detection is a "never ending" subsystem program that manages a Subsystem Environment data base, such that the data base always describes the current configuration, usage, and error environments for the Direct Access Storage (DAS) subsystem. A Problem Description procedure is conditionally invoked whenever the Subsystem Environment indicates either a clear or probable need for subsystem service attention.

2. The Problem Description procedure CREATES a VIEW of the Subsystem Environment data base, such that the view is relevant to the current exception boundary within the subsystem. Information from the exception boundary view is used to develop a Problem Profile that efficiently describes the current problem state for the subsystem. This profile information is either INSERTED into the Problem Profile data base as a "new problem" record, or used to UPDATE another record describing a previous detection of the same problem condition. A Problem Evaluation procedure is invoked whenever the problem state is determined to be "new" or "recurrent". Problem Evaluation is not invoked if the detected problem has already been reported to a host system and service action is pending.

3. The Problem Evaluation procedure first determines Fault Isolation, Fault Severity, and Service Requirements for a Problem Profile record. This information is JOINED with the Problem Profile record in the data base. IF Problem Evaluation concludes that system operations have been disrupted OR degraded beyond the limits of acceptable subsystem performance THEN service alert message Sense Data is generated, JOINED with the Problem Profile record and also reported to a host system so as to initiate a service action.

Problem Detection Procedure

The Problem Detection Procedure monitors DAS subsystem activity with respect to changes in subsystem configuration, such as unit install/remove, data format mode change, Engineering Change install, and resource

fence/spare due fault condition; also, seek/read/write usage of subsystem control paths and devices; and subsystem exception activity caused by error conditions detected and handled during the processing of system tasks. The monitored data is retained in a Subsystem Environment data base that always describes the current subsystem configuration, usage and error status for the DAS subsystem.

The Problem Detection procedure is more explicitly described by the pseudo code sequence included in Appendix 1. The Problem Detection mechanism is illustrated schematically in FIG. 3.

During normal (error free) operation, the Problem Detection stage does "housekeeping" so as to maintain the normal (model) description of the subsystem environment.

Current usage information is received from the disk storage subsystem as an Input Log Record. The current time is applied to the Input Log Record, to place a time mark on the data. Also combined with the Input Log Record is selected historical information from the Subsystem Environment data base. The combined information is read into the current event and trend portion 43 of the Subsystem Environment data base 29(a)(b)(c).

Thus, stored in the subsystem usage log or data base (which is a portion of the Subsystem Environment data base), is information relating to which devices are used, which devices are used more heavily than others, which are used consistently at a certain level, which have intermittent periods of high use, and other information relating to the distribution and amount of usage of the devices in the system. Such information is important to assist in the determination of the nature of failures or exceptions that may be reported by the subsystem. A portion of the data base stores current use rate patterns.

The current time and current use report information (Input Log Record) is read into the current event and trend portion 43 of the subsystem usage data base memory 29(a)(b)(c). The current use rate pattern also becomes a part of the subsystem usage or history data base or log. A relational view 47 is applied to the Subsystem Environment data base to provide a relational configuration and usage history. Thus, the new usage data is always being examined for trends or patterns in view of the subsystem physical configuration and previous usage history. This creates a continuously updated record of subsystem usage and trends and patterns in subsystem usage.

The relational view 47 is developed according to rules 45 stored in a rules memory, which is a portion of the problem procedures 29(e). The Input Log Record is applied to the rules to generate the relational view instruction used to create the relational view 47 of the history data in the Subsystem Environment data base in view of the current use. Rule selection is determined by usage type (such as control unit, path, device data, or device access) and physical address boundary. The rules stored in the rules memory will be determined by expert input as to the way in which the history of usage in the particular device being analyzed should be examined. Thus, this information is entirely a function of the device being examined, and will differ for different devices and subsystems. The rules might specify that when a certain type of usage occurs, a specific portion of the usage data base might be called out as possibly or likely related to that usage.

An Input Log Record may also be supplied by the DAS subsystem when the subsystem physical configuration is changed. Such an Input Log Record is used to update the Subsystem Environment data base to reflect the current subsystem configuration.

When an "unexpected" exception event is input to the data base, a decision is made to either invoke or not invoke the Problem Description procedure.

FIG. 3 also presents a conceptual diagram of storing and analyzing data relating to exception event reports, or error reports, that may be received from various parts of the storage subsystem, such as the device, the channel, or the path from the channel to the device. A journal or data base 29(c) of exceptions and trends is developed and stored in the I/O device error event log memory. The log includes a subpart 43 containing the current exception event and trend information.

When a new exception event report is received (current exception event or current fault) as an Input Log Record, it is time stamped and recorded in the current exception event portion of the exceptions log or data base in the Subsystem Environment data base. The current exception event report is also applied to the set of rules 45 that have been previously stored in the device memory as part of the problem procedures 29(e). Those rules create an instruction representing relational view 47 to be applied to the Subsystem Environment data base in view of the Input Log Record containing the most recently received exception event report. When the Subsystem Environment data base is read through this relational view, history information is provided relative to the specific area of the machine that is of concern, as determined by the current exception event. This allows the determination to be made of how the new exception or failure or error relates to the past history of the subsystem and to the subsystem physical configuration. This is important in viewing and determining patterns and trends in the error reports. Applying this relational view to the Subsystem Environment data base, data related to the current event can be found and applied to the current exception or error event.

The rules 45 to create the relational view 47 of the Subsystem Environment data base are preferably accessed by the current exception event report in a backward chain manner so that the rules are addressed se-

quentially by the error event report. Operating in this manner, as is known, if the first rule is met and its condition satisfied, it will become the relational view instruction, while if it is not, the next rule is examined, and so forth. The order of the rules for examination, and the rules themselves are again a function of the device in which the service alert function is contained, and are generated with expert knowledge from persons intimately familiar with the error and exception characteristics of the particular I/O device.

The current error event, and current trend information, are applied to a set of rules for analysis alert 29(d), which rules contain fault criteria. The output of the rules for analysis alert controls a gate 49 for passing the current exception event and trend information to the next stage for problem description.

The decision to invoke the Problem Description procedure is clear if Exception Type indicates a "hard" fail. System operation is disrupted and service action is very likely to be necessary. More probable though are "soft" fail Exception Types that do not cause significant disruption, but can occur due to a repairable fault condition.

