



SEMI E53-96

EVENT REPORTING

1 Purpose

Access to process data in equipment is crucial for effective process monitoring and control in a semiconductor manufacturing facility. This standard addresses the communication needs of semiconductor equipment and other factory objects, such as cell controllers or recipe servers, with respect to the timely collection and reporting of such data.

The purpose of this standard is to provide a general purpose set of event reporting services that may be offered by equipment suppliers. This document may be referenced, in whole or in part, by other standards addressing higher level application domains.

The communications services defined here will enable standards based inter-operability of independent systems. They shall allow application software to be developed which can assume the existence of these services and allow software products to be developed which offer them.

2 Scope

This standard is applicable to any stand-alone equipment, cluster module, cluster tool, or cell of automation in a factory. As such it addresses event reporting at all levels in the factory and equipment control hierarchy.

This standard requires significant communication and computational resources and is therefore not applicable at or below the level of I/O distribution (e.g., sensor bus) within the equipment.

This standard covers the reporting of data periodically and/or in response to events. Reports may also be requested on demand.

This standard presents a solution from the concepts and behavior down to the messaging services. It does not define the messaging protocol.

A messaging service includes the identification that a message shall be exchanged and definition of the data which is contained in that message. It does not include information on the structure of the message, how the data is represented within the message, or how the message is exchanged. This additional information is contained with the message protocol. The defined ser-

vices may be applied to multiple protocols. Information on the mapping of these services to special protocols (e.g., SECS II) are added as adjunct standards.

The services assume a communications environment in which a reliable connection has been established between the user and the provider of the services. Establishing, maintaining, and releasing a connection is beyond the scope of this standard.

3 Referenced Documents

3.1 SEMI Standards

SEMI E5 — SEMI Equipment Communications Standard 2 Message Content (SECS II)

SEMI E30 — Generic Model for Communications and Control of SEMI Equipment (GEM)

SEMI E38 — Cluster Tool Module Communications (CTMC)

SEMI E39 — Object Services Standard: Concepts, Behavior, and Services

SEMI E40 — Standard for Processing Management

4 Terminology

4.1 The following definitions are arranged in alphabetical order. Some terms are defined using terms defined elsewhere within this section. Other terms may be defined within SEMI's Compilation of Terms.

4.1.1 *attribute* — A data item associated with an object. An attribute may be referenced by zero or more data reports.

4.1.2 *behavior* — The manner in which something functions; how an object acts and reacts, in terms of its state changes and message passing.

4.1.3 *collection event* — An event that may be used to initiate the collection and reporting of data. A collection event may trigger an event report. A collection event may also start or stop one or more trace reports.

4.1.4 *data report* — A data report is a list of attribute names for a single object. Data reports may be predefined by a factory object or defined dynamically by the service user.



4.1.5 *default object* — The object assumed when no object specifier is supplied.

4.1.6 *event* — Represents the occurrence of a change in the condition of a system (e.g., lot complete, temperature over range).

4.1.7 *event report* — A class of objects that has information related to an event and can be linked to user defined data reports and can send messages containing this information to a service user.

4.1.8 *factory object* — Any identifiable object within the factory information and control architecture. Examples include equipment, a cluster process module, a cell controller, a recipe namespace server.

4.1.9 *object* — Defined in the Object Services (SEMI E39).

4.1.10 *object specifier* — Defined in the Object Services (SEMI E39).

4.1.11 *services* — A set of closely related messages.

4.1.12 *service provider* — A service provider is an application responsible for providing services to service users.

NOTE: There may be one or more service users concurrently accessing a single service provider. It is the responsibility of the service provider to provide its services transparently to each service user.

4.1.13 *service user* — A service user is any application that uses the services.

4.1.14 *trace report* — A class of objects which provides to a service user a means for collecting periodic readings of selected attributes of a system.

4.2 Data Types

4.2.1 *form* — Type of data: float, positive integer, unsigned integer, integer, enumerated, boolean, text, formatted text, structure, list, ordered list.

4.2.2 *float* — A number represented by a mantissa and an exponent. It is used to represent numeric data which is continuous in value.

4.2.3 *positive integer* — May take the value of any positive whole number. Messaging protocol may impose a limit on the range of possible values.

