# SeQent
### Real-Time Visualization

*First*PAGE Alarm Manager – v9.0
Protocol Specification (BETA)

SeQent
Suite 137
4500 Blakie Road
London, Ontario
Canada
N6L 1G5

Document Date: September 24, 2009

# Edits

| Version | Date | Author | Description |
|---------|------|--------|-------------|
| V9.0 | November 24, 2006 | S. Burns, C. Longman, D. Vlasman | Initial release<br>– ADMIN;REGISTER/CLEAR/LIST<br>- Define packet carriage control<br>- Defined %TIPCOUNT% value<br>- Clarified ENABLE/DISABLE of events for ANS |

# Table of Contents

# C h a p t e r   O n e
## Protocol Overview

This chapter introduces ***First*PAGE Alarm Manager** protocol and describes how to utilize it from within your applications.

| | |
|---|---|
| ☞ | Refer to the *First*PAGE Alarm Manager Server Installation and User manual for more details on the *First*PAGE Alarm Manager Server product. |

# Getting Started

This manual contains procedural and conceptual information about the **First*PAGE* Alarm Manager Protocol.** It is written for System Administrators and Developers to enable them to utilize *First*PAGE Alarm Manager from within their applications.

While it is not required that the user have any prior programming or alarm management experience, the user should understand basic Windows concepts.

# Documentation Conventions

This document contains the following documentation conventions to help you navigate through the manual, obtaining a better understanding of the material.

- Notes contain tips or reminders about procedural and conceptual information within the manual.

- Cross-references provide you with a link to further information about the section of the document that you are currently reading.

# C h a p t e r  T w o
## *First*PAGE Alarm Manager Server

This chapter describes ***First*PAGE Alarm Manager Server** fundamentals that the application developer should know before attempting to develop a *First*PAGE Alarm Manager enabled application.

## Server Engine

The *First*PAGE Alarm Manager Server engine is a Windows service which provides an advanced interface for managing alarm notification. To interface with the engine applications indicate which host they are running on, which application they are and what alarm from their environment is now active, or inactive.

If this alarm is configured within the engine the notification process will begin. The escalation process may be initially delayed, may begin immediately or an occurrence count may be incremented for TIP level notification. As the start time of each escalation level arrives, or when a TIP level threshold is exceeded, the notification destinations at this level will be notified.

This document will describe these items in detail so that the developer may form the correct packets to achieve the desired results.

## Connecting to the Server

The *First*PAGE Alarm Manager Server engine listens, by default, on TCP/IP port number 8001. To connect to the engine the user must open a TCP/IP socket to the appropriate IP host, using this port. Once the connection is open the user may begin to execute the commands indicated below. You may opt to maintain this connection or it may be open and closed for each transaction if the application is not robust enough to provide reconnect on loss of this connection. The ADMIN;REGISTER; command should be

used after initiate connection. Command should be executed to terminate any currently processing escalation.

## Disconnecting from the Server

When the application is finished with the *First*PAGE Alarm Manager Server any outstanding transactions to EVENTS, ALARM etc. should first be terminated, before the application disconnects from the server. Unless the application is prepared to remember the assigned serial numbers for it's transactions across restarts, this is important as to not orphan messages within the engine as they will continue to escalate until all escalation levels have been exhausted.

On a shutdown of the client alarming engine the ADMIN;CLEAR; can be utilized to clear any outstanding notification without having to remember all transaction serial numbers.

## Object Terminology

Within this document the following acronyms (as detailed in the glossary) will be utilized.

| Term | Description |
|------|-------------|
| ALARM | |
| EVENT | |

## Packet Formats

All packets sent to and from the *First*PAGE Alarm Manager Server follow a standard format. A standard transaction follows the following convention:

Client:     operation;sub-op;[[object][;data]];<CR>

Server:     operation;STATUS;code;message;<CR>

| Field | Description | Notes |
|-------|-------------|-------|
| operation | The operation family being requested. | |
| sub-op | The sub-operation within the operation family desired. | |
| Object | The object the operation/sub-op is to be performed against. | |
| Data | Any optional data required for the operation/sub-op to be executed. | |
| STATUS | A static value always included in a 'STATUS' packet to indicate its type. | |
| code | The status code resulting from the execution of the operation/sub-option. | |
| message | The status message explaining the reason for the status code. | |
| <CR> | A carriage return, ASCII 13. | |

For extended status commands the format above is consistent but one or more result packets may follow. Result packets will only be returned, for commands need to return more data to the client than a status code, and if the 'code' value of the STATUS packet

is 0, which indicates the command was a success. The format of an extended transaction is as follows:

Client:        operation;sub-op;[[object][;action]];<CR>

Server:        operation;STATUS;code;message;<CR>

Server:        operation;RESULT;eofflag;data;<CR>

For the 'RESULT' packet, the 'eofflag' will be '0' if another result packet will follow, or '1' if this is the last result packet. The data portion of the packet is specific to the command being executed, and the trailing ';' is always provided.

| Field | Description | Notes |
|-------|-------------|-------|
| RESULT | A static value always included in a 'RESULT' packet to indicate its type. | |
| Eofflag | End of File flag. 1 if this is the last RESULT packet, 0 if more will follow. | |
| Data | Operation/Sub-Op specific data. | |
| <CR> | A carriage return, ASCII 13. | |

Note: The ';', since it is used as the protocol field separator, must be escaped if it is used within a field. To accomplish this prefix any ';' character with a '\' character, and escape any '\' character with another '\' character.

# C h a p t e r  T h r e e
## ADMIN Operations

This chapter describes the **First**PAGE **Alarm Manager** protocol subset known as the ADMIN operations. These commands, typically used by administration tools and gateways, allow the developer to request information about the **First**PAGE **Alarm Manager** Server status, force engine reload and manage client registrations.

## CLEAR

The CLEAR sub-operation requests that the engine clear all transactions tagged with the specified host;appl;instance;. If the ADMIN;REGISTER; command has been issued before this command then no parameters are required, it will clear the values supplied with the REGISTER sub-op. The use of this command will cause any currently processing notifications to be terminated without a termination message being sent.

Format:  ADMIN;CLEAR;[host;appl;[instance;]]

Status:   ADMIN;STATUS;code;message;

| INPUT | |
|---|---|
| **Item** | **Description** |
| host | |
| appl | |
| instance | |

## DETAIL

The DETAIL sub-operation requests that the engine return certain details to allow the administrator to determine which version is running.

Format: ADMIN;DETAIL;

Status: ADMIN;STATUS;statuscode;statusmessage;
Result: ADMIN;RESULT;1;FirstPAGE Alarm Manager
Server|hostname|major_ver|complete_ver|platform|start_datetime;

| OUTPUT | |
|---|---|
| **Field** | **Description** |
| Server Name | Always "FirstPAGE Alarm Manager Server" |
| hostname | Host name of the server running the server |
| major_ver | Major version # |
| complete_ver | Complete version specification |
| platform | Platform specification |
| start_datetime | Date/time the engine started specified in UCT |

## LIST

The LIST sub-operation requests that the engine return a list of the known unique host;appl;instance; combinations which have been registered. One of these unique values may then be used with the ADMIN;CLEAR; command to remove all transactions from that source.

Format: ADMIN;LIST;
Status: ADMIN;STATUS;code;message;
Result: ADMIN;RESULT;eof;host;appl;instance;

| OUTPUT | |
|---|---|
| **Field** | **Description** |
| host | Host name of registered server |
| appl | Application name of the registered server |
| instance | Optional instance name of the registered server |

## REGISTER

The REGISTER sub-operation requests that the engine tag any incoming transactions on this connection with the specified host/application information. This tagging can then be used in the future to remove all transactions. This command should be executed on the initial connection to *First*PAGE Alarm Manager.

    Format: ADMIN;REGISTER;host;applname;[instance;]
    Status: ADMIN;STATUS;code;message;

| INPUT | |
|--------|-------------|
| **Field** | **Description** |
| host | Host name of registered server |
| applname | Application name of the registered server |
| instance | Optional instance name of the registered server |

## RELOAD

The RELOAD sub-operation indicates to the *First*PAGE Alarm Manager Server that it should reload, most likely because its database contents have been manipulated and it needs to make changes to its internal data structures as appropriate.

    Format: ADMIN;RELOAD;
    Status: ADMIN;STATUS;statuscode;statusmessage;

## SECMODE

The SECMODE sub-operation requests that the engine return details as to which security mode it is running under.

    Format: ADMIN;SECMODE;
    Status: ADMIN;STATUS;statuscode;statusmessage;
    Result: ADMIN;RESULT;1;Secmode;

| OUTPUT | |
|--------|-------------|
| **Field** | **Description** |
| Secmode | Security mode bitmask.<br>Bit: 0 – Authentication required<br>    Off = No authentication required |

| | |
|---|---|
| | On = Authentication required<br>Bit: 1-4 - Unused<br>Bit: 5 – Ford Authentication<br>    (Requires bit 0 be on)<br>    Off = Ford authentication off<br>    On = Ford authentication on |

## SHUTDOWN

The SHUTDOWN sub-operation requests that the engine cleanly shutdown. The engine will proceed to clear all displays and release all connections to devices.

Format: ADMIN;SHUTDOWN;
Status:  ADMIN;STATUS;statuscode;statusmessage;

# C h a p t e r  F o u r
# **AGENT Operations**

This chapter describes the **FirstPAGE Alarm Manager** protocol subset known as the AGENT operations. These commands allow the developer to request alarms be initiated or terminated within the engine.

## ALARMS

The ALARMS sub-operation facilitates the listing of known alarms for a particular instance of an agent within the engine. This list contains a complete list of all configured alarms. Gateways should use this list to determine which alarms in the source application should be monitored and/or forwarded to *First*PAGE Alarm Manager for processing.

```
Format: AGENT;ALARMS;host;application;[instance;]
Status:  AGENT;STATUS;statuscode;statusmessage;
Result:  AGENT;RESULT;eof;alarmid;
```

| INPUT | |
|---|---|
| **Field** | **Description** |
| host | Host name of registered server |
| applname | Application name of the registered server |
| instance | Optional instance name of the registered server |

| OUTPUT | |
|---|---|
| **Field** | **Description** |
| alarmid | Host name of registered server |

## LIST

The LIST sub-operation facilitates the listing of known agent instances within the *First*PAGE Alarm Manager engine. This list contains a complete list of all configured agents. Consoles should

use this list to determine which host/agent combinations are available to be monitored.

    Format: AGENT;LIST;
    Status:  AGENT;STATUS;statuscode;statusmessage;
    Result:  AGENT;RESULT;eof;host|appl|descr|[instance|];

| OUTPUT | |
|---|---|
| **Field** | **Description** |
| host | Host name for agent instance |
| appl | Application name of agent instance |
| descr | Description of agent instance |
| instance | Name of the instance (optional) |

## QUERY

The QUERY sub-operation facilitates the listing of active agent alarms/events on an agent instance basis. This sub-op allows Consoles to get a list of the initiated alarms or events active within the *First*PAGE Alarm Manager engine. Consoles should use this list to act upon the active alarm list. See AGENT;LIST; for valid Operation Field values.

    Format: AGENT;QUERY;host;appl;[instance;]
    Status:  AGENT;STATUS;statuscode;statusmessage;
    Result:  AGENT;RESULT;eof;alrmnm|serno|starttime|
              descr|elap|esclvl|lastnot|msg|;

| INPUT | |
|---|---|
| **Field** | **Description** |
| host | Host name for the desired agent |
| appl | Agent name |
| instance | Name of instance (optional) |

| OUTPUT | |
|---|---|
| **Field** | **Description** |
| alrmnm | Name of active alarm |
| serno | Serial number of active alarm |
| startime | Time alarm became active: YYYY-MM-DD HH:MM:SS (Localtime) |
| descr | Alarm description |

| elap | Elapsed time in seconds alarm has been active.(Integer) |
|---|---|
| esclvl | Current escalation level (0-5) (see note below) |
| lastnot | First destination on current level. Distribution lists prefixed with @. |
| msg | Message text on current level. |

Note:  If there is an initial delay associated with the active alarm, and that duration has not yet elapsed, the "Esclvl" field will show "0", and the description and last notification will both be blank.

# C h a p t e r   F i v e
# ALARM Operations

This chapter describes the **First PAGE Alarm Manager** protocol subset known as the ALARM operations. These commands allow the developer to request alarms be initiated or terminated within the engine.

## INITIATE

The INITIATE sub-operation facilitates the raising of alarms within the engine. Repeated calls specifying a host;appl;alarm; of an active alarm will update any passed user text values for subsequent messages related to the specified alarm, it will not initiate a second copy of the alarm. If a currently populated user text item is not specified in the subsequent initiate then it will be set to a blank value. For alarms which are associated with timed based escalation schemes notification will being to immediately follow the schemes definition. For alarms associated with TIP level based escalation schemes the TIP count will be updated and notification will begin once the scheme rule has been met.

Format: ALARM;INITIATE;host;appl;alarm;[[ut1;]…[ut20;];]
Status:  ALARM;STATUS;statuscode;statusmessage;

| INPUT | |
|---|---|
| **Field** | **Description** |
| host | |
| appl | |
| alarm | |
| ut1 - ut20 | |

## TERMINATE

The TERMINATE sub-operation facilitates the termination of an initiated alarm to request the engine stop notifying destinations of alarms and to perform any termination options.

Format: ALARM;TERMINATE;host;appl;alarm;
Status:  ALARM;STATUS;statuscode;statusmessage;

| INPUT | |
|---|---|
| Field | Description |
| host | |
| appl | |
| alarm | |

# Chapter Six

## EVENT Operations

This chapter describes a subset of the ***First*PAGE Alarm Manager** protocol known as the EVENT operations. These commands enable the developer to obtain information about *First*PAGE Alarm Manager events defined within the server and remit transactions to engine for processing. This set of operations can be used to interrogate events as well as remit transactions to the server.

An EVENT may consist of one or more alarms if wildcard matching is used within the alarm definition.

### DISABLE

The DISABLE sub-operation facilitates the disabling of an event within the engine. It requires the specification of the host/appl/alarm combination.

Format: EVENT;DISABLE;host;appl;alarm;
Status:  EVENT;STATUS;statuscode;statusmessage;

| INPUT | |
|---|---|
| **Field** | **Description** |
| host | |
| appl | |
| alarm | |

### ENABLE

The ENABLE sub-operation facilitates the enabling of a disabled event within the engine. It requires the specification of the host/appl/alarm combination.

Format: EVENT;ENABLE;host;appl;alarm;
Status:  EVENT;STATUS;statuscode;statusmessage;

| INPUT | |
|---|---|
| **Field** | **Description** |
| host | |
| appl | |
| alarm | |

## INITIATE

The INITIATE sub-operation facilitates the raising of an instance of an event within the engine. It requires the specification of the host/appl/alarm combination, and optionally up to 20 user text parameters. For events which are associated with timed based escalation schemes notification will being to immediately follow the schemes definition. For events associated with TIP level based escalation schemes the TIP count will be updated and notification will begin once the scheme rule has been met.

    Format: EVENT;INITIATE;host;appl;alarm;[[ut1;]…[ut20;];]
    Status:  EVENT;STATUS;statuscode;statusmessage;
    Result:  EVENT;RESULT;1;serial#;

| INPUT | |
|---|---|
| **Field** | **Description** |
| host | |
| appl | |
| alarm | |
| ut1 - ut20 | |

| OUTPUT | |
|---|---|
| **Field** | **Description** |
| serial# | |

## TERMINATE

The TERMINATE sub-operation facilitates the lowering of a specific instance of an event within the engine. This will stop the

process of notifying destinations of alarms which exist within the event, and perform any terminate options on those alarms.

Format: EVENT;TERMINATE;serial#;
Status: EVENT;STATUS;statuscode;statusmessage;

| INPUT | |
|-------|-------------|
| Field | Description |
| serial# | |

## ACK

The ACK sub-operation facilitates the acknowledgement of an existing event within the engine. This will terminate messaging for the destinations at the specified level, sending them the [optionalreasonmessage] indicating that their level of responsibility has be acknowledged. Normal messaging for subsequent levels will occur after normal timing rules expire.  This sub-operation is basically a no-op on events which have been configured to ignore ACK's.

Format: EVENT;ACK;serial#;level;[optionalreasonmessage;]
Status: EVENT;STATUS;statuscode;statusmessage;

| INPUT | |
|-------|-------------|
| Field | Description |
| serial# | |
| level | |
| optionalreasonmessage | |

## ESCALATE

The ESCALATE sub-operation facilitates the immediate escalation of an existing event within the engine. This will force the engine to begin messaging the specified escalation level immediately. Normal messaging for subsequent levels, after the level escalated to, will occur after normal timing rules expire.

Format: EVENT;ESCALATE;serial#;level;[optionalreasonmessage;]
Status: EVENT;STATUS;statuscode;statusmessage;

| INPUT | |
|---|---|
| **Field** | **Description** |
| serial# | |
| level | |
| optionalreasonmessage | |

## QUERY

The QUERY sub-operation facilitates the querying of event information. It will return a list of events which match the query. If the optional [event] field is not specified then all alarms which are configured for the specified host;appl will be returned. If the optional [event] field is specified then only alarms which match that specification will be returned. The wildcard characters * (match more than one character) and ? (match a single character) will be processed.

    Format: EVENT;QUERY;host;appl;[event;]
    Status:  EVENT;STATUS;statuscode;statusmessage;
    Result:  EVENT;RESULT;0;firstevent;
    Result:  EVENT;RESULT;1;lastevent;

| INPUT | |
|---|---|
| **Field** | **Description** |
| host | |
| appl | |
| event | |

| OUTPUT | |
|---|---|
| **Field** | **Description** |
| firstevent | |
| lastevent | |

## UPDATE

The UPDATE sub-operation facilitates the updating of user text items on an event instance. These new user text item values will be used on subsequent messages. If a currently populated user text item is not specified in the update statement then it will be set to a blank value.

Format: EVENT;UPDATE;serial#;ut1;ut2;ut3;…;ut20;
Status: EVENT;STATUS;statuscode;statusmessage;

| INPUT | |
|---|---|
| **Field** | **Description** |
| serial# | |
| ut1 – ut20 | |

# Chapter Seven
## USER Operations

This chapter describes the *First***PAGE** protocol subset known as the USER operations. These commands allow the developer to define user options and obtain status.

### LOGIN

The LOGIN sub-operation facilitates the authentication or identification of the user to the *First*PAGE Server engine. If the engine is running in secure mode then the username and password must exist in the user table before the user may interact with *First*PAGE Server. If the engine is running in in-secure mode then the user account is used only to track preferences.

Format: USER;LOGIN;domain;user;password;
Status:  USER;STATUS;statuscode;statusmessage;

| INPUT | |
|---|---|
| **Field** | **Description** |
| domain | Domain of user account. Note: If security mode is Ford Authentication then domain must be "FORD". |
| user | User account |
| password | Password |

# C h a p t e r   E i g h t
## Formatting/Operation Tokens

This chapter describes a subset of the *First***PAGE Alarm Manager** protocol formatting and operation tokens. These tokens, when inserted within the message data, are replaced with the values they represent.

The tokens listed below will only be supported if within the destinations capabilities.

## Server Tokens

The server tokens allow you to specify various items such as insertion of date and time values into messages etc.

| Token | Description | Notes |
| --- | --- | --- |
| %ALARM% | Specifies that the engine place the alarm id in this location. | |
| %APPLICATION% | Specifies that the engine place the agent application name in this location. | |
| %DATE% | Specifies that the engine place the current date, in server timezone format, in this | |

| | | |
|---|---|---|
| | location. | |
| %DATETIME% | Specifies that the engine place the current date/time, in server timezone format, in this location. | |
| %DESCRIPTION% | Specifies that the engine place the alarms description, in this location. | |
| %DESTNAME% | Specifies that the engine place the current destination name in this location. | |
| %ESCLEVEL% | Specifies that the engine place the current escalation level in this location. | |
| %HOST% | Specifies that the engine place the current agent's host specification at this location. | |
| %TIME% | Specifies that the engine place the current time, in server timezone format, in this location. | |
| %TIPCOUNT% | Specifies that the engine place the TIP occurance | Valid only for alarms triggered via TIP Level |

| | counter in this location. | based escalation. |
| --- | --- | --- |

# Chapter Eight
## Example Session

This chapter provides a sample session of an application interacting with the *First*PAGE Alarm Manager Server.

# Technical Assistance

## FAQ

Every SeQent product contains a FAQ file specific to that product. You can view this FAQ file by visiting the SeQent web site (www. SeQent.com) and browse the product page for the product you are using.

## Support Contracts

You may purchase a support contract for *First*PAGE Alarm Manager based products.  Various levels of support are available. View the SeQent web site (**www.SeQent.com**) and browse the product page for product you are using.

## Help Desk

To reach our help desk please use one of the following methods:

> **Telephone:** +1.519.652.0401
> **Fax:** +1.519.652.9275
> **Web: www. SeQent.com** - Customer Care
> **E-Mail: support@ SeQent.com**

Please have your support contract, or product license key ready, before calling or include it in your correspondence.

# G l o s s a r y

## Terms

### Agent

An agent within *First*PAGE Alarm Manager is????????

### Agent Instance

An agent instance within *First*PAGE Alarm Manager is a specific agent assigned to either a certain host, a certain application instance on a host or both of the above.

### Alarm

Within the *First*PAGE Alarm Manager engine a specification for an alarm consists of: host name, application name, alarm name. Only one instance of an alarm may be active within the engine at one time.

### Escalation Scheme

There are two types of escalation schemes within *First*PAGE Alarm Manager. The first type of escalation scheme is timed based. You may define the number of messages and the delay between the messages for each of the escalation scheme levels when an alarm or event is initiated. The second type of escalation scheme is TIP level escalation where you can define the number of alarms or events which must occur within a time frame before the notification process begins.

### Event

The *First*PAGE Alarm Manager engine specification for an event consists of: host name, application name, alarm name. This is the same specification as an alarm but using the event interface multiple instances of the event may be active within the engine at

one time. To accomplish this each instance is assigned a serial number.

## TIP Level

TIP level is a rule which defines the number of alarms or events which must occur within a time frame before the alarm or event is acknowledged as an issue. TIP levels are implemented as an escalation scheme type within *First*PAGE Alarm Manager.