For "soft" fail exceptions, the decision to invoke Problem Description is made when the data base INSERT and UPDATE process detects a significantly recurring sequence of "soft" fail events that are related by symptom AND source. The objective is to not react to random non-recurring exception events, most probably caused by the external subsystem environment.

The fault criteria 29(d) may be set up as a backward chained set of rules for looking at the current event and trend. The rules should provide that when a "hard" failure is encountered (corrective action is needed to restore proper subsystem operation) or when a "soft, pattern threshold exceeded" exception condition is detected an Analysis Alert is generated and the gate 49 passes the current event and trend information to the Problem Description procedure. Expert knowledge of the subsystem being analyzed is required to generate a set of rules containing appropriate fault criteria. Generally, the chain will be ordered by exception priority so that the most severe fault symptom recognized fires first.

The rules for analysis alert should be established so that an analysis alert message is generated to pass the current exception event and trend information on to invoke the Problem Description Procedure more frequently than when there is truly a problem, so that the information can be further analyzed when there is any doubt.

Problem Description Procedure

Illustrated conceptually in FIG. 4 is the mechanism to implement the Problem Description Procedure. The Problem Description Procedure is described in the pseudo code listing of Appendix 2.

The current error event record, when passed to the Problem Description Procedure by the Invoke Problem Description Procedure signal, is applied to a set of rules 51 that are part of the problem procedures 29(e) and create a relational view 53 of the Problem Profile data base 29(f) of exception records of subsystem faults for which analysis is in progress for which service is pending or completed. This relational view insures that only the portion of the data base that relates to the current exception event under consideration is used when applying the selection rules to the current exception event. Again, this is to reduce the computational effort required in the selection. As before, the rules 51 to create the view 53 are generated with expert knowledge as to the factors relating to the particular subsystem and devices in which the service alert function is contained regarding what types of error events might be related or connected.

The selection processor 61 compares the current exception record to the relational view of the Problem Profile data base 29(f) to select one of:

a) new fault;
b) repeat fault;
c) repeat call;
d) known fault (analysis in progress); or
e) known fault (service action pending).

If the current exception record represents a new fault, a status indicator or control signal is placed on the line A. The Build/Update Problem Profile procedure is then invoked, and the signal on the line A becomes a command to analyze for a new fault.

If a link to an existing Problem Profile record is found for which analysis is in progress (the fault has not been reported to the host system yet), a control signal or status indicator is placed on the line B. The Build/Update Problem Profile procedure is then invoked, and the signal on the line B becomes a command to analyze for a known fault.

If the comparison determines that the current exception record represents a repeat fault or a repeat call, the selection rules cause a control signal to be placed on the output line A. A repeat fault is a fault for which a service alert message was sent to the host system, but a service action has not been initiated. A repeat call is a fault for which a service action has been completed, indicating that the service action was not successful.

Although a repeat fault or a repeat call means record of the fault exists in the Problem Profile data base, that Problem Profile data is old history, which should not be used currently by the Problem Evaluation procedure. Therefore, the Build/Update Problem Profile Procedure builds a new Problem Profile record as though the current exception were a new fault. Further analysis is done by invoking the Problem Evaluation Procedure.

If the comparison finds a link between the current exception event record and a Problem Profile record for which a service action is pending, the recurrent problem counter in the Problem Profile record is incremented and the Problem Description Procedure is ended. In this event, the Problem Evaluation Procedure is not invoked.

Build/Update Problem Profile Procedure

The Build/Update Problem Profile Procedure is invoked as a subroutine of the Problem Description Procedure. The procedure is used to build a Problem Profile record for the Problem Profile data base when no Problem Profile record is found in the data base to match the current exception record, and to update the Problem Profile record when a Problem Profile record is found in the data base to match the current exception record.

The Problem Profile records in the Problem Profile data base 29(f) preferably have a common record format. Different bytes in the record indicate if an analysis is in progress for the fault represented by that Problem Profile record, or if a service action is pending, or if a service alert message has been transmitted to the host system, or if a service action has been completed, or other information relevant to the analysis and treatment of subsystem faults.

The Build Problem Profile Record Procedure is illustrated as element 71 (FIG. 4), and is described more fully in the pseudo code listing of Appendix 3. This procedure builds or updates a Problem Profile record in the Problem Profile data base 29(f).

The current exception or Problem Profile record is applied to the Build Problem Profile Record Procedure. If a known fault for which analysis is in progress is involved, the updated Problem Profile record from the Problem Profile data base (not including the current exception record) is applied to the Build Problem Profile Record Procedure so that it can be updated with the current Problem Profile (exception) record. The control signal for a known fault (analysis in progress) causes a set of rules 65 to be invoked to create a view 66 of the Subsystem Environment data base 29(a)(b)(c) relevant to the current exception. The viewed data is provided to the Build/Update Problem Profile Procedure 71, along with the current record and the existing Problem Profile record from the Problem Profile data base 29(f). After the Problem Profile record has been updated, the process then returns to the point in the Problem Description Procedure at which the Problem Evaluation Procedure (Profile Record, Recurrent Problem) is invoked.

If the current exception record is for a new fault, a Problem Profile record is built with the Build Problem Profile Record Procedure 71. The control signal for a new fault causes a set of rules 63 to be invoked to create a view 66 of the Subsystem Environment data base 29(a)(b)(c) relevant to the current exception. The viewed data is provided to the Build/Update Problem Profile Procedure 71, along with the current record. For a new fault, no existing Problem Profile record exists in the Problem Profile data base 29(f). After the Problem Profile record is built, the process returns to the Problem Description Procedure at the point at which the Problem Evaluation Procedure (Profile Record, New Problem) is invoked.

If the current exception record is for a repeat fault, the Problem Profile record is created as for a new fault, after which the process returns to the Problem Description Procedure at the point at which the Problem Evaluation Procedure (Profile Record, Repeat Alert) is invoked.

If the current exception record is for a repeat call, a Problem Profile record is also built as for a new fault, after which the process returns to the point in the Problem Description Procedure at which the Problem Evaluation Procedure (Profile Record, Repeat Call) is invoked.

Problem Evaluation Procedure

The Problem Evaluation Procedure is illustrated schematically in FIG. 5, and is described in the pseudo code listing of Appendix 4.

The Problem Evaluation Procedure is invoked when the Profile Record input parameter contains new problem information to be evaluated. The evaluation task is to determine appropriate service action requirements for the problem condition and determine if a service action should be initiated for problem resolution.

The problem evaluation process is RULE driven, such that information within the Problem Profile Record determines both RULE selection and RULE products. Information produced by RULE execution is JOINED with the Profile Record to become part of the Problem Profile data base. The result is a data base record that

describes both the problem and the results of problem analysis.

If the evaluation process determines that service action is not (yet) required, the updated Problem Profile Record is returned to the Problem Profile data base. A subsequent invocation of the service analysis message process may update and re-evaluate the problem again. Otherwise the record will eventually be purged from the data base.

A set of rules 73 is installed to analyze the faults. The rules 73 are applied to the Problem Profile record (updated) received from the Build/Update Problem Profile 71 (see FIG. 4). If the current fault is a known fault for which analysis is in progress, a service alert message is contained in the Problem Profile record. That service alert message is placed in the message data space 83, from which it is read to the analysis rules 73. The control commands on the lines (A) or (B) instruct the rules 73 whether to analyze for a new fault (including repeat faults and repeat calls), or analyze for a known fault for which analysis is in progress. For a new fault, the rules will generate a service alert message to read into the message data space 83. For a known fault, the rules will modify the existing message read from the data space 83 and read the new message back in the space.

The rules 73 isolate the fault, determine the fault severity, and determine a recommended service action. Output from the rules is a report and update data base control information, which are applied to the service alert and data base update control 81.

The service alert and data base update control 81 receives the report from the rules to analyze faults 73, and the service alert message. The output from the fault analysis rules 73 determines whether the service alert and data base update control 81 forwards service alert message report to the host, or whether it is simply returned to the Problem Profile data base 29(f) to be a part of the observation for future faults. If the report is to be sent to the host, then the Problem Profile data base is also augmented with that report.

If the Problem Evaluation process concludes that a service action is required, a procedure is called to generate service analysis message Sense Data for an appropriate service analysis text message. Service analysis message Sense Data is JOINED with the Problem Profile Record and also presented to a host system.

Appendix 1

```
PROCEDURE Problem Detection .(Input Log Record)

(* The Service Alert Message Generate process requires the
subsystem to periodically report path and device usage count
values, and to report changes in subsystem configuration and
exception events as they occur.  Information is reported in
the form of a log record, where any log record consists of
the components:
     Log Type, Log Source, Log Data and Time Stamp *)

INSERT  (Input Log Record)
   INTO Subsystem Environment.Next Record
   VALUES (Log Type, Log Source, Log Data, Time Stamp)

Current Record := Next Record
Next Record    := Next Record +1

CASE Current Record.Log Type OF
   Device Usage Report
      UPDATE Log Source (Usage Statistics)
        (* Use Current Record and Log Source.Previous Record
           to update current and term usage rate values for:
           path access, motion seek, data write and data
           read. *)

   Changed Subsystem Configuration
      UPDATE Log Source (Vital Configuration Data)
        (* Use the Current Record to determine how the
           configuration has changed; verify change validity;
           and update appropriate configuration data records.
           *)

   Exception Event (type = 'soft')
       SELECT  Time Stamp, Exception Symptom, Symptom
               Domain, Fault Pattern
        FROM   Subsystem Environment
        WHERE  The Exception Data Relation RULE is
               determined from:
               Current Record (Log Data.Symptom * Log
               Source.Boundary)

        UPDATE Current Record (Fault Pattern)
          (* The current record is updated to include a
             fault pattern description derived from the
             data base relational view.
             A fault pattern is characterized by:
             Symptom Profile, Fault Incidents, Fault
             Period. *)

      IF Fault Pattern > Symptom (Threshold Criteria)
```

```
            THEN GOTO Problem Description (App. 2)

      Exception Event (type = 'hard')
         GOTO Problem Description (App. 2)
      END (of case)
END. (Procedure Problem Detection)
```

Appendix 2

```
PROCEDURE Problem Description (Current Record)

(*  This procedure attempts to link the Current Record to a
    Profile Record already generated for a previous
    occurrence of the same problem.  If the link is
    successful, and depending on Profile Record.Status, the
    profiles are updated to reflect new information from
    Subsystem Environment.  Otherwise, a new Profile Record
    is generated. *)

SELECT Profile Record (*.ID, *.Status)
  FROM Problem Profile
  WHERE The Profile Record Relation RULE is determined from:
        Current Record (Log Data.Symptom * Log
        Source.Boundary)

CASE Select Results OF
   Empty Set (* No history link found *)
      Build Profile (Current Record) (App.3)
         (* Procedure creates a Current Record relational
            view of Subsystem Environment.  Patterns for a
            new Profile Record are constructed from
            information within the view. *)

      GOTO Problem Evaluation (Profile Record, New Problem)

   Link Found (*.Status = Not Reported To System) (analysis
   in progress)
      Update Profile (Current Record, Profile Record) (App.3)
         (* Procedure creates a Current Record relational
            view of new (since *.ID) Subsystem Environment
            data base entries.  Profile Record patterns are
            updated from information within the view. *)

      GOTO Problem Evaluation (Profile Record, Recurrent
           Problem)

   Link Found (*.Status = Service Action Pending)
      Profile Record.Recurrent Problem Count := *.* +1
         (* Maintain count of repeated exception events. *)

   Link Found (*.Status = No Service Response) (Repeat Fault)
      Update Profile (Current Record, Profile Record) (App.3)
         (* Update service alert message and report again to
         system. *)

      GOTO Problem Evaluation (Profile Record, Repeat Alert)

   Link Found (*.Status = Service Action Complete) (Repeat
   Call)
      Build Profile (Current Record) (App.3)
         (* Service action not successful.  Build new service
```

```
                alert message for Support Trained CE service
                action. *)

            GOTO Problem Evaluation (Profile Record, Repeat Call)
        END
    END. (Procedure Problem Description)
```

Appendix 3

```
PROCEDURE Build Profile (Current Record)

INITIALIZE Profile Record (Time Stamp = Null)
(*  Prepare to build new Profile Record. *)

PROCEDURE Update Profile (Current Record, Profile Record)

(*  This procedure builds or updates a Profile Record that
    represents a Current Record relational view of Subsystem
    Environment. *)

SELECT  SCU Config, String Config, Device Config
  FROM  Subsystem Environment (Time Stamp > Profile
        Record.Time Stamp)
  WHERE The Configuration Data Relation RULE is determined
        from: Current Record (Log Data.Symptom * Log
        Source.Boundary)

     PROFILE (* Subsystem channels, SP controllers, string
        path controllers, spindles, devices and model/feature
        characteristics installed and operational (not
        fenced) within the "relevant" subsystem boundary. *)

SELECT  SP Use, String Use, Path Use, Spindle Use, Device Use
  FROM  Subsystem Environment (Time Stamp > Profile
        Record.Time Stamp)
  WHERE The Usage Statistics Relation RULE is determined
        from: Current Record (Log Data.Symptom * Log
        Source.Boundary)

     PROFILE (* Usage patterns for subsystem components that
                are installed and operational within the
                "relevant" subsystem boundary. *)

SELECT  ERP Action, ERP Resource, ERP Result
  FROM  Subsystem Environment (Time Stamp > Profile
        Record.Time Stamp)
  WHERE The ERP Data Relation RULE is determined from:
        Current Record (Log Data.Symptom * Log
        Source.Boundary)

     PROFILE (* Results of exception recovery actions invoked
        for "relevant" exceptions, and using components
        within the "relevant" subsystem boundary. *)

SELECT  Command Exception, Exception Symptom, Sub-Symptom
        Parameters, Symptom Domain, Power Status
  FROM  Subsystem Environment (Time Stamp > Profile
        Record.Time Stamp)
  WHERE The Exception Data Relation RULE is determined from:
        Current Record (Log Data.Symptom * Log
```

```
        Source.Boundary)

      PROFILE (* Syndrome patterns detected for "relevant"
               exceptions, and using components within the
               "relevant" subsystem boundary.  Profile
               patterns provide cross-point-isolation to
               probable failure mode(s) and probable failure
               boundary(s). *)
Profile Record.Time Stamp := Current Time

END  (procedure Build/Update Profile) (Return to Procedure
      Problem Description)
```

Appendix 4

```
PROCEDURE Problem Evaluation (Profile Record, Status)

(*  This procedure processes information in the Profile
    Record to generate Service Requirements for the detected
    problem.  Subsequently, service alert message Sense Data
    is generated and reported to a host system IF the Service
    Requirements indicate that the detected and evaluated
    problem requires service attention.  Problem evaluation
    information generated by this procedure is JOINED with
    the Profile Record and stored in the Problem Profile data
    base. *)

    LOAD RULE Set for Profile Record.Exception Class Group
(* Exception Class Group identifies the failing DAS function.
   The RULE Set will be executed to determine service action
   requirements. *)

EXECUTE RULE Set.Fault Isolate
   PROFILE Isolate Status INTO Profile Record.Service
   Requirements
      Isolate Certainty  = SET OF 1..10
      CE Diagnose Proc   = SET OF 1..n
      Primary FRU        = None _ SET OF 1..n
      Secondary FRU      = None _ SET OF 1..n
      Media Defect Site  = None _ Media Address (Boundary
                                  Parameter)
      Media Maint Proc   = None _ SET OF 1..n

(* RULE Set processes Profile Record data to select the
   recorded fault syndrome pattern with highest significance.
   For Example:  evidence of a power fault syndrome takes
   precedence over any logic fault syndrome that might also
   be recorded; or evidence of a logic or analog fault
   syndrome takes precedence over a data fault syndrome that
   might also be recorded.

   Minimum Isolate Status is Isolate Certainty = 1 and
   reference to a CE Diagnose Procedure.
   Maximum Isolate Status is Isolate Certainty = 10 and
   reference to a single Primary FRU or Media Defect Site. *)

EXECUTE RULE Set.Fault Severity
   PROFILE Severity Status INTO Profile Record.Service
   Requirements
      Permanence  = SET OF (Hard, Soft)
      Persistence = SET OF (Event, Burst, Recurrent,
                    Intermittent)
      Disruption  = SET OF (None, Support, Channel, Path,
                    Device, Data)
      Scope       = SET OF 0..n

(* RULE Set processes Profile Record data to determine
```

```
      Severity Status for the fault syndrome determined by Fault
      Isolate.
      Severity Status is added to the Profile Record. *)

   EXECUTE RULE Set.Service Action
      PROFILE Service Status INTO Profile Record.Service
      Requirements
         Service Status = SET OF (Probable Fault, Repair Fault)

   (* This RULE Set processes the procedure input parameters
      (New Problem, Recurrent Problem, Repeat Alert, Repeat
      Call), along with Isolate Status and Severity Status so as
      to determine the course of action for this Profile Record.
      *)

   CASE Service Status OF
      Probable Fault
         UPDATE Profile Record (Problem Status := Analysis In
         Process)
            (* The service alert message process for the Current
             Record is terminated by returning the Profile Record
             to the Problem Profile data base.  If the problem
             recurs, the recursive process of update and
             re-evaluation is invoked again.  If the problem does
             not recur within a specified period, then the
             Profile Record is purged from the Problem Profile
             data base. *)

      Repair Fault
         UPDATE Profile Record (Problem Status := Problem
         Reported)
         Generate service alert message (Profile Record.Service
         Requirements)
            JOIN service alert message Sense Data INTO Profile
            Record.Service Alert Message
         Report service alert message to host (Profile
         Record.Service Alert Message)

           (* The service alert message process for the Current
            Record is terminated by generating a service alert
            message Sense Data record from information in the
            Profile Record.  Service alert message Sense Data is
            then sent to a host system so as to initiate the
            machine service process.  The reporting process is
            repeated at a specified interval until service
            action is initiated, or until the a specified
            reporting window has expired. *)
      END
   END. (Procedure Problem Evaluation)
```

Claims

1. A method of automatically detecting and analyzing exception events occurring in a computer peripheral subsystem connected to a host computer system, the method being characterised by the steps of:

```
      Severity Status for the fault syndrome determined by Fault
      Isolate.
      Severity Status is added to the Profile Record. *)

   EXECUTE RULE Set.Service Action
      PROFILE Service Status INTO Profile Record.Service
      Requirements
         Service Status = SET OF (Probable Fault, Repair Fault)

   (* This RULE Set processes the procedure input parameters
      (New Problem, Recurrent Problem, Repeat Alert, Repeat
      Call), along with Isolate Status and Severity Status so as
      to determine the course of action for this Profile Record.
      *)

   CASE Service Status OF
      Probable Fault
         UPDATE Profile Record (Problem Status := Analysis In
         Process)
            (* The service alert message process for the Current
             Record is terminated by returning the Profile Record
             to the Problem Profile data base.  If the problem
             recurs, the recursive process of update and
             re-evaluation is invoked again.  If the problem does
             not recur within a specified period, then the
             Profile Record is purged from the Problem Profile
             data base. *)

      Repair Fault
         UPDATE Profile Record (Problem Status := Problem
         Reported)
         Generate service alert message (Profile Record.Service
         Requirements)
            JOIN service alert message Sense Data INTO Profile
            Record.Service Alert Message
         Report service alert message to host (Profile
         Record.Service Alert Message)

           (* The service alert message process for the Current
            Record is terminated by generating a service alert
            message Sense Data record from information in the
            Profile Record.  Service alert message Sense Data is
            then sent to a host system so as to initiate the
            machine service process.  The reporting process is
            repeated at a specified interval until service
            action is initiated, or until the a specified
            reporting window has expired. *)
      END
   END. (Procedure Problem Evaluation)
```

Claims

1. A method of automatically detecting and analyzing exception events occurring in a computer peripheral subsystem connected to a host computer system, the method being characterised by the steps of:

maintaining a Subsystem Environment data base comprising information relating to the current configuration of said computer peripheral subsystem and usage and error information relating to said peripheral subsystem;

conditionally invoking a Problem Description procedure when the data in said Subsystem Environment data base reflects an exception event that indicates either a clear or a probable need for service attention to said peripheral subsystem, said Problem Description procedure comprising:

developing a current Problem Profile record describing said exception event;

creating from stored rules a view of a Problem Profile data base relevant to said exception event;

scanning a Problem Profile data base comprising Problem Profile records of previously occurring exception events to relate said current Problem Profile record to a Problem Profile record that already exists in said Problem Profile data base;

if said current Problem Profile record matches an existing Problem Profile record in said Problem Profile data base, and said existing Problem Profile record indicates that a service action is pending for the covered problem, updating said existing Problem Profile record;

if said current Problem Profile record matches an existing Problem Profile record in said Problem Profile data base, and said existing Problem Profile record indicates that a service action is not pending for the covered problem, updating said existing Problem Profile record and invoking a Problem Evaluation procedure;

if said current Problem Profile record does not match an existing Problem Profile record in said Problem Profile data base, adding said current Problem Profile record to said Problem Profile data base and invoking a Problem Evaluation procedure;

wherein said Problem Evaluation procedure comprises:

examining said current Problem Profile record in view of stored rules relating to acceptable subsystem performance; and

if operations of said peripheral subsystem have been disrupted or degraded beyond the limits of acceptable subsystem performance, generating a service alert message, and inserting said service alert message into said Problem Profile record, updating said existing Problem Profile record in said Problem Profile data base, and transmitting said service alert message to said host system.

2. The method of Claim 1, wherein said step of conditionally invoking said Problem Description procedure comprises examining said Subsystem Environment data base in view of rules describing normal performance and possible exception events.

3. The method of Claim 2, wherein said step of conditionally invoking said Problem Description procedure additionally comprises examining the data regarding said exception event to determine if said exception event is a soft event or a hard event, and wherein:

if said exception event is a hard event, invoking said Problem Description procedure; and

if said exception event is a soft event, examining said exception event against a set of rules to determine if a threshold criterion has been exceeded, and invoking said Problem Description procedure if said threshold criterion has been exceeded.

4. The method of Claim 3, wherein said step of examining said exception event against a set of rules to determine if a threshold criterion has been exceeded comprises examining said exception event against a subset of rules relating to the type of exception event and the portion of said computer peripheral subsystem relating to said exception event.

5. The method of Claim 4, wherein determining if a threshold criterion has been exceeded comprises determining that a trend of related exception events have been detected within a common portion of said computer peripheral subsystem.

6. A system for automatically detecting and analyzing exception events occurring in a computer peripheral subsystem connected to a host computer system, the system being characterised by:

means for maintaining a Subsystem Environment data base comprising information relating to the current configuration of said computer peripheral subsystem and usage and error information relating to said peripheral subsystem;

means for conditionally invoking a Problem Description procedure means when the data in said Subsystem Environment data base reflects an exception event that indicates either a clear or a probable need for service attention to said peripheral subsystem, said Problem Description procedure means com-

* C L M S *

prising:

means for developing a current Problem Profile record describing said exception event;

means for creating from stored rules a view of a Problem Profile data base relevant to said exception event ;

means for scanning a Problem Profile data base comprising Problem Profile records of previously occurring exception events to relate said current Problem Profile record to a Problem Profile record that already exists in said Problem Profile data base;

if said current Problem Profile record matches an existing Problem Profile record in said Problem Profile data base, and said existing Problem Profile record indicates that a service action is pending for the covered problem, updating said existing Problem Profile record;

if said current Problem Profile record matches an existing Problem; Profile record in said Problem Profile data base, and said existing Problem Profile record indicates that a service action is not pending for the covered problem, updating said existing Problem Profile record and invoking a Problem Evaluation procedure means;

if said current Problem Profile record does not match an existing Problem Profile record in said Problem Profile data base, adding said current Problem Profile record to said Problem Profile data base and invoking a Problem Evaluation procedure means;

wherein said Problem Evaluation procedure means comprises:

means for examining said current Problem Profile record in view of stored rules relating to acceptable subsystem performance; and

means for determining if operations of said peripheral subsystem have been disrupted or degraded beyond the limits of acceptable subsystem performance, means for generating a service alert message, and inserting said service alert message into said Problem Profile record, updating said existing Problem Profile record in said Problem Profile data base, and transmitting said service alert message to said host system.

7. A system as claimed in Claim 6, wherein said means for conditionally invoking said Problem Description procedure comprises means for examining said Subsystem Environment data base in view of rules describing normal performance and possible exception events.

8. A system as claimed in either of claims 6 or 7, wherein said means of conditionally invoking said Problem Description procedure additionally comprises means for examining the data regarding said exception event to determine if said exception event is a soft event or a hard event, and

wherein:

if said exception event is a hard event, invoking said Problem Description procedure means; and

if said exception event is a soft event, examining said exception event against a set of rules to determine if a threshold criterion has been exceeded, and invoking said Problem Description procedure means if said threshold criterion has been exceeded.

Patentansprüche

1. Ein Verfahren zur automatischen Erfassung und Analyse von Ausnahmeereignissen, die in dem peripheren Subsystem eines Rechners auftreten, das an ein Host-Rechnersystem angeschlossen ist, wobei das Verfahren durch folgende Schritte gekennzeichnet ist:

Pflegen einer Subsystemumgebungs-Datenbank, die Informationen bezüglich der aktuellen Konfiguration des genannten peripheren Rechner-Subsystems sowie Nutzungs- und Fehlerinformationen bezüglich des genannten peripheren Subsystems enthält;

bedingungsabhängiges Aufrufen einer Problembeschreibungsprozedur, wenn die Daten in der genannten Subsystemumgebungs-Datenbank ein Ausnahmeereignis wiedergeben, das entweder eine eindeutige oder eine wahrscheinliche Notwendigkeit für eine Dienstmaßnahme an dem genannten peripheren Subsystem anzeigt, wobei die genannte Problembeschreibungsprozedur folgendes umfaßt:

Entwicklung eines aktuellen Problemprofil-Datensatzes, der das genannte Ausnahmeereignis beschreibt;

Erzeugen einer Betrachtung einer Problemprofil-Datenbank für das genannte Ausnahmeereignis aus gespeicherten Regeln;

Abfragen einer Problemprofil-Datenbank, die Problemprofil-Datensätze von bereits früher aufgetretenen Ausnahmeereignissen enthält, um den genannten aktuellen Problemprofil-Datensatz mit einem Problem-

* C L M S *

profil-Datensatz zu vergleichen, der bereits in der genannten Problemprofil-Datenbank vorhanden ist;
wenn der genannte aktuelle Problemprofil-Datensatz mit einem bereits in der genannten Problemprofil-Datenbank vorhandenen Problemprofil-Datensatz übereinstimmt, und der genannte bereits vorhandene Problemprofil-Datensatz anzeigt, daß eine Dienstmaßnahme für das betreffende Problem bereits ansteht, Aktualisieren des genannten vorhandenen Problemprofil-Datensatzes;
wenn der genannte aktuelle Problemprofil-Datensatz mit einem in der genannten Problemprofil-Datenbank bereits vorhandenen Problemprofil-Datensatz übereinstimmt, und der genannte bereits vorhandene Problemprofil-Datensatz anzeigt, daß eine Dienstmaßnahme für das betreffende Problem nicht ansteht, Aktualisieren des genannten vorhandenen Problemprofil-Datensatzes und Aufrufen einer Problemauswertungsprozedur;
wenn der genannte aktuelle Problemprofil-Datensatz nicht mit einem in der genannten Problemprofil-Datenbank bereits vorhandenen Problemprofil-Datensatz übereinstimmt, Hinzufügen des genannten Problemprofil-Datensatzes zu der genannten Problemprofil-Datenbank und Aufrufen einer Problemauswertungsprozedur;
wobei die genannte Problemauswertungsprozedur folgendes umfaßt:
Prüfen des genannten aktuellen Problemprofil-Datensatzes im Hinblick auf gespeicherte Regeln, die sich auf eine akzeptable Subsystem-Leistung beziehen; und
wenn die Funktionen des genannten peripheren Subsystems unterbrochen wurden oder sich soweit verschlechtert haben, daß bestimmte Grenzwerte im Hinblick auf eine akzeptable Subsystem-Leistung überschritten sind, Erzeugen einer Dienstwarnmeldung und Einfügen der genannten Dienstwarnmeldung in den genannten Problemprofil-Datensatz, Aktualisieren des genannten vorhandenen Problemprofil-Datensatzes in der genannten Problemprofil-Datenbank und Übertragen der genannten Dienstwarnmeldung zu dem genannten Host-System.

2. Die Methode nach Anspruch 1, bei der der genannte Schritt des bedingungsabhängigen Aufrufens der genannten Problembeschreibungsprozedur die Prüfung der genannten Subsystemumgebungs-Datenbank im Hinblick auf Regeln umfaßt, die eine normale Leistung und mögliche Ausnahmeereignisse beschreiben.

3. Das Verfahren nach Anspruch 2, bei dem der genannte Schritt des bedingungsabhängigen Aufrufens der genannten Problembeschreibungsprozedur außerdem das Prüfen der Daten umfaßt, die sich auf das genannte Ausnahmeereignis beziehen, um festzustellen, ob es sich bei dem genannten Ausnahmeereignis um ein weiches oder ein hartes Ereignis handelt, und bei dem:
wenn das genannte Ausnahmeereignis ein hartes Ereignis ist, die genannte Problembeschreibungsprozedur aufgerufen wird; und
wenn das genannte Ausnahmeereignis ein weiches Ereignis ist, das genannte Ausnahmeereignis geprüft und mit einem Regelsatz verglichen wird, um festzustellen, ob ein Schwellenkriterium überschritten wurde, und die genannte Problembeschreibungsprozedur aufgerufen wird, wenn das genannte Schwellenkriterium überschritten wurde.

4. Das Verfahren nach Anspruch 3, bei dem der genannte Schritt des Prüfens des genannten Ausnahmeereignisses im Vergleich mit einem Regelsatz, um festzustellen, ob ein Schwellenkriterium überschritten wurde, das Prüfen des genannten Ausnahmeereignisses im Vergleich mit einem Regel-Nebensatz umfaßt, der sich auf den Typ des Ausnahmeereignisses und den Abschnitt des genannten peripheren Rechner-Subsystems beziehen, der sich auf das genannte Ausnahmeereignis bezieht.

5. Das Verfahren nach Anspruch 4, bei dem das Feststellen, ob ein Schwellenkriterium überschritten wurde, die Feststellung umfaßt, ob ein Trend von miteinander verglichenen Ausnahmeereignissen in einem gemeinsamen Abschnitt des genannten peripheren Rechner-Subsystems erkannt wurde.

6. Ein System zur automatischen Erkennung und Analyse von Ausnahmeereignissen, die in einem peripheren Rechner-Subsystem auftreten, das an ein Host-Rechnersystem angeschlossen ist, wobei das System durch folgende Schritte gekennzeichnet ist:
Mittel zur Pflege einer Subsystemumgebungs-Datenbank, die Informationen bezüglich der aktuellen Konfiguration des genannten peripheren Rechner-Subsystems sowie Nutzungs- und Fehlerinformationen bezüglich des genannten peripheren Subsystems enthält;
Mittel zum bedingungsabhängigen Aufrufen eines Problembeschreibungsprozedur-Mittels, wenn die Daten in der genannten Subsystemumgebungs-Datenbank auf ein Ausnahmeereignis hinweisen, das ent-

weder eine eindeutige oder eine wahrscheinliche Notwendigkeit einer Dienstmaßnahme an dem genannten peripheren Subsystem anzeigt, wobei das genannte Problembeschreibungsprozedur-Mittel folgendes umfaßt:
Mittel zur Entwicklung eines aktuellen Problemprofil-Datensatzes, der das genannte Ausnahmeereignis beschreibt;
Mittel zur Erzeugung einer Betrachtung einer Problemprofil-Datenbank bezüglich des genannten Ausnahmeereignisses aus gespeicherten Regeln;
Mittel zum Abfragen einer Problemprofil-Datenbank, die Problemprofil-Datensätze von bereits früher aufgetretenen Ausnahmeereignissen enthält, um den genannten aktuellen Problemprofil-Datensatz mit einem Problemprofil-Datensatz zu vergleichen, der bereits in der genannten Problemprofil-Datenbank vorhanden ist;
wenn der genannte aktuelle Problemprofil-Datensatz mit einem in der genannten Problemprofil-Datenbank vorhandenen Problemprofil-Datensatz übereinstimmt, und der genannte vorhandene Problemprofil-Datensatz anzeigt, daß für das betreffende Problem eine Dienstmaßnahme ansteht, Aktualisieren des genannten vorhandenen Problemprofil-Datensatzes;
wenn der genannte aktuelle Problemprofil-Datensatz mit einem vorhandenen Problemprofil-Datensatz in der genannten Problemprofil-Datenbank übereinstimmt, und der genannte vorhandene Problemprofil-Datensatz anzeigt, daß für das betreffende Problem keine Dienstmaßnahme ansteht, Aktualisieren des genannten vorhandenen Problemprofil-Datensatzes und Aufrufen eines Problemauswertungsprozedur-Mittels;
wenn der genannte aktuelle Problemprofil-Datensatz nicht mit einem in der genannten Problemprofil-Datenbank bereits vorhandenen Problemprofil-Datensatz übereinstimmt, Hinzufügen des genannten aktuellen Problemprofil-Datensatzes zu der genannten Problemprofil-Datenbank und Aufrufen eines Problemauswertungsprozedur-Mittels;
wobei das genannte Problemauswertungsprozedur-Mittel folgendes umfaßt:
Mittel zum Prüfen des genannten aktuellen Problemprofil-Datensatzes im Hinblick auf gespeicherte Regeln, die sich auf eine akzeptable Subsystem-Leistung beziehen; und
Mittel, um festzustellen, ob Funktionen des genannten peripheren Subsystems unterbrochen wurden oder sich soweit verschlechtert haben, daß eine bestimmte Grenze im Hinblick auf eine akzeptable Subsystem-Leistung überschritten wurde, Mittel zum Erzeugen einer Dienstwarnmeldung und Einfügen der genannten Dienstwarnmeldung in den genannten Problemprofil-Datensatz, Aktualisieren des genannten Problemprofil-Datensatzes in der genannten Problemprofil-Datenbank und Übertragen der genannten Dienstwarnmeldung an das genannte Host-System.

7. Ein System nach Anspruch 6, bei dem das genannte Mittel zum bedingungsabhängigen Aufrufen der genannten Problembeschreibungsprozedur Mittel zum Prüfen der genannten Subsystemumgebungs-Datenbank im Hinblick auf Regeln umfaßt, die eine normale Leistung und mögliche Ausnahmeereignisse beschreiben.

8. Ein System nach jedem der Ansprüche 6 oder 7, bei dem das genannte Mittel zum bedingungsabhängigen Aufrufen der genannten Problembeschreibungsprozedur zusätzlich Mittel zur Prüfung der Daten des genannten Ausnahmeereignisses umfaßt, um festzustellen, ob es sich bei dem genannten Ausnahmeereignis um ein weiches oder ein hartes Ereignis handelt, und bei dem:
wenn das genannte Ausnahmeereignis ein hartes Ereignis ist, das genannte Problembeschreibungsprozedur-Mittel aufgerufen wird; und
wenn das genannte Ausnahmeereignis ein weiches Ereignis ist, das genannte Ausnahmeereignis mit einem Regelsatz verglichen wird, um festzustellen, ob ein Schwellenkriterium überschritten wurde, und, wenn das genannte Schwellenkriterium überschritten wurde, Aufrufen der genannten Problembeschreibungsprozedur-Mittels.

Revendications

1. Méthode pour détecter et analyser automatiquement des événements exceptionnels survenant dans un sous-système périphérique d'ordinateur connecté à un système ordinateur hôte, la méthode étant caractérisée par les étapes de :
maintenir une base de données d'environnement de sous-système comprenant des informations relatives à la configuration courante dudit sous-système périphérique d'ordinateur et des informations

* C L M S *

d'usage et d'erreurs relatives audit sous-système périphérique,

appeler de façon conditionnelle une procédure de description de problème lorsque les données dans ladite base de données d'environnement de sous-système reflètent un événement exceptionnel qui indique un besoin clair ou probable d'un service audit sous-système périphérique, ladite procédure de description de problème comprenant :

le développement d'un enregistrement de profil de problème courant décrivant ledit événement exceptionnel,

la création à partir de règles emmagasinées d'une vue d'une base de données de profils de problèmes d'intérêt pour ledit événement exceptionnel,

le balayage d'une base de données de profils de problèmes comprenant des enregistrements de profils de problèmes d'événements exceptionnels survenus précédemment pour faire correspondre ledit enregistrement de profil de problème courant à un enregistrement de profil de problème qui existe déjà dans ladite base de données de profils de problèmes,

si ledit enregistrement de profil de problème courant correspond à un enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, et ledit enregistrement de profil de problème existant indique qu'une action de service est en attente pour le problème couvert, mettre à jour ledit enregistrement de profil de problème existant,

si ledit enregistrement de profil de problème courant correspond à un enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, et ledit enregistrement de profil de problème existant indique qu'une action de service n'est pas en attente pour le problème couvert, mettre à jour ledit enregistrement de profil de problème existant et appeler une procédure d'évaluation de problème,

si ledit enregistrement de profil de problème courant ne correspond pas à un enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, ajouter ledit enregistrement de profil de problème courant à ladite base de données de profils de problèmes et appeler une procédure d'évaluation de problème,

dans laquelle ladite procédure d'évaluation de problème comprend :

l'examen dudit enregistrement de profil de problème courant en vue de règles emmagasinées concernant une mise en oeuvre acceptable du sous-système, et

si les opérations dudit sous-système périphérique ont été interrompues ou dégradées au-delà des limites de performances acceptables du sous-système, générer un message d'intervention et insérer ledit message d'intervention dans ledit enregistrement de profil de problème, mettre à jour ledit enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, et transmettre ledit message d'intervention audit système hôte.

2. Méthode selon la revendication 1, dans laquelle ladite étape d'appeler de façon conditionnelle ladite procédure de description de problème comprend l'examen dans ladite base de données d'environnement de sous-système des règles décrivant la mise en oeuvre normale et les événements exceptionnels possibles.

3. Méthode selon la revendication 2, dans laquelle ladite étape d'appeler de façon conditionnelle ladite procédure de description du problème comporte l'examen des données relatives audit événement exceptionnel pour déterminer si ledit événement exceptionnel est un événement bénin ou un événement grave, et dans laquelle

si ledit événement exceptionnel est un événement grave, appeler ladite procédure de description de problème et,

si ledit événement exceptionnel est un événement bénin, examiner ledit événement exceptionnel par rapport à un ensemble de règles pour déterminer si un critère de seuil a été dépassé, et appeler ladite procédure de description de problème si ledit critère de seuil a été dépassé.

4. Méthode selon la revendication 3, dans laquelle ladite étape d'examiner ledit événement exceptionnel par rapport à un ensemble de règles pour déterminer si un critère de seuil a été dépassé comprend l'examen dudit critère de seuil par rapport à un ensemble de règles relatives au type d'événement exceptionnel et à la partie dudit sous-système périphérique d'ordinateur en rapport avec ledit événement exceptionnel.

5. Méthode selon la revendication 4, dans laquelle la détermination qu'un critère de seuil a été dépassé comprend la détermination qu'une tendance d'événements exceptionnels en relation a été détectée dans une portion commune dudit sous-système périphérique d'ordinateur.

* C L M S *

6. Système pour automatiquement détecter et analyser des événements exceptionnels survenant dans un sous-système périphérique d'ordinateur connecté à un système ordinateur hôte, le système étant caractérisé par :

des moyens pour maintenir une base de données d'environnement de sous-système comprenant des informations relatives à la configuration courante dudit sous-système périphérique d'ordinateur et des informations d'usage et d'erreurs relatives audit sous-système périphérique,

des moyens pour appeler de façon conditionnelle des moyens de procédure de description de problème lorsque les données dans ladite base de données d'environnement de sous-système reflètent un événement exceptionnel qui indique un besoin clair ou probable d'un service audit sous-système périphérique, lesdits moyens de procédure de description de problème comprenant :

des moyens pour développer un enregistrement de profil de problème courant décrivant ledit événement exceptionnel,

des moyens pour créer à partir de règles emmagasinées une vue d'une base de données de profils de problèmes d'intérêt pour ledit événement exceptionnel,

des moyens pour balayer une base de données de profils de problèmes comprenant des enregistrements de profils de problèmes d'événements exceptionnels survenus précédemment pour faire correspondre ledit enregistrement de profil de problème courant à un enregistrement de profil de problème qui existe déjà dans ladite base de données de profils de problèmes,

si ledit enregistrement de profil de problème courant correspond à un enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, et ledit enregistrement de profil de problème existant indique qu'une action de service est en attente pour le problème couvert, mettre à jour ledit enregistrement de profil de problème existant,

si ledit enregistrement de profil de problème courant correspond à un enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, et ledit enregistrement de profil de problème existant indique qu'une action de service n'est pas en attente pour le problème couvert, mettre à jour ledit enregistrement de profil de problème existant et appeler une procédure d'évaluation de problème,

si ledit enregistrement de profil de problème courant ne correspond pas à un enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, ajouter ledit enregistrement de profil de problème courant à ladite base de données de profils de problèmes et appeler une procédure d'évaluation de problème,

dans laquelle ladite procédure d'évaluation de problème comprend :

des moyens pour examiner ledit enregistrement de profil de problème courant en vue de règles emmagasinées concernant une mise en oeuvre acceptable du sous-système, et

des moyens pour déterminer si les opérations dudit sous-système périphérique ont été interrompues ou dégradées au-delà des limites de performances acceptables du sous-système, des moyens pour générer un message d'intervention et insérer ledit message d'intervention dans ledit enregistrement de profil de problème, mettre à jour ledit enregistrement de profil de problème existant dans ladite base de données de profils de problèmes, et transmettre ledit message d'intervention audit système hôte.

7. Système tel que revendiqué dans la revendication 6, dans lequel lesdits moyens pour appeler de façon conditionnelle ladite procédure de description de problème comprennent des moyens d'examen dans ladite base de données d'environnement de sous-système des règles décrivant la mise en oeuvre normale et les événements exceptionnels possibles.

8. Système tel que revendiqué dans l'une des revendications 6 ou 7, dans lequel lesdits moyens pour appeler de façon conditionnelle ladite procédure de description du problème comportent des moyens d'examen des données relatives audit événement exceptionnel pour déterminer si ledit événement exceptionnel est un événement bénin ou un événement grave, et dans lequel

si ledit événement exceptionnel est un événement grave, on appelle ladite procédure de description de problème et,

si ledit événement exceptionnel est un événement bénin, on examine ledit événement exceptionnel par rapport à un ensemble de règles pour déterminer si un critère de seuil a été dépassé, et on appelle ladite procédure de description de problème si ledit critère de seuil a été dépassé.