4.2.4 *unsigned integer* — May take the value of any positive integer or zero. Messaging protocol may impose a limit on the range of possible values.

4.2.5 *integer* — May take the value of any negative or unsigned number. Messaging protocol may impose a limit on the range of possible values.

4.2.6 *enumerated* — May take on one of a limited set of possible values. These values may be given logical names, but they may be represented by any single-item data type.

4.2.7 *boolean* — May take on one of two possible values; TRUE or FALSE.

4.2.8 *text* — A text string. Messaging protocol may impose restrictions, such as length or ASCII representation.

4.2.9 *formatted text* — A text string with an imposed format. This could be by position, by use of special characters, or both.

4.2.10 *structure* — A complex structure consisting of a specific set of items, of possibly mixed data types, in a specific arrangement.

4.2.11 *list* — Set of one or more items that are all of the same form (one of the above forms).

4.2.12 *ordered list* — A list for which the order in which the items appear is significant.

5 Conventions

5.1 *Harel State Model* — This document uses the Harel State Chart notation to describe the dynamic behavior of the objects defined. An overview of this notation is presented in an appendix of SEMI E30. The formal definition of this notation is presented in Science of Computer Programming 8, "Statecharts: A Visual Formalism for Complex Systems", by D. Harel, 1987.

Transition tables are provided in conjunction with the state diagrams to explicitly describe the nature of each state transition. A transition contains columns for transition #, current state, trigger, new state, action(s). The "trigger" (column 3) for the transition occurs while in the "current" state. The "actions" (column 5) include a combination of; 1) actions taken upon exit of the current state, 2) actions taken upon entry of the new state, and 3) actions taken which are most closely associated with the transition. No differentiation is made.

5.2 *OMT Object Information Model* — The object models are presented using the Object Modeling Technique developed by Rumbaugh, James, et al, in "Object-Oriented Modeling and Design", Prentice



Hall, Englewood Cliffs, NJ, ©1991. Overviews of this notation are provided in an appendix of the Object Services SEMI E39.

5.3 Object Attribute Representation — The object information models for standardized objects will be supported by an attribute definition table with the following column headings:

Attribute	Definition	Rqmt	Access	Form
The formal text name of the attribute	Description of the information contained	Y or N	RO or RW	(see below)

The access column uses RO (read only) or RW (read and write) to indicate the access that users of object services have to the attribute.

A 'Y' or 'N' in the requirement (rqmt) column indicates if this attribute must be supported in order to meet fundamental compliance for the service.

The form column is used to indicate the format of the attribute. See Section 4.2 for definitions.

5.4 Service Message Representation

5.4.1 Service Resource Definition — The service resource definition table defines the specific set of messages for a given service group, as shown in the following table:

Service	Type	Description
Message name	N or R	The intent of the service

Type can be either notification (N) or request (R). Notification messages are initiated by the service provider. No response is expected. Request messages are initiated by the service user. Request messages ask for data or for an operation to be performed. Request messages expect a specific response (no presumption on the message content).

5.4.2 Service Parameter Dictionary — Each parameter should relate to either attributes from the object model or events of the dynamic model (Harel State Chart). All parameters to the services are listed in a single table or dictionary. The column headings for this parameter dictionary are as follows:

Parameter	Description	Form
Parameter X	Data type	A parameter

A row is provided in the table for each parameter of the service. The first column contains the name of the parameter. This is followed by columns describing the form and the contents of the parameter.

The description column describes the meaning of the parameter and interrelationships with other parameters.

The form column is used to indicate the type of data contained in the parameter (see Section 4.2 for definitions) and the possible values it may take on.

To prevent definition of numerous parameters named "XxxList", this document adopts the convention of referring to the list as "(list of) Xxx". In this case, the definition of the parameter "Xxx" will be given, not of the list. Where a list is used in both the request and the response, the list order in the request is retained in the response. A list must contain at least one element unless zero elements are specifically allowed.

5.4.3 Service Message Definition — There is a table for each service showing the parameter detail. The column headings for the service detail are as follows:

Parameter	Req/ Ind	Rsp/ Cnf	Comment
Parameter X	(see below)	(see below)	A description of the service.

The columns labeled req/ind and rsp/cnf link the parameters to the direction of the message. The message sent by the initiator is called the "request" (req). The receiver terms the message the "indication" (ind). The receiver may then send a "response" (rsp), which the original sender terms the "confirmation" (cnf).

The request (req/ind) and response (rsp/cnf) entries can take on the following values:

"M"	Mandatory parameter — Must be given a valid value.
"C"	Conditional parameter — May be defined in some circumstances and undefined in others. Whether a value is given may be completely optional or may depend on the value of another parameter.
"U"	User-defined parameter
"."	The parameter is not used.
"="	The entries M and C in the response can be modified with (=) to indicate that the value in the response must match the request.

6 Overview

Event reporting provides a dynamic and flexible means by which the service user can receive notification of events and data relating to objects defined here and in other SEMI standards.

Figure 1 illustrates a simple event reporting situation.

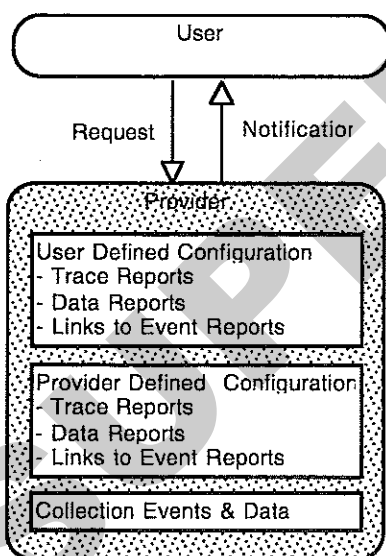


Figure 1
Simple Event Reporting

Event and variable identifiers may be defined by another standard with which equipment or an object complies. In either case, the definitions must be included in the documentation from the supplier.

The service user may define and access data reports using the event reporting request services. Data reports are sampled and sent by the service provider using the event reporting notification services.

Data reports may be pre-defined by the supplier. Such reports cannot be deleted and their report identifiers may not be used for dynamically created reports.

6.1 Multiple Service Users — In a more complex situation, several service users may require simultaneous access to the event reporting services (ERS). In this case, the service provider manages data report definitions independently for each service user as shown in Figure 2. A copy of each pre-defined report is made for each service user. Support for more than one simultaneous service user is optional.

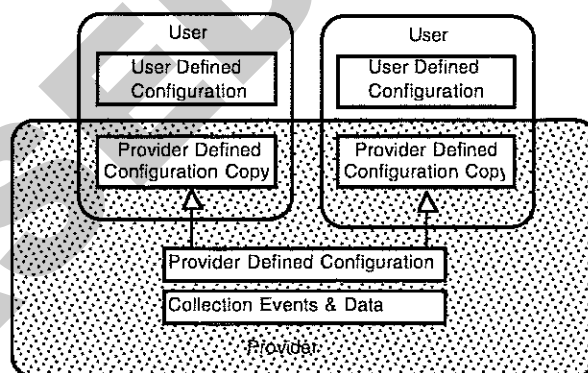


Figure 2
Multiple Service Users

6.2 Using Object Services — An event reporting service user can create reports dynamically by accessing attribute information using the object services (see SEMI E39) when they are provided (see Figure 3).

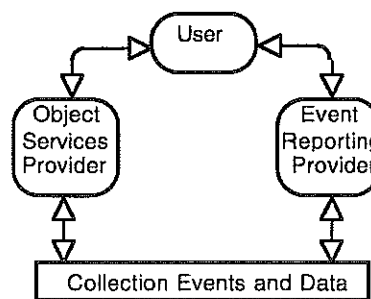


Figure 3
Using Object Services



7 Concepts

7.1 Objects and Identifiers — Much of the discussion and specification for event reporting services relies on an understanding of object technology and modeling techniques. The use of object technology for ERS implementations is not required. Further, it is not required that ERS implementations be compliant to SEMI E39 (Object Services Standard (OSS)). This standard does specify mechanisms by which ERS can take advantage of OSS when those services are provided.

Events are associated with an object and data is stored in object attributes. To uniquely identify an event or an attribute the object to which it relates must first be identified.

An object may be a physical entity such as stand-alone process equipment, a cluster tool or a cluster module. An object may be an abstraction such as a recipe, a process job or a transfer job. An object may be defined by the equipment supplier, or by a standard with which the equipment complies. An example of a cluster tool, containing process modules, is used in this section as the service provider. This example is for illustration purposes only. The same concepts apply for all factory objects as defined above.

Objects may be part of an object hierarchy or contained in other objects. The default object is the object which provides the service connection. In the case for batch processing equipment which is compliant to the GEM, SEMI E30 specification, the connection to the host is provided by the equipment and the equipment is the default object. Also, a cluster controller would be the default object of equipment containing process modules. However, from the point of view of the cluster controller, each connection to the service providers in each connected module would be the default object. For example, the processing management service of PM:ETCH would be a default object for that module connection. But, the cluster controller would have different default objects on each of its other service connections.

An object specifier, (see SEMI E39), identifies an object within the scope of the service provider. An object specifier consists of one or more object type and object instance identifier pairs. For example, "PM:ETCH>" might be the object specifier of a process module (type=PM, identifier=ETCH) within a

cluster tool. If the object specifier is omitted, then the default object is assumed.

Objects are considered local to the service provider even if they exist on a remote module. That means the service provider is responsible for detecting events and accessing data from those objects in a timely manner, by whatever means, to provide the specified services.

Equipment which implements OSS compliant event reporting services allows the service user to reference equipment components by an object specifier; for example, "PM:ETCH>MFC:CCL4".



7.2 Event Report Object Model — Figure 4 shows the objects, attributes (variables), and relationships involved in event reporting. The notation used is defined in Section 5. The diagram illustrates the relationships between the service's messages and attributes (variables). The objects and their attributes are defined Section 8. Only the trace, data report, collection event, and data source objects' attributes and behaviors are standardized in that section. Note that this model presents a view of event reporting services from the perspective a single user. It is optional for the service provider to support simultaneous access from multiple service users. But for that option the provider shall present this view to each user independently.

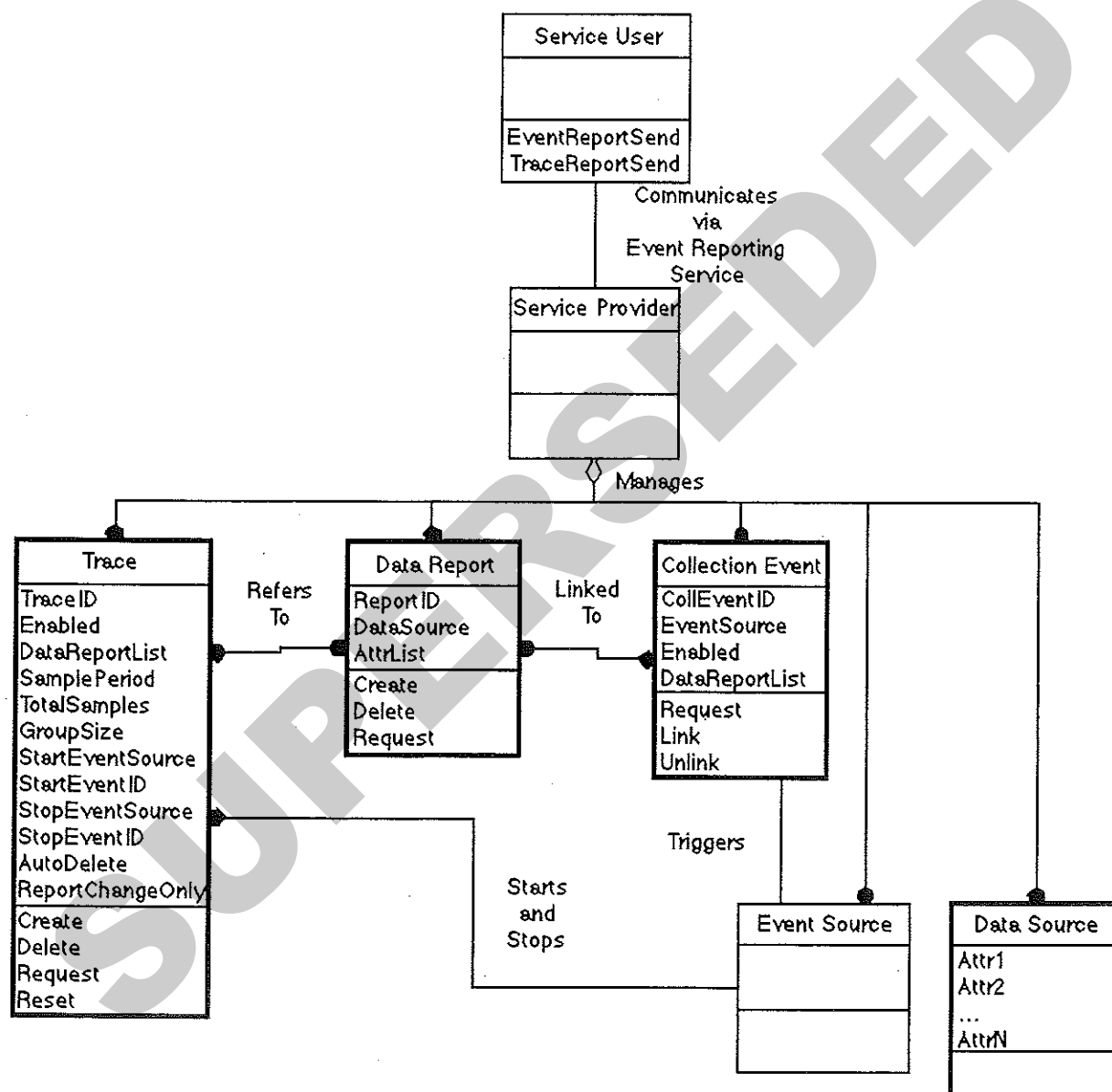


Figure 4
Event Report Object Model



7.2.1 Service User Object — The service user is responsible for configuring its own event reporting setup and receives reports from the service provider according to that setup. ERS does not specify the attributes or behavior of this object. It is included in the event report object model to provide context for other objects within ERS. However, the service user object shall be able to receive the event report send and trace report send messages from the service provider.

7.2.2 Service Provider Object — The service provider is responsible for compiling report samples according to each service user setup. It detects collection events, provides trace timing and accesses the value of data source object attributes (variables) in order to compile the data reports. ERS does not specify the attributes or behavior of this object. It is included in the event report object model to provide context for other objects within ERS.

7.2.2.1 Collection Event Object — This object type or class is responsible to provide all the collection event identifiers for which event reports can be generated. Other factory services and objects may provide event notification messages which are important in the context of a service or functionality that they provide. But, in general, these notifications do not provide mechanisms to report information beyond their domain boundaries. For example, Processing Management, SEMI E40, includes an event notification for process complete (PRJobAlert (Milestone)). In order to collect data associated with this event it must be “known” to the collection event object type. When it is “known” by the collection event object, then event reports can be generated using data or attributes from the data sources which are “known” by the event reporting service provider. A data source could provide information on process chamber pressure and temperature, which are attributes or variables important to the process but which are not known to the processing management service.

7.2.2.2 Event and Data Source Objects — The use of the event source and data source objects provides extensive capability to access the attributes of objects in a distributed environment. Where the event services provider has a domain which extends over multiple resources, such as a cell controller for a furnace bay or a cluster tool controller, these objects provide an unambiguous mechanism for addressing the data that the service user wishes to access.

The event and data source objects are to the event reporting services what directory object is in a file system of a computer's operating system. They should be thought of as containers of information which provide pointers (or names) of objects that are the actual sources of events or data.

The event source object is not directly accessible to the service user. Knowledge of events and event sources can be accumulated by interrogation of the collection event object using OSS.

Collection event identifiers shall be determined by either of two possible methods. One, the equipment supplier may define and supply with the equipment a list of collection event IDs (identifiers). Two, the equipment's collection event object can be interrogated with object services requests for collection event sources and identifiers. For example, using the second method, one could use object services to ask the collection event object for a list of collection event sources, which might include objects such as “PM:Etch>MFC:CCL4” or “TM>Aligner”. Then using OSS, the collection event object can be queried for events associated with these sources. For the MFC a list of events might include “preslow”, “overlimit”, “valvemax”, etc..

7.2.3 Data Report Object — A data report defines the variables (or attributes) of a specific object (data source) which are to be sampled. Data reports are either pre-defined or defined dynamically by the service user. A data report sample is a set of data containing the values of the variables defined for the data report. To sample a data report means that the current values of the variables at their data source object should be recorded. Data reports may be sampled and sent either with an event report and/or with a trace report. The service user may request:

- To sample an individual data report.
- Create a data report.
- Delete a data report.

7.2.4 Event Report — Event reports may be sent as messages to the service user whenever the associated collection event occurs. The collection event object is the source of the event report message. Event reporting is initially disabled. If the data to be sent with an event report has more than one data source, then there must be at least one data report for each data source to be



included in the event report link definition. The service user may request:

- Event reporting to be enabled.
- Event reporting to be disabled.
- One or more data reports to be linked (included in) the event report.

7.2.5 Trace Report — Trace reports are messages which are sent to the service user periodically. The trace report will contain the value(s) of the data item(s) specified in the data report(s) referred to by the trace object which is the source of the trace report.

The trace object contains the description of what to include in the trace report and trace objects are either pre-defined or defined dynamically by the service user. The service user may request:

- Trace reporting to be enabled.
- Trace reporting to be disabled.
- One or more trace reports to be reset at any time to their initial state.
- One or more trace reports to be deleted at any time.

When the enabled attribute of the trace object is set, then trace reporting may be started or in response to a start event. When the trace reporting starts, the sample count is reset to zero. A trace report may be stopped when a certain number of samples have been reported or when a stop event occurs.

The service user may specify a trace object to be automatically deleted once it has been stopped by which-ever mechanism. Otherwise, the trace reporting simply stops. If no start event is defined, then the trace is restarted immediately. Otherwise, the next start event will restart the trace. Disabling or deleting a trace report stops reporting immediately.

7.3 Scenarios — The communications involved in event reporting are introduced below in the form of some representative scenarios. The full behavior and message set are covered in sections 8.2 and 8.3 respectively.

7.3.1 Event Reporting — Figure 5 shows a scenario in which a data report is created by the service user and linked to an event report. Event reporting is then enabled. Subsequently, an event report is sent with the data reports attached whenever the collection event occurs. The service user then disables event reporting,

preventing subsequent collection events from triggering the report. Data reports and links to event reports may be pre-defined by the equipment in which case the DataReportCreate and LinkCreate messages are omitted.

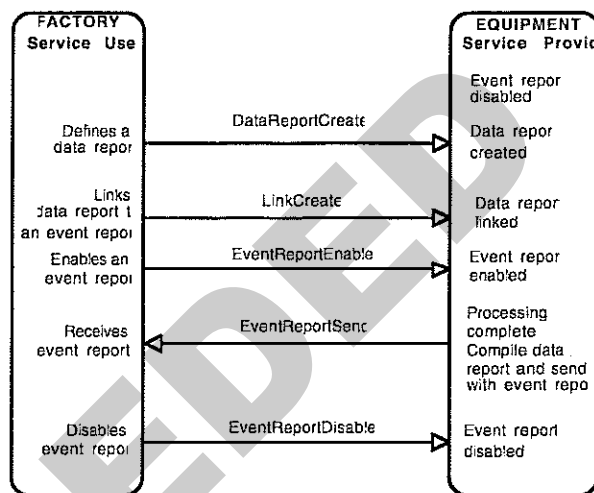


Figure 5
Event Reporting Scenario

7.3.2 Trace Reporting — Figure 6 shows a scenario in which a trace object is created with enabled and AutoDelete both set to TRUE. Since the trace object is enabled and there is no start event defined the data reports assigned to the trace object are sampled and sent immediately as a trace report message. The trace object timer is started and the sample counter set to zero. Each time the sample period elapses thereafter, a new trace report is sent, and the sample counter incremented. When the sample counter reaches the total samples parameter, then the trace report is automatically deleted.

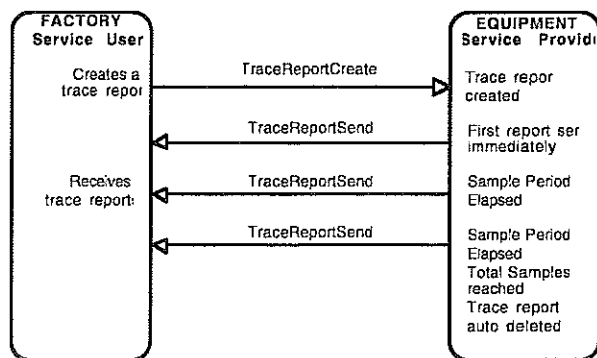


Figure 6
Trace Reporting Scenario