Paging Interface 1.5.2

Interface Series

Installation and User's Guide

Trademarks

First Choice Software, Inc. (FCS) develops and supplies add-on software and customizations to the following Clarify products:

- ClearSupport
- ClearHelpDesk
- Policies and Customers
- Product Manager
- ClearLogistics
- ClearQuality
- ClearContracts
- ClearSales
- ClearCallCenter

FCS uses the following Clarify tools:

- User Interface Editor
- Data Dictionary Editor
- Data Exchange
- ClearBasic Exchange
- ClearBasic Batch

Clarify, Inc. and all of its product and tool names are trademarks of Amdocs.

Publication Notice

The information in this document does not constitute a warranty of performance. First Choice Software, Inc. reserves the right to modify the content of this document without obligation to notify any person or party of such revisions or changes. First Choice Software, Inc. assumes no liability for losses incurred as a result of the use of information contained in this document.

Table of Contents

Overview

This document describes an interface to add enhanced paging connectivity to the Clarify family of applications. This Interface is known as First Choice Software, Inc.'s Paging Interface.

Many Clarify customers already use RPA (Radio Page America) as a paging service. RPA allows customers to use a single point of contact (and single protocol) to perform sophisticated paging operations. RPA uses store and forward technology to handle all of the complicated problems associated with employees who use many different pager services and levels of support. Other Clarify customers use PageMate as their paging service. Version 1.3 and later versions of the FCS Paging Interface includes PageMate support. PageMate does not provide two-way messaging, so some features provided may not apply for PageMate.

This interface provides basic paging capabilities with Clarify and RPA or PageMate:

- Notifications/escalations (from business rules) that should be sent via pager will be sent automatically via the Paging Interface through to the paging service.

In addition, this interface provides several new capabilities that enhance the Clarify paging system:

- The ability to send *ad-hoc* pages. These are pages that are sent at any time for any purpose, and are not tied at all to the business rule system.
- A new GUI form that allows a user to send a page related to a specific case, subcase, or employee.
- A new page monitor form that allows a user to view pages that have been sent, and responses that have been received (for two-way paging).
- A new API that can be customized in Clarify forms to send hard-coded pages at any time.
- APIs that allow the page monitor or page request forms to be displayed from any customization.
- Business rule support that allow for business rules to be fired on the page request or page response events.

The interface is extremely configurable. Each page request can optionally be tracked in the database, and any responses that RPA sends back can also be tracked. Activity log entries can be made when pages are made in relation to cases, subcases, and employees. Business rules can be added that will perform actions based on requests or actual/expected responses. Responses can be filtered so that only certain types are stored in the database.

This product is a Clarify-based application that is internationalized and localized. The application ships with a set of default strings in US English. This document describes how to easily modify the database to allow users to operate the product in different locales (simultaneously).

Before You Upgrade to a New Version…

You should be aware of both the version of the product you currently have as well as the version you are upgrading to.

You may upgrade several revision levels at one time. To perform this multiple-upgrade, you must create a list of files to upgrade. Find the section of the manual with instructions for upgrading from your current software revision. Make a list of the required steps. Repeat this process with each update, adding these steps to the list. If a step is required by more than one version, you only have to list it once.

When your list is completed, you should perform all the steps (in the same order you would install the product (schema first, then files and forms, then resource configurations, then code compilation) for a clean install) using the files provided with the most recent version of the product.

For example, suppose you had version 1.1 of a product, and versions 1.1.1, 1.2, and 1.3 have all been released. You now wish to upgrade from version 1.1 directly to version 1.3. Suppose that the versions require the following steps:

Version 1.1.1 requires that you compile files a.cbs and b.cbs.

Version 1.2 requires that you add a new field to the case table, and recompile a.cbs.

Version 1.3 requires that you import a data file d.dat, import a new form file, c.dat (add it to the resource configuration), and compile c.cbs.

To perform the total upgrade in this situation you would perform the following steps:

1. Add the new field to the case table (schema changes always come first)
2. Import the d.dat file file (data imports come next)
3. Import the c.dat form, and add it to the proper resource configuration(s).
4. Compile the a.cbs, b.cbs, and c.cbs files.

All of these steps would be accomplished with the files provided with the 1.3 version of the product. If you have questions about this process, please contact First Choice Software.

What's New in Version 1.1

Version 1.1 adds more business rule functionality for two-way paging. Response files that are received from RPA can be processed and the information returned can be attached to the page request object. The Page Monitor form has been enhanced to include the confirmation field from the Page Response table. A new business rule has been included in the *pg_br50.dat* file for taking action on new Page Response records. The new rule is designed to pass the page_response objid to a cbbatch program which will then create an act_entry record for the original page_request object. A sample cbbatch program titled log_cf.bat is included in the updated package.

What's New in Version 1.2

Version 1.2 adds PageMate support functionality.

What's New in Version 1.3

Version 1.3 enhances PageMate support functionality for UNIX. The *pg_daemon.cbs* program has had some modifications to it, along with the *pg.ini* file. Unix shell processing from cbbatch has been improved.

What's New in Version 1.3.1

Version 1.3.1 fixes a small issue with the way rulemanager invokes pager_clerk. When a notification for a business rule is sent, and the notification method is pager, then the rulemanager invokes a SHELL to call pager_clerk, as shown in the rulemanager log file:

Executing SHELL 'E:\Clarify\eFrontOffice9\ClarifyServer\Rulemgr\pager_clerk.exe -p SKY_WORD -t "1234567890" -m "Testing pager clerk" -e "some_address@company.com" &'

One would expect that the ampersand is handled only by the SHELL. However, the ampersand seems to get passed to pager_clerk as a parameter, which was causing pager_clerk to fail.
This was only observed on Windows NT/2000 based systems.

The pager_clerk program has been modified such that it ignores an ampersand passed in as the last parameter. The source code for pager_clerk (pager_clerk.c) has been modified, and, for Windows based systems, pager_clerk.exe has been compiled using the new source code.

What's New in Version 1.3.2

Version 1.3.2 fixes a small issue with the pager manager daemon program. Multi-line page requests were not being handled correctly, resulting in only the first line being sent out to the pager. This has been corrected.
To apply the patch, copy the new version of pg_daemon.cbs to your system and restart the Paging Manager Daemon.

What's New in Version 1.4

Version 1.4 adds support for RPA's pass-through technology.

What's New in Version 1.5

Version 1.5 addresses the issue of the paging daemon causing high CPU utilization when running on UNIX systems. The cause for this is the ClearBasic *sleep* command. The paging daemon 'sleeps' between cycles, and on UNIX systems, this can cause high CPU utilization. To workaround this issue, the paging daemon will not use the ClearBasic sleep command, but will instead call on database commands to perform the sleep.
To apply the patch, copy the new version of pg_daemon.cbs to your system. In addition, there is a new parameter in the pg.ini file – DB_TYPE. You should either copy the new pg.ini file to your system and edit accordingly, or else simply add the DB_TYPE parameter to your existing pg.ini file. Finally, restart the Paging Manager Daemon.

Note for Oracle database users: The ClearBasic *sleep* command has been replaced by a call to the Oracle procedure *dbms_lock.sleep*. The **dbms_lock** package is created as part of the standard installation when **catproc.sql** is run. If you system does not have it installed, it can be created by running the script **$ORACLE_HOME/rdbms/admin/dbmslock.sql**, after connecting to the database under the **SYS** account. There is a public synonym in place for the package, but under 8.1 execute rights are granted to the role EXECUTE_CATALOG_ROLE, so you may not be able to call the function under a low-privilege account. Please refer to your Oracle documentation for more details.

What's New in Version 1.5.1

Version 1.5.1 fixes the issue of the Rule Manager process not handling the end date that is set by the page_request function that is in the pg_global.cbs file. The date used originally was "1/1/1800", but as of February 27, 2004, this date was no longer processed correctly. This version changes the end date for the time bomb records to "1/1/2000", and that resolves the issue.

What's New in Version 1.5.2

Version 1.5.2 includes a new pager_clerk.exe application for Widows, and a new pager_clerk.c for UNIX.
Om previous versions, if multiple pager_clerk.exe processes started and finished within the same second, the output file could be overwritten, causing outgoing pages to be "lost" and not sent.

Previously, the file name (of the paging request file generated by pager_clerk) was made "unique" by using a timestamp (down to the level of a second) as part of the filename. This was fine, until multiple pager_clerk.exe processes ran at the same time.

Now, for Windows systems, a GUID is used in the filename, which will guarantee uniqueness. For UNIX systems, the timestamp is combined with the PID (Process ID).

To upgrade to this version:
On Windows, replace the old version of pager_clerk.exe file with the new one.
On UNIX, compile the pager_clerk application, and replace the old version with the new.

Terminology

This section includes definitions of some basic terminology that may be needed when reading this document:

Term	Definition
Clear Support	Clarify call-tracking product that will be used with this interface by customers.
RPA (Radio Page America)	This is a service that allows customers to send (and receive) notifications via pager. The RPA service is a store/forward service. That is, customers can use one phone number and protocol to speak with RPA, even though their employees may have pagers from many different vendors/services (with many different protocols).
PageMate	This is a service that allows customers to send notifications via pager.
Serial Interface	RPA-supplied program that reads ASCII text files, and generates requests to the RPA system (via modem or internet).
ad-hoc page	A page is a paging request made as a direct request from a Clarify user.
notification page	Pager notification is a paging request made as the result of a notification from a Clarify business rule.
page_request	Paging Interface ability to request that a page be sent.
page_response	For two-way paging, the ability to send a response back from the page request. The page_response may or may not be stored in the Clarify system (customer setting).
Activity Log (act_entry)	A standard Clarify term for an audit trail. The Paging Interface adds the concept of an activity log for an individual page request.
Business Rule (time_bomb)	Ability in Clarify to specify business conditions that will trigger notifications, escalations, or actions (such as generating a page request).
Configuration Items (config_itm)	Settings in the Clarify system that control the behavior of the Paging Interface.
Page Monitor	A new GUI form in the Clarify systems that enables users to filter, sort, and view page requests (and, optionally, responses to those requests).
Page Request Form	A new GUI form in the Clarify system that users can use to enter and send ad-hoc page requests.
Clear Basic	A Clarify-supplied scripting language that will be used for most of the customizations for this interface.

Prerequisites

The Paging Interface will require the following components be present for a customer to properly implement the product:

- A valid Clarify system (5.0 or later)
- Valid accounts registered on the RPA system, and a properly configured RPA Serial Interface - OR - PageMate software and license for the server(s) used to send page requests to PageMate.
- Clarify's User Interface Editor
- Clarify's Data Exchange Utility (bulk load program)
- Clarify's Data Dictionary Editor (or the DDComp program on the Clarify server)
- Clarify's Clear Basic Toolkit (compiler)

Using the Interface

This section describes what the Paging Interface can do, how it does it, and how to use it.

Paging Interface Functionality

- Clarify provides a variety of notification mechanisms for delivery of messages from the firing of a business rule. Among these methods are numeric, digital, and text paging. The Paging Interface allows paging type notifications to be dispatched using RPA or PageMate services. The Paging Interface can optionally track page requests made as the result of business rule notifications.
- In many situations, Clarify users require the ability to initiate *ad-hoc* pages to employees. The Paging Interface provides a client-side API that allows pages to be generated from nearly any Clarify form. The Paging Interface can track page requests made as the result of ad-hoc paging. Business rule events (time-bombs) can be generated from an ad-hoc page request which can drive business rules related to ad-hoc paging. Activity log entries can be optionally created against the focus object (case or subcase) as the result of an ad-hoc page request.
- RPA provides a response mechanism that processes responses from two-way pager users. The Paging Interface can track responses from both ad-hoc paging as well as business rule pages. The Paging Interface can also create activity log and business rule event (time_bomb) entries against the focus object (case or subcase) as the result of a page response.
- In concert with the tracking of page requests and responses, the Paging Interface provides a GUI that allows users to monitor page requests and responses.
- The Paging Interface provides a page request GUI that allows pages to be sent at any time. Optimally, ad-hoc pages created via the GUI can be related to a case or subcase. The paging text can be entered, and the employee to be paged can be selected from the Clarify database.
- The Paging Interface provides an API for the generation of page requests, as well as the launching of the page monitor and page request GUIs. These can be added to virtually any form in Clarify.

Using the Paging Interface

Standard (Notification) Paging

Using the Paging Interface for normal (business rule-based) paging is no different than base Clarify. As long as the user's preferences are set to paging (text, digital, or tone), and the user's profile in Clarify has the proper pager number set up, the business rule engine will use the Paging Interface to page the user via RPA.

Relating Notification Pages to Cases, Subcases, and Employees

If you have a normal business rule that will notify a user via a pager, it is often the case that you wish to relate that page to a case, subcase, or employee. The reason for wanting to do that is so that the page can be viewed (and filtered) in the Page Monitor (see below).

The Paging Interface provides a mechanism for doing this. If the message text begins with one of the following keywords, followed by the object ID (the variable [Database ID]), then the relation will be made for you by the Paging Interface. The three keywords are:

- CASE
- SUBCASE
- EMPLOYEE

For example, if you wish to have the page (for a case dispatch action) related to the current case, you could have the following message:

```
CASE [Database ID] [Object Type] [Object ID] was dispatched to
Queue [Current Queue].
```

This will insure that each copy of the page will be related to the case.

Multiple tags can be used. For example, you could place both the CASE and EMPLOYEE tags at the start of a message to relate the page to the employee and the case.

The tags are removed from the text message, so the tag word (and object ID) will not be seen by the user. In the example above, a user might receive the text page of:

```
Case 42 was dispatched to Queue High.
```

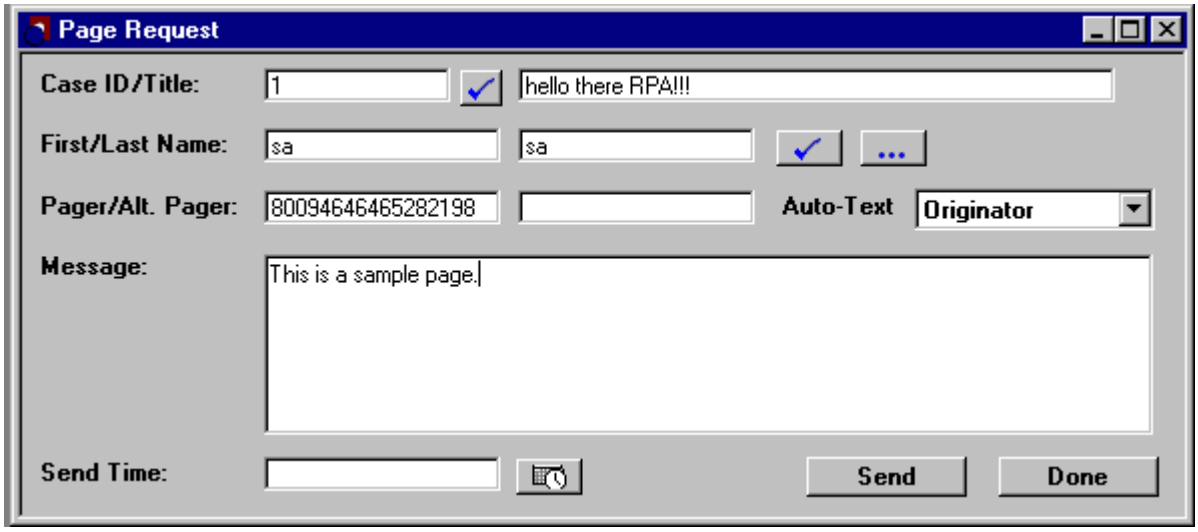
There is currently no business rule variable for the employee object ID. You must take any properties you desire (such as [Current Case Owner]), and create new ones that extend the path with the "user2employee:objid" fragment). This new row must be added to table_prop_name. For more information on how to created business rule variables, see the Clarify System Administrator's Guide.

Also, a sad side effect is that if a notification goes to multiple employees, and one is paged, and one receives either Email or Notifier notifications, the tag word and object ID cannot be stripped from the non-pager notifications. This is due to the way in which Clarify works, and is not currently customizable.

The use of these tags can be disabled via a configuration item. See the configuration item section for more information.

Ad-hoc Paging

Using the new ad-hoc paging capability is also very simple. A new form (callable via a menu item or via customization) is provided.



This form allows a user to send a page. The page can be related to a case or subcase, but does not have to. Enter the case/subcase ID number, and press the checkbox to validate it.

If the page is to be related to an employee, the first or last name (or as much as is known) can be entered, and the checkbox pressed. Or simply press the checkbox to bring up the *Select Employee* form. Once an employee is selected, their pager and alternate pager numbers will be filled in.

A message can be entered in the message box. If a case or subcase has been selected, the *Auto-text* list provides a series of variables (items of interest from the case, such as the contact name, or site_id). Selecting one of these items will place the proper text in the message text box.

Normally, the page is sent right away, but placing a time in the future in the send time box will cause the page to be sent at that time.

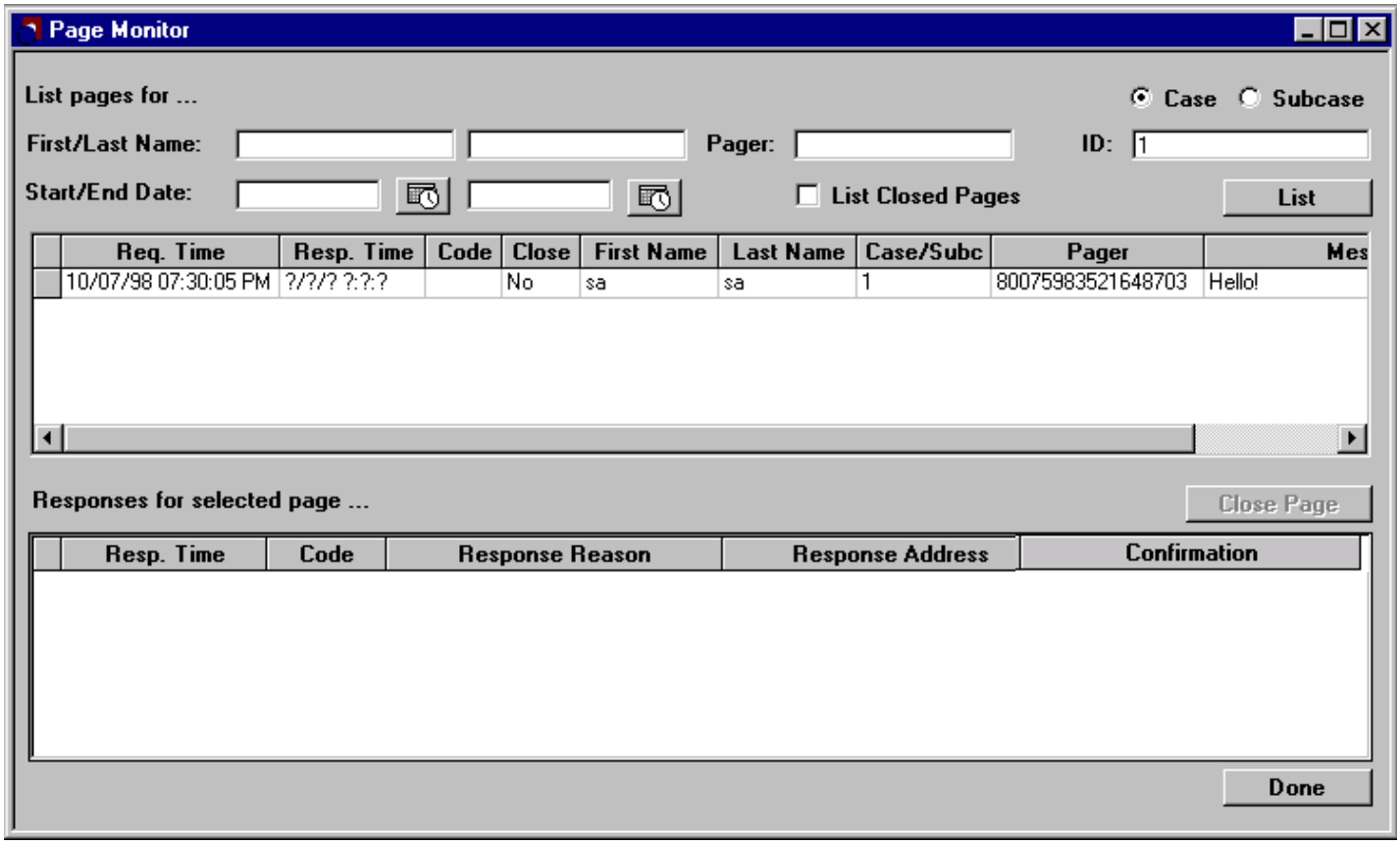
Finally, pressing the *Send* button starts the paging process. The page is sent to the number in the pager field. If the pager field is blank, the alternate pager number is used.

It is not required that an employee be related to the page. Simply entering a pager number and message (and pressing Send) is enough to launch a page.

The page request form can be displayed via a menu item, or any customization. An API (see the reference section at the end of this document) can be used to display the form at any point in the Clarify system.

Page Monitoring

The Paging Interface provides the capability of monitoring pages that have been sent out (and the responses sent back for two-way paging). This selection form is very useful for following the pages for a particular case, subcase, or employee. Again, the PageMate service does not provide two-way messaging.



This form allows (via the filter at the top of the form) to view all pages, or to select pages for a particular case or subcase, for a particular employee (or pager number), or for a date range. The user may view open pages, or open and closed pages.

Pressing the *List* button views all of the pages for the selection criteria. Selecting (by single clicking the row) a particular page will display all of the responses to that page in the bottom list. The information about the response include the response time, code, reason, address, and confirmation (if applicable).

If a page is selected, the *Close* button (on the right side of the form, between the two grids) can be used to Close the page (or then to re-open the page). Note that the "closing" of a page is only used for your own use in filtering the pages. Some customers choose not to use the function, whereas some customers always "close" the page after it has been received. It can be a useful function to help limit the number of pages displayed in the page monitor.

The page monitor can be displayed from a menu item, or from a customized button. In calling the page monitor form from a new button it is easy to pre-filter the displayed pages. For example (see below), a new button could be placed on the case form. Pressing the button will cause the page monitor form to be displayed with the pages for that case.

Page Request API

The Paging Interface ships with an API that can be used to send a page via any customization (or business rule). The API does not need the Page Request form to be posted.

For example, suppose that a new button was to be placed on the case form. Every time it is pressed, the owner of the case should receive a page saying "Call back to the help desk". Such a page could easily be customized (in a very few lines of code) using the page request API. See the reference manual at the end of this document for more information.

Activity Logs

All fundamental Paging Interface activities can generate activity log records to provide an audit trail of the activity. The activity logs will be related to the page_request object, as well as to the focus object or employee related to the page. This allows for the activity entries to be reported for the page_request object.

The following new activities will cause activity log entries to be made in table_act_entry:

- Creating a page request through the page_request API
- Processing a page response for a page request made through the page_request API
- Processing a page response for a page request made through a business rule notification

Using configuration items, the activity log generation can be suppressed if you do not want such activity logs in your system.

Business Rules

The page request operation (and the response operation for advanced users) can be used to trigger business rule events.

The business rule form (#472) has been augmented with new business rule objects (Page Request and Page Response). There are also two new actions, the *Page Request* and *Page Response* actions. If a page is sent (via ad-hoc paging), a time-bomb (business rule notification record) is added to the proper table. If you have set up a business rule to fire on this event, it will fire (and take whatever action is desired).

Note that if you do not want this functionality, you do not have to import the new form #472, and you can disable the generation of page request time bombs via the configuration items.

Note that this function only works for Clarify 5.0 and later.

For example, suppose you want to set up a business rule that each time a page is sent that the user "Joe" is notified, the following rule could be entered:

Rule Name: Test Rule
Rule Set: Paging Rules
Rule Status: Active
Object Type: Page Request
Comment: This is a test rule
Start Event: Page Request
Cancel Event: <None>
Conditions: <None>
Action Title: Notify Joe
Notify: Joe
Urgency: <Any>

Start Action: 0 minutes after event creation using elapsed time
Repeat: 0 times
Activity Log: No
Type: Message
Message: Hi Joe, a page was sent!

Employee Preferences

To set the employee's preferences to paging (for business rule notification), use the *My Preferences* form from the *Desktop* menu.

Setting the Employee's Pager Number

Each employee should have their pager number (as understood by RPA or PageMate) set in their employee profile. An administrator should modify (using Policies & Customers) the employee record and set the pager number in the pager field.

If alternate pager numbers are to be used, that field can be customized on the employee form. In base Clarify, the field is defined in the employee table, but not on the form. See Clarify documentation for more information on how to add a text box to a form.

Paging Interface Processes

This section is informational only and is intended for advanced administrators.

Pager Notification

Pager notification is a paging request made as the result of a notification from a Clarify business rule. Clarify implements a pager notification via its Rule Manager server process. When the rule manager is required to make a pager notification, it calls the *pager_clerk* executable located in its working directory.

The pager_clerk executable is called with the following parameters:

Parameter	Description
-p	Pager type • SKY_PAGER – numeric pager (Clarify pager type specified as TONE) • DIGITAL – digital pager (Clarify pager type specified as DIGITAL) • SKY_WORD – text pager (Clarify pager type specified as TEXT)
-t	Pager number
-m	Message. Message is an empty string if pager type is SKY_PAGER or DIGITAL. Message is a text string if pager type is SKY_WORD.
-e	Email address of the paged employee

The location and parameters used by the Clarify Rule Manager in calling the pager_clerk executable are hard-coded into the Rule Manager. The Paging Interface provides a new *pager_clerk* executable program to implement pager notification via RPA or PageMate. The FCS pager_clerk executable replaces the default Clarify pager_clerk, and generates a Page Request file on the Clarify rule manager server. The Paging Manager processes the page request and submits it via RPA's Serial Interface or PageMate.

After the Paging Interface is set up, the end-user will not need to know about (or see) the pager_clerk executable program.

Response Processing (RPA Only)

The RPA Serial Interface provides a set of response codes for page requests. The Paging Interface processes responses to page requests, and optionally tracks these responses in the Clarify database in the page_response table. For responses to ad-hoc requests, the Paging Interface also optionally creates business rule events (time-bombs). The

customer can define business rules to fire on responses to ad-hoc paging requests. The RPA Serial Interface responses can be categorized as reject/time-out responses, non-delivery/non-confirm, status and two-way responses.

Reject/Time-out

These response codes indicate that the Serial Interface was unable to transmit the page request to RPA:

Response Code	Description
RJ or PN	Page Reject - communications network rejected the page request
TN	CTIM Reject – communications network rejected the CTIM page request
TO	Time Out – RPA Serial Interface was unable to deliver the page to the communications network

Non-delivery/Non-confirm

These response codes indicate that RPA was unable to deliver the page to the device:

Response Code	Description
ND	Non-delivery – RPA Serial Interface was unable to deliver the request to the communications net
NC	Non-confirm – communications network did not receive a confirmation within a period of time

Status

These response codes indicate that RPA delivered the page to the device:

Response Code	Description
AC or PA	Page Accept – RPA accepted page request for delivery to communications network
TK	CTIM Accept – RPA accepted CTIM request for delivery to communications network
DL	Delivery – RPA delivered page request to communications network

Two-way

The Paging Interface optionally processes responses to ad-hoc page requests. For status tracking reasons, the customer may request that the Paging Interface track page requests. If tracking requests, the customer can also specify that the Paging Interface track page responses.

The following table lists the valid two-way response code:

Response Code	Description
CF	Confirmation – RPA received a confirmation from the field via two-way pager

The Paging Interface can track page responses in two ways. The first tracks only the last response received from the RPA Serial Interface. The second tracks all responses received from the RPA Serial Interface. In both cases, the Paging Interface can be directed to create business rule events (time-bombs) upon receipt of ad-hoc page responses. The customer can define business rules to fire on response to ad-hoc page requests.

Server Processes

This section is informational only and is intended for administrators.

Pager Clerk

When the Clarify Rule Manager determines that a pager notification is required, it invokes the 'pager_clerk' program in the Rule Manager working directory. This behavior is hard-coded within the Rule Manager. There is no configuration option that would allow an alternate program to be invoked for pager notifications. As a result, to process business rule pager notifications by way of RPA or PageMate, the original pager_clerk program must be replaced with one providing the functionality described here. The FCS Pager Clerk is the replacement for the Clarify provided pager_clerk program. This program is also used to process page requests made through the page_request

API. The API creates a time_bomb record which is processed by the Clarify Rule Manager, which in turn calls the pager_clerk executable.

The pager_clerk program is provided in executable form for Windows NT, and the source code is provided for compilation under Unix. The executable needs to be placed in the Clarify rulemgr directory on the server. This program does not need to be started by user action since it gets executed by the Clarify Rule Manager. Its output goes into the directory specified by the FCS_PAGING_REQUEST_DIR environment variable, or in the rulemgr directory by default.

Paging Manager

This process is responsible for processing request files from the pager_clerk application, and creating serial interface page request files for submittal to RPA or PageMate. The Paging Manager application is also responsible for processing response files from the RPA Serial Interface. The Paging Manager has two primary jobs. The first is to process page requests from the Page Request table in Clarify, and create serial interface page request files. The second is to process responses from the RPA Serial Interface and update the Clarify database with the response information (N/A for PageMate systems).

Starting the Paging Manager

The Paging Manager is written in CB and runs as a daemon process. It is started once, and runs until stopped. The program can be customized via the *pg.ini* file. That file contains the following values that can be changed for your system.

Parameter	Definition
SLEEP=2	Indicate the number of seconds to sleep between empty cycles.
CYCLE=5000	Indicate the number of process cycles to complete before respawning.
DB_KEY=ST	Indicate the database key to use as a unique identifier.
LOG_LVL=2	Indicate the message level (0=none, 1=limited, 2=full).
SHELL_FILE=cbbatch –f pg_daemon.cbs –r pg_daemon	This is the program name to respawn when the Paging Manager has processed the number of cycles in the item above. It should be a path that will restart the Paging Manager program.
FCS_DIR=c:\fchoice\fcs_dir\	Indicate the directory where the fcspr files are created by pager_clerk.
PAGING_DIR=c:\fchoice\pg_dir\	Indicates the directory where the serial interface files are stored.
SEND_RPA=c:\fchoice\DATAIN\	Indicates the directory where the pass-through page request files are stored.
FROM_RPA=c:\ fchoice \DATAOUT\	Indicates the directory where the pass-through page response files are stored.
LOG_DIR=c:\fchoice\pg_log\	Indicates the directory where the log files are stored.

PM_PATH=C:\PageMate\page.exe	Indicates the directory containing the PageMate executables.
DB_TYPE	The type of database you are using. Valid values: ORACLE, SYBASE, MSSQL

A batch or script file is used to start the process running on the server. An example is shown below:

```
c:\apps\clarify\server\60\rulemgr\cbbatch -f
c:\apps\clarify\server\60\rulemgr\rpa_daemon.cbs -r rpa_daemon
```

Make sure that a valid *clarify.env* file exists in the rule manager directory (along with the line *auto_login=TRUE* so that the Paging Manager will connect to the proper database).

Restarting (Spawning) the Paging Manager

You might wonder why the Paging Manager program has to spawn (restart) itself every so often. This is due to a small memory leak in the Clarify ClearBasic program cbbatch. It is possible, if the Paging Manager program runs long enough that it will slow down due to the memory leak.

To make sure that this doesn't happen, the Paging Manager has been built so that (every so often) it will stop itself, and launch another copy (which frees the lost memory) of itself.

How often that should happen is up to you. With Clarify 6.0, the memory leak is much smaller, but still remains. An administrator should test and see what a reasonable value is for the CYCLE parameter, but it should be lower (perhaps 500) on a Clarify 5.x system, whereas it can be much higher on a Clarify 6.0 system.

Managing Files

When the Paging Manager has finished with a page request file, it is placed in a directory called *Done* which is created under the directory where pager_clerk places the page request files. The files are renamed, with a comma at the start of the name, and moved into the directory.

It is up to the administrator to decide how often to clean out this directory, and what procedure to follow to perform the cleanup.

The RPA serial interface uses a similar mechanism for the files that it creates. Consult the RPA documentation for more information.

Managing Log Files

The Paging Manager will create a new log file each day in the directory specified by LOG_DIR, and has the ability to produce different levels of log activity, ranging from no logging to debug level logging. The level of logging activity is specified in *pg.ini* by LOG_LVL.

Processing Page Request Files

The Clarify application drives the generation of Paging Manager page request files in two ways. The first is by use of the ad-hoc page request API (page_request) and the second is the result of a pager notification generated by a Clarify business rule. In both cases, the result is a Page Request file. The Paging Manager processes a specified number of page request files, then sleeps. The Paging Manager will process request files in the directory specified by FCS_DIR.

Process API Page Request File

In this path, the API has already created a record in the page_request table. The pager number and message is retrieved from the record (the objid was provided as a field in the page request file). The next step is to create the Serial Interface page request file. If the customer does not wish to track page requests, the page request record is deleted. Otherwise, if the customer wishes to create business rules for page requests, a time_bomb is created.

Process NOTIFY Page Request File

In this path, the Clarify Rule Manager has processed a business rule whose notification has resulted in a page. In this case, no page_request record has been created, so the first step is to create one. The next step is to create the Serial Interface page request file. The objid of the page_request record is used as part of the Serial Interface cross reference number. Should the customer not be interested in tracking NOTIFY page requests, the page_request record is deleted. Otherwise, the page_request record is updated with the actual request time.

Processing Page Response Files

The second major function of the Paging Manager is to process responses from the RPA Serial Interface. The Paging Manager will process response files in the directory specified by PAGING_DIR. The Paging Manager processes response files in order of creation date, oldest first. The customer can select to only process selected responses from RPA. The page_request record can be updated with filtered RPA Serial Interface page responses. The Paging Manager will optionally create a page_response record, time bomb record, and activity log entry when processing filtered RPA Serial Interface page response files. Once processed (or ignored due to filtering), the Paging Manager renames the RPA Serial Interface response file to eliminate it from consideration for later processing.

Implementation

This section provides requirements, a list of the files that are included in this product, installation instructions, and other implementation considerations.

Configuration

There are many configuration options that can be set for the Paging Interface. These are contained in environment variables, Clarify configuration items, and an initialization file.

You must set up all of the necessary variables, .ini files, and configuration items for your Paging Interface to work correctly.

Environment Variables

There are only two environment variables that need to be set up for the Paging Interface. These should be set up on the database server that runs the Clarify Rule Manager background process.

FCS_PAGING_REQUEST_DIR

Each database server must set this environment variable. It indicates the directory in which the Paging Interface pager_clerk program will create the Paging Manager Request Files. It should be

the same as the PAGING_DIR value specified in the *pg.ini* file used by the Paging Manager process.

LC_ALL or LC_STRING
Per-client environment variable for internationalization. Default is EN_US (US English).

Configuration Items

The following configuration items are stored in the file *pg_config.dat*, and may be modified with your favorite text editor. Once the items have been modified, you will import the configuration file with the Clarify Data Exchange Tool (dataex). See below for more instructions on importing files using dataex.

Item Name	Description
pg_Track_API_REQ	When pages are made through the page_request API this item will indicate whether to keep or delete the page request record after processing. (set i_value to 0 to delete, 1 to keep)
pg_Create_API_Request_Event	When pages are made through the page_request API this item will indicate whether or not to create a time_bomb record for triggering other business rules. (set i_value to 0 to omit, 1 to create)
pg_Create_API_Request_Focus_act_Entry	When pages are made through the page_request API this item will indicate whether or not to create a focus object activity log entry. (set i_value to 0 to omit, 1 to create)
pg_Track_NOTIFY_Request	When pages are made through a notification process this item will indicate whether to keep or delete the page request record after processing. (set i_value to 0 to delete, 1 to keep)
pg_Response_Filter	This item indicates which response codes to process of those returned by RPA. It can be set to include or exclude selected response codes. The str_value for this item should be set as follows: INCLUDE ALL = Process all responses (same as EXCLUDE NONE) INCLUDE <filter> (Process response codes specified by filter.) INCLUDE NONE (Process no responses (same as EXCLUDE ALL)) EXCLUDE ALL (Exclude all responses (same as INCLUDE NONE)) EXCLUDE <filter> (Exclude response codes specified by filter.)

	EXCLUDE NONE (Process all responses (same as INCLUDE ALL)) <filter>=Process selected responses. The filter is a set of space-separated 2-digit response codes. The response codes are defined by RPA. These codes are: AC=RPA accepted page RJ=RPA rejected page TK=RPA accepted CTIM page request TN=RPA rejected CTIM page request TO=Time-out occurred DL=RPA Delivered page to communications network ND=RPA was unable to deliver page to communications network. CF=RPA received a confirmation NC=RPA did not receive a confirmation For example, if the administrator wished to track AC and DL only, then the <filter> would be: "AC DL" (without the quotes).
pg_Track_API_Response	When pages are made through the page_request API this item will indicate whether or not to track all responses returned by RPA or only the latest response. (set str_value to ALL or LAST)
pg_Create_API_Response_Event	When pages are made through the page_request API this item will indicate whether or not to create a focus object activity log entry for each response returned by RPA. (set i_value to 0 to omit, 1 to create)
pg_Create_API_Response_Focus_Act_Entry	Create or don't create focus object act_entry for response to API request (set i_value to 0 to omit, 1 to create)
pg_Track_NOTIFY_Response	When pages are made through a notification process this item will indicate whether or not to track all responses returned by RPA or only the latest response. (set str_value to ALL or LAST)
pg_Create_NOTIFY_Response_Focus_Act_Entry	When pages are made through a notification process this item will indicate whether or not to create a focus object activity log entry for each response returned by RPA. (set i_value to 0 to omit, 1 to create)
pg_Max_Request	This item is used by the Paging Manager process to determine how many page

	requests to process during each cycle. (set i_value to the desired number. 10 is the default)
pg_Sleep	How many milliseconds to sleep since zero requests or responses were processed? (set I_value to the number of milliseconds)
pg_Max_Response	This item is used by the Paging Manager process to determine how many page responses to process during each cycle. (set i_value to the desired number. 10 is the default)
pg_Customer_Name	This item is used to reflect the RPA provided customer name in the page request files that are sent to RPA. (set the str_value to the appropriate 8 character text)
pg_Customer_ID	This item is used to reflect the RPA provided customer ID in the page request files that are sent to RPA. (set the str_value to the appropriate 7 character text)
pg_Use_NOTIFY_Focus_Tag	Should the system allow focus tags for normal system pages? (set the I_value to 1 for yes, to 0 for no).
pg _WARN40	Some paging systems can only handle 40 characters. If this item is set (i_value = 1) then the ad-hoc paging form will warn the user when 40 characters have been entered
pg_WARN80	Some paging systems can only handle 40 characters. If this item is set (i_value = 1) then the ad-hoc paging form will warn the user when 80 characters have been entered
pg_WARN120	Some paging systems can only handle 40 characters. If this item is set (i_value = 1) then the ad-hoc paging form will warn the user when 120 characters have been entered
pg_WARN240	Some paging systems can only handle 40 characters. If this item is set (i_value = 1) then the ad-hoc paging form will warn the user when 240 characters have been entered

Note: The pg_Ignore_Filter has default settings to prevent processing of certain character types. You can add and delete these character types to meet your needs. Also, LC_ALL and LC_STRING have default settings of EN_US. These can also be

Menu Items

In the file *pg_global.cbs* there are two menu items that are added to the Clarify system. The code from that file is as follows:

```
pc_menu.MenuBarID = 1002
pc_menu.AddItem "Desktop", "Page Monitor", "OpenPageMonitor"
pc_menu.AddItem "Desktop", "Page Request", "OpenPageRequest"
```

Each of these two items adds a new menu item to Clarify. The first allows the Paging Interface Page Monitor GUI to be called from Clear Support. The second allows the Paging Interface Page Request GUI to be called from Clear Support.

The administrator may wish to modify these entries. They can be removed entirely. The MenuBarID property can be changed to make the menu item appear in a different Clarify application (Valid MenuBarID properties are described in the ClearBasic Object Reference Guide in the MenuBarID section). In addition, the menu item can be moved to a different menu by changing the name of the menu in the first argument to the AddItem call. Finally, the name that appears in the menu bar can be modified by changing the second argument in the AddItem functional call.

Strings

The Paging Interface contains a set of strings that are used for labels, captions, and messages. Every string that is displayed by the Paging Interface is contained in a new database table, and the input for that table is contained in the file *pg_strings.dat*. The list of strings used by the Paging Interface follows in this section. To modify any of the strings, edit the *pg_strings.dat* file with a text editor, and change the string. Then import the file (as detailed below).

There are two primary modifications that are commonly made to this file:

Changing the English value for the string. Simply locate the string object in the file and change the STRING="<the string>" line. Make sure that you leave any parts of the string that start with a percent sign and contain a number. For example, %1s, %2d. These are used for parameter substitution.

Adding a new version of the string for another locale. Make a copy of the English version of the string (from the OBJECT line to the END_OBJECT line). Change the locale from EN_US to the locale of your choice, and change the STRING="<the string>" line. For more information, contact First Choice Software.

The strings for the Paging Interface are as follows:

ID	PURPOSE	STRING
	Labels for Page Monitor form (1990)	
11000	Form Header	List Pages for …
11001	Name Label	First/Last Name:
11002	Pager Label	Pager:
11003	Date Label	Start/End Date:
11004	List button	List
11005	Responses Label	Responses for selected page …
11006	Case ID Label	ID:
11007	Case Option Label	Case
11008	Subcase Option Label	Subcase

11009	Close Page Button	Close Page
11010	Done Button	Done
11011	Check Box Label	List Closed Pages
	Labels for Page Request form (1991)	
11020	Case Label	Case ID/Title:
11021	Name Label	First/Last Name:
11022	Pager Label	Pager/Alt. Pager:
11023	Message Label	Message:
11024	Send Time Label	Send Time:
11025	Send Button	Send
11026	Done Button	Done

10200 - 10399 Error/Informational Messages

ID	STRING
11100	Requested Date/Time has elapsed. Please respecify.
11101	Warning...message length has exceeded length of '%1s' characters
11102	Pager or Alt. Pager must be filled in.
11103	Message must be filled in.
11104	Date/time entered is not valid.
11105	Focus type is not valid
11106	Employee type is not valid
11107	pg.ini line ignored: '%1s'
11108	Parsing Error: '%1s'
11109	config_itm ignored: '%1s'
11110	Cannot open file: '%1s'

Requirements

This version of the **Paging Interface** requires the following:

Clarify Version:	5.0 or later
Clarify Tools:	Data Dictionary Editor (ddeditor) User Interface Editor (uieditor) Clear Basic Toolkit (cbex) Data Exchange Utility (dataex)
Other:	Valid accounts registered on the RPA System and properly configured RPA Serial Interface - OR - Properly configured and licensed PageMate Software Zip and unzip tools

Packaging

The Paging Interface 1.3.1 is shipped as a zip file, depending upon your Clarify server platform:

- For example – pg.1.3.1.zip

Installation Tree

It is recommended that you uncompress the zip file containing the Paging Interface into an fchoice subdirectory created at the top of the Clarify install tree. For example, on NT, should your Clarify server install tree be "c:\clarify", then:

Switch to "c:\clarify" directory.
Create an "fchoice" directory, if necessary.

Switch to “fchoice” directory.
Unzip into “c:\clarify\fchoice” directory.

After uncompressing, you will have the following installation tree:
c:\clarify\fchoice\pg

The following files are provided with this product:

File Name	Purpose
paging_user.pdf	Paging Interface user guide and reference document
cb	Directory containing the ClearBasic code
forms	Directory for the forms files
import	Directory containing data files to be imported
manager	Directory containing the Paging Manager program
pager	Directory containing the pager_clerk executable program
schema	Directory containing the schema modifications

The following files are provided in the *cb* directory:

File Name	Purpose
pg.dir	Clear Basic directives file for compilation
pg1990.cbs	Clear Basic code for the page monitor form
pg1991.cbs	Clear Basic code for the page request form
pg391.cbs	Clear Basic code to augment the activity log form
pg_global.cbs	Global Clear Basic routines for the Paging Interface
STRING.cbs	Global internationalized string routines

The following files are provided in the *forms* directory:

File Name	Purpose
pg1990.dat	Page Monitor Form
pg1991.dat	Page Request Form
pg391.dat	Activity Log Form
pg472.dat	Business Rule Select Form

The following files are provided in the *import* directory:

File Name	Purpose
pg_config.dat	Initial data to import for the Paging Interface
pg_strings.dat	Internationalized strings for the Paging Interface
pg_br50.dat	Business rule objects for the Paging Interface

The following files are provided in the *manager* directory:

File Name	Purpose
pg_daemon.cbs	Paging Manager program
pg.bat	Batch file example for starting the Paging Manager
pg.ini	Initialization file for the Paging Manager process
log_cf.bat	Executable script for processing response files
log_cf.cbs	Sample program for taking action on a Page Response record

The following files are provided in the *pager* directory:

File Name	Purpose
pager_clerk.exe	Pager Clerk executable for Windows NT
pager_clerk.c	Pager Clerk C source code
pager_clerk.h	Pager Clerk include file

The following files are provided in the *schema* directory:

File Name	Purpose
pg_schema.sch	Schema modifications required for the Paging Interface

Installation

The **Paging Interface** files may be installed on any server machine that can execute the Clarify Data Dictionary Editor (ddeditor), the Clarify User Interface Editor (uieditor), the Clarify Data Exchange Tool (dataex) and the Clarify ClearBasic Compiler (cbex). The ClearBasic source files (*.cbs) should be installed in the same directory as the directives file (pg.dir).

Now that you have unpacked the install package, you will be performing the following tasks during the installation of **Paging Interface**:

Updating the Clarify Schema
Importing configuration files
Importing updated forms
Setting up resource configurations
Compiling ClearBasic code
Compiling/copying the pager clerk program
Setting up the Paging Manager Process
Setting up environment variables
Setting up the Paging Interface
Oracle DBMS_LOCK Package
Start all necessary Processes

Note: It is highly recommended that you first install the **Paging Interface** on a test system and become familiar with its operation before installing it on a production

New Objects

First Choice Software has defined the following new objects to implement the **Paging Interface**:

New Tables/Views

Table Name	Table Number	Purpose
fc_string	3500	Internationalized Strings
fc_list_hdr	3501	Internationalized Lists
fc_list_level	3502	Internationalized Lists
fc_list_elm	3503	Internationalized Lists
fc_list_locel	3504	Internationalized

m		Lists
fc_locale	3505	Internationalization
page_request	3550	Page requests
page_response	3551	Page responses
view_pages	3552	Join of page requests, cases, subcases, and employees

If you are already using any of the above table numbers, you may modify the *pg_schema.sch* file and change them to other, unused numbers in the user-defined range (2000-4999).

New Forms

The following new forms will be added to the Clarify system for the Paging Interface:

Form Name	Form Number
Page Monitor Form	1990
Page Request Form	1991

If you are already using any of the above form numbers, you may modify the appropriate .dat file and change them to other, unused numbers in the user-defined range (1005-4999).

New Activity Codes

There will be two new activity codes defined for this interface. These codes will be used to audit any actions taken with the Paging Interface. The new codes are:

Code Value	Code Number
Page Request	90220
Page Response	90221

If you are already using any of the above activity numbers, you may modify the *pg_config.dat* file and change them to other, unused numbers in the user-defined range (90000-100000). You will also have to modify the appropriate .cbs files. For example, if you were to change the page request activity code you would have to change the number 90220 in the *pg_global.cbs* source file and the *pg_daemon.cbs* source file.

These activity logs can be added to your Clarify system, or not. A variety of configuration items control if the activity log entries are added for page requests and responses. See the configuration item section for more details.

New Business Rule Events/Properties

The Paging Interface system is augmented to generate custom business rules. For example, imagine a business rule as follows: If a page request does not receive a response within 30 minutes, a notification will be sent to the page request's creator. This will be a simple business rule with the Paging Interface.

A series of new properties (table_prop_name) are supplied. These must be imported with the dataex utility into the database. They are stored in the file *pg_br50.dat*.

The new business rule events are as follows:

Event	Event Number
Page Ad-Hoc	USER2530
Page Request	USER2531

Page Response	USER2532

The *Page Ad-hoc* event is used internally by the Paging Interface, and should not be used by administrators. The default business rule imported with *pg_br50.dat* uses this event.

Administrators should concentrate on the latter two events (which are supported with the supplied Business rule form (#472)). Every time a page is sent via the page request form or the page API, an event of type *Page Request* is added to the system. Every time a two-way response is logged, an event of *type Page Response* is added to the system. Currently, normal business rule notification via pagers does not create a new event, since the business rule that triggered the page can be used for any other notification conditions.

These business rules can be added to your Clarify system, or not. A variety of configuration items control if the activity log entries are added for page requests and responses. See the configuration item section for more details.

Existing Objects

The following changes will be made to existing Clarify objects:

New Fields Added to Existing Tables

The following fields will be added to Clarify objects:

Table Name	Field Name	Data Type	Comments
employee	x_pg_custname	varchar(8)	Customer Name provided by paging service
employee	x_pg_custid	varchar(7)	Customer ID provided by paging service

Relations Added to Existing Tables

The following relations will be added to Clarify objects:

Table Name	Relation Name	Cardinality	Comments
employee	employee2page_request	OTM	The employee related to this page_request
case	case2page_request	OTM	The case related to this page_request
subcase	subcase2page_request	OTM	The subcase related to this page_request
act_entry	act_entry2page_request	MTO	Activity log entries related to this page_request

Updating the Schema

This section details the steps necessary to modify the Clarify database schema for the Paging Interface.

Back up the Database

Prior to making schema changes, it is always a good idea to backup your database using standard backup procedures.

Importing the New Schema

To make the required schema changes:

1. Start the Data Dictionary Editor.
2. Choose *Save To File* from the *File* menu.
3. Type **my_db.sch** and press the *OK* button.
4. With a text editor, edit the *my_db.sch* file (it will be stored in the DD Editor directory) and make the changes listed in the *pg_schema.sch* file provided in the Paging Interface install tree.
5. Choose *Apply Changes* from the *Actions* menu.
6. Select the *my_db.sch* file.
7. When asked, press *Continue*.
8. Review the results presented. If any errors, fix them and repeat steps 5-7.
9. If there are no errors, press the *Apply Changes* button.
10. On the next form press the *Upgrade* button.
11. When the upgrade completes, press *Done*.
12. Exit the Data Dictionary Editor

Note: Please ensure that the "application" field in the fc_string table is set to an empty string rather than to NULL. This seems to only be a problem when using the Oracle database server. If the field is set to NULL, you can enter the "update table_fc_string set application='' where application is null" database command.

Import Data Files

There are several files that must be imported into the Clarify system with the Data Exchange tool. These files are:

pg_config.dat	Overall configuration items
pg_strings.dat	Internationalized strings
pg_br50.dat	Business Rule Properties

The pg_config.dat file has two items that need to be modified before the import is done. The str_value for "pg_Customer_Name" needs to be changed to the customer name assigned by the paging service. The str_value for "pg_Customer_ID" needs to be changed to the customer ID assigned by the paging service. Locate the two config items in the file and modify the str_value for each one using any text editor available.

To import the data files, execute the following command for each:

```
<path>\dataex-user_name <user> -password <pass> -db_server
<serv>
       -db_name <db> -imp <file>
```

Where:

<path>	Is the path to the dataex program
<user>	Is the system administrator user
<pass>	Is the system administrator password
<serv>	Is the database server name
<db>	Is the database name
<file>	Is the name of the file to import

Note: The slash between <*path*> and *dataex* should be a forward slash for UNIX systems.

Note: If the <*file*> is not in the current directory, it must be preceded by the path to the directory it is in.

Note: Each file should import with 0 errors and 0 warnings. If there are any errors or warnings, the dataex.mes file should be investigated for the reason.

Importing the User Interface Forms

The Paging Interface is provided with two new forms that need to be imported.

- pg1990.dat
- pg1991.dat

The Paging Interface also includes two modified forms that need to be imported.

- pg391.dat
- pg472.dat

Note: If your system already has a customized version of the activity log form (#391), you may add the customized code (pg391.cbs) to your form module. If you have already have a customized version of the business rule form (#472) you may add the Page request object and event to that form (as they are defined in the provided pg472.dat – just import the pg472.dat form, and use it as a model for your customized form.

The form files are imported with the following command:

```
<path>\dataex –user_name <user> -password <pass> -db_server
<serv>
        -db_name <db> -imp <file>
```

Where: <path> Is the path to the dataex program
<user> Is the system administrator user
<pass> Is the system administrator password
<serv> Is the database server name
<db> Is the database name
<file> Is the name of the file to import

Note: The slash between <*path*> and *dataex* should be a forward slash for UNIX systems.

Note: If the <*file*> is not in the current directory, it must be preceded by the path to the directory it is in.

Note: Each file should import with 0 errors and 0 warnings. If there are any errors or warnings, the dataex.mes file should be investigated for the reason.

Adding a New Command Button to existing forms

Depending on how the interface is to be implemented, you may want to add a method to an existing form to bring up the Page Request or Page Monitor form. The modification to a form is simply to add one new button:

Type: Command (push) button
Name: BTN_LOAD_PAGE_REQUEST
Caption: Page Request (or anything you would like for the caption)

CB source can then be defined for the new button so that an action takes place. The example below will load the request form with employee and subcase default values.

```
Sub btn_LOAD_PAGE_REQUEST_Click()
  Dim req_form As Integer
  Dim bulk_rt  As New BulkRetrieve
  Dim rt_list  As List
  Dim emp_rec  As New Record
  Dim case_rec As New Record

  Set emp_rec.RecordType = "employee"
  Set case_rec.RecordType = "subcase"

  bulk_rt.SimpleQuery 0, "employee"
  bulk_rt.AppendFilter 0, "first_name", cblike, "J%"
  bulk_rt.RetrieveRecords

  Set rt_list = bulk_rt.GetRecordList(0)
  If rt_list.count > 0 Then
    Set emp_rec = rt_list.ItemByIndex(0)
  End If

  bulk_rt.SimpleQuery 0, "subcase"
  bulk_rt.AppendFilter 0, "id_number", cbEqual, "2-1"
  bulk_rt.RetrieveRecords

  Set rt_list = bulk_rt.GetRecordList(0)
  If rt_list.count > 0 Then
    Set case_rec = rt_list.ItemByIndex(0)
  End If

  req_form = show_page_request_form(case_rec,emp_rec)
End Sub
```

Set up Resource Configurations

After all of the form files have been imported successfully, the forms must be added to the proper resource configuration(s) in UI Editor. You must decide which user or groups of users should see the new versions of the imported forms.

To add the forms to the resource configuration(s):

1. Start the User Interface Editor.
2. Choose *Resource Configs…* from the *Select* menu (File/Open resource configuration for Clarify 6.0+)
3. Press the *List* button.
4. Select the resource configuration you want to modify and press the *Open* button.
5. Type **pg** in the text box to the right of the starts with dropdown list and press the *List* button.
6. Select the forms you wish to add to the resource configuration from the left grid control, and press the *Copy>>* button.
7. If any confirmation dialogs appear, press the *OK* button to continue.

8. Press the *Replace* button.
9. After the save is completed, press the *Done* button.
10. Repeat steps 4-9 for any other resource configurations you wish to modify.
11. Exit the User Interface Editor.

Compile ClearBasic Code

You can now compile the supplied Clear Basic code against the forms you have imported.

When Clarify starts up it runs the routine 'initialize_app' (if one exists). Only one routine with this name will be run on client start-up. The module <module> contains an 'initialize_app' routine for this application. If some other custom application has a version of this routine, only one of these versions will execute. It is REQUIRED that you merge any existing version of 'initialize_app' into the version in module <module>. Then, replace the original version of 'initialize_app' with the merged version and recompile it. This will prevent the original version from overwriting the merged version if you recompile your original code, and it will ensure that all your custom applications will still be executed.

Not doing this as specified may cause this application or previous customizations to function improperly.

Before you compile the code, you should edit the *pg.dir* file supplied. Verify that the Clarify version (the 1st number after the form number for the lines that contain a letter "F") is the same as the form that is saved in the desired resource configurations. Also, the user version (2nd string after the "F") should be the same.

To compile the code modules, make sure that the code modules (*.cbs) and the directives file (*pg.dir*) are located in the same directory. Enter the following command to compile the code:

```
<path>\cbex -dir pg.dir
            -batch
            -overwrite
```

Where:

<path> is the path to the cbex program.

When the compilation is completed, you should look at the *cbex.log* file to check on the status of the compilation. There should be two rows for each file compiled that start with the word **SUCCESS**. If there are any failures (there shouldn't be), check the above instructions to make sure that your compilation environment is set up correctly.

Note: If you already have a global module with an *InitializeApp* routine in it, you must incorporate the code in rpa_global.cbs into the existing one. Clarify only allows one InitializeApp routine to exist. If you do not have such a routine, you may simply compile the provided rpa_global.cbs module.

Compiling *pager_clerk*

This section details the steps necessary to compile *pager_clerk* program for Paging Interface. This step is not needed if running Windows NT, since the pager_clerk.exe file is included in the zip file. Before compiling, copy the pager_clerk.c and pager_clerk.h files to the proper directory. Using a standard compiler, compile the program with the following syntax (the example below assumes compiling for a UNIX system: for NT, change the –DUNIX to –DNT).

Then compile with:

```
cc –DUNIX –o pager_clerk pager_clerk.c
```

Note: You must copy the pager clerk executable program to the rule manager directory.

Paging Manager Process

Copy the *pg_daemon.cbs* and *pg.ini* files to the Clarify rulemgr directory. Create three directories on the server, one for holding pager_clerk output (FCS_DIR), one for holding files for transfer to or from the paging service (PAGING_DIR), and one for holding log files for the Serial Interface (LOG_DIR).

Edit the *pg_daemon.cbs* file to indicate which paging service is being used, RPA or PageMate. There are two lines of code in the pg_daemon subroutine that assign either "pm" or "rpa" to the pm_rpa variable, and the inappropriate line should be commented out or deleted.

Edit the *pg.ini* file as needed for your environment. In particular, the three directories listed in the last paragraph must be set up in the file.

You should also make sure that the restart variables are set. Specifically you must make sure that the *SHELL_FILE* variable points to the correct path for cbbatch and the pg_daemon.cbs program, so that it may restart itself. The *CYCLE* variable should be set to a reasonable number of cycles.

Also, the PAGING_DIR directory must match the directory set up for the RPA serial interface program. This directory is how the two systems communicate (by placing files in the directory).

Environment Variables

There are two environment variables that should be set up.

1. You must set up the environment variable **FCS_PAGING_REQUEST_DIR** on the machine that is running the Clarify rule manager. It must be set to the same directory as the PAGING_DIR value in the *pg.ini* file (see the previous section).
2. On each client machine that is running the Clarify client you should set up the environment variable **LC_ALL**. This variable should contain the locale name that the client desires for displaying internationalized strings. First Choice Software uses the standard X/Open locale names. By default, US English should be used. The value to use for US English is:

```
LC_ALL=EN_US
```

Setting Up the Serial Interface

The serial interface is a program that is supplied by RPA. Please contact RPA for the program and installation instructions. If PageMate is being used, the PageMate software must be installed and configured. Please contact Systemetrics (www.system.com) for the program and installation instructions.

Make sure that when you configure the program (by editing the *parms.ini* file) that the two directories listed in that file agree with the directories set up for the First Choice software. Specifically, the serial interface has a directory called “DATA_PATH” that must be the same as the PAGING_DIR set up above.

Oracle DBMS_LOCK Package

Note for Oracle database users: As of version 1.5, the ClearBasic *sleep* command has been replaced by a call to the Oracle procedure *dbms_lock.sleep*. The **dbms_lock** package is created as part of the standard installation when **catproc.sql** is run. If you system does not have it installed, it can be created by running the script **$ORACLE_HOME/rdbms/admin/dbmslock.sql**, after connecting to the database under the **SYS** account. There is a public synonym in place for the package, but under 8.1 execute rights are granted to the role

EXECUTE_CATALOG_ROLE, so you may not be able to call the function under a low-privilege account. Please refer to your Oracle documentation for more details.

Start Processes

The Paging Interface is now installed and ready to use. Make sure that all of the proper processes are started:

- Clarify Rule Manager
- First Choice Paging Manager (pg_daemon.cbs)
- RPA Serial Interface or PageMate software

All of these processes must be running for the Paging Interface to function properly.

Limitations

There are a few limitations known to exist with the Paging Interface:

1. Internationalization issues. There are two that are currently known. All have been opened as issues with Clarify. When they are corrected, a patch will be issued for this product.
   - The column headers of the custom lists (forms 1990, 1991) cannot be set at run time. To localize, you must have a new version of each form per locale.
   - There is not currently an internationalized date dialog available. The date dialogs used on the page monitor form (#1990) and the page request form (#1991) use the standard Clarify date/time dialog.

   For more information about internationalization and localization, please contact First Choice Software, Inc.

2. Clarify Solution PD07450 describes a known bug with the Shell() function when used on a unix platform. You must pass in the full path of the shell with the –c option before passing in the command file with it's arguments. In this case, the assignment for the SHELL_FILE parameter in the pg.ini file should be as follows:

   SHELL_FILE=/usr/bin/sh -c\ /export/home/clarify/80/solaris_server/rulemanager/cbbatch -db_server <server_name> -db_name <database_name> -user_name sa -password sa -f pg_daemon.cbs -r pg_daemon

Internationalization

The Paging Interface will be internationalized and localized, using the First Choice Software Inc.'s Internationalization library. Thus, all strings that are displayed on custom forms may be changed to another language/locale, simply by changing an environment variable. The same is true for user-defined lists defined for the Paging Interface. Finally, strings that are displayed can also be stored in the database in multi-locale format.

The Paging Interface will be localized and internationalized, but it will not be shipped in a translated state. Any translations to other languages are the responsibility of the customer.

There are a variety of environment variables that can be set to control various aspects of the internationalized code. For the Paging Interface, only two such variables are used.

LC_ALL – Sets all internationalization features to the desired locale
LC_STRING – Sets string handling features to the desired locale

Note: For the Paging Interface, you may set either variable. If neither environment variable is set, the Paging Interface defaults to US English (LC_ALL = EN_US).

Note: First Choice Software uses the standard locale identifiers defined by X/Open. Each individual user may set their environment variables as they wish. Thus, two different users may be seeing the Paging Interface displayed in different languages at the same time against the same database.

Note: All internationalization is optional. The base Paging Interface product will be shipped in US English. No additional work must be done for that environment.

Trademarks

The following are trademarks of First Choice Software, Inc. and may not be used without the express written consent of First Choice Software, Inc.

- Workflow Series
- Administration Series
- Productivity Series
- Development Series
- Interface Series
- Select Sampler
- Commitment Plus
- Queue Groups
- Contact Merge
- Survey Taker
- ClearBasic API Toolkit For Clear Support
- ClearBasic API Toolkit For ClearQuality
- ClearBasic API Toolkit For ClearLogistics
- ClearBasic Internationalization Toolkit

How to Contact Us

For more information about other First Choice Software, Inc. products, or if you have any questions or feedback about this product, please contact us at:

First Choice Software, Inc.
8900 Business Park Drive
Austin, TX 78759
(512) 418-2905
www.fchoice.com

API Reference

Most of the critical functions of the Paging Interface are implemented as ClearBasic methods. These methods are provided with the Paging Interface so that advanced users may call on these APIs as needed.

In addition, the Paging Interface functions can be imbedded in new customization code that individual administrators may wish to add to their GUI. For example, a new button could be added to the Employee form. When pressed, it creates a page request to the employee with a message to check their email. This new routine could be written with a call to the supplied page_request method, thus simplifying the coding.

All of the Paging Interface methods are provided in two formats: function and subroutine. If called from the GUI, the function version will almost always be used. The subroutine version is provided primarily to be called from the CBBatch utility. There are two reasons that the CBBatch program requires a subroutine version:

1. CBBatch does not know what to do with the return value from the function. It will crash if you call on a function directly from CBBatch.
2. CBBatch cannot call a method with a boolean argument. So the subroutine version uses strings of "True" and "False" instead of the boolean values.

One reason administrators might wish to use CBBatch is to create a page request as the action of a business rule. There are many possibilities for designing business rules with this mechanism.

There are two arguments that are found in all of the Paging Interface APIs. Passing information through either argument is optional. Both arguments are described here for informational purposes:

- rec_focus – This is the case or subcase record that the page_request is related to (if it is related to a case or subcase).
- rec_emp – This is the employee record that the page_request is related to (if it is related to an employee).

The next section of this document lists the available APIs in the Paging Interface. Each API is documented with sections for:

- Syntax (both function and subroutine)
- Parameters
- Return Values
- Description
- Examples

Page Request

Function

```
Public Function page_request(page_num As String, _
                             message As String, _
                             req_time As String, _
                             rec_emp As Record, _
                             rec_focus As Record) As Integer
```

Subroutine

```
Public Sub page_request(page_num As String, _
                        message As String, _
                        req_time As String, _
                        rec_emp As Record, _
                        rec_focus As Record)
```

Parameters

Name	Description
page_num	The pager device number as registered with a paging service. Specify an empty string if you wish the API to use the employee's pager number. The employee is specified by rec_emp. Either page_num or rec_emp must be specified. If both are specified, page_num will be used as the pager number.
message	The message to be sent. Must be compatible with pager type. The string should be empty for numeric pagers. The string should contain only numeric characters for digital pagers. Note that many text pagers have message length limits such as 240 characters.
req_time	The scheduled delivery time of the page_request.
rec_emp	The employee being paged. If page_num is an empty string, then this employee's record will be used to determine the pager number.
rec_focus	The 'focus' record of the page. Can be a case record, a subcase record, or nothing. If a case or subcase record is specified, then the 'page_request' record will be relate to the specified record.

Return Code Values

Return Code	Description
0	Page request successfully submitted.
-1	Pager number could not be determined from page_num nor provided employee record.
-2	Focus record specified is not a case nor subcase.
-3	Message provided is longer than 2048 bytes.

Description

This API will allow the generation of a page request from any form. The pager number, message, and delivery time are passed to the routine as parameters. In addition, a related employee or case/subcase can be passed and the relationships to the page request will be set up. This process also creates a time bomb event that will be processed by the Paging Manager server process.

Example

Page employee Sam Adams with a message of “Conference Call at 10:00 am”. This message should be delivered at 9:00 am on the day of the call. The page is not related to a case in Clarify.

```
Sub btn_page_request_Click()
  Dim bulk_rt  As New BulkRetrieve
  Dim rt_list  As List
  Dim emp_rec  As Record
  Dim case_rec As New Record

  Set case_rec.RecordType = "case"

  bulk_rt.SimpleQuery 0, "employee"
  bulk_rt.AppendFilter 0, "first_name", cbEqual, "Sam"
  bulk_rt.AppendFilter 0, "last_name", cbEqual, "Adams"
  bulk_rt.RetrieveRecords
  Set rt_list = bulk_rt.GetRecordList(0)
  If rt_list.count > 0 Then
    Set emp_rec = rt_list.ItemByIndex(0)
  End If

  page_request("5124180001”,"Call at 10:00 am”, _
               #07/31/1998 09:00#, emp_rec, case_rec)
```

Show Page Request Form

Function

```
Public Function show_page_request_form(rec_focus As Record , _
                                       rec_emp   As   Record)   As
Integer
```

Subroutine

```
Public Sub show_page_request_form(rec_focus As Record , _
                                  rec_emp As Record)
```

Parameters

Name	Description
rec_focus	The 'focus' record of the page. Can be a case record, a subcase record, or nothing. If a case or subcase record is specified, then the 'page_request' record will be relate to the specified record.
rec_emp	The employee being paged. If page_num is an empty string, then this employee's record will be used to determine the pager number.

Return Code Values

Return Code	Description
	This function returns the page request form instance.

Description

This API will launch the Page Request form. This form provides a simple page request GUI that can easily be plugged in to most Clarify forms or menu bars. The API also demonstrates the use of the page_request() API described above.

Example

Display an empty Page Request form to allow the generation of a new page request.

```
Sub btn_show_request_form_Click()
  Dim emp_rec  As New Record
  Dim case_rec As New Record

  Set emp_rec.RecordType = "employee"
  Set case_rec.RecordType = "case"

 show_page_request_form(emp_rec, case_rec)

End Sub
```

Show Page Monitor Form

Function

```
Public Function show_page_monitor_form(rec_focus As Record , _
                                       rec_emp As Record, _
                                       max_age as Long, _
                                       show_closed as Boolean) As
Integer
```

Subroutine

```
Public Sub show_page_monitor_form(rec_focus As Record , _
                                  rec_emp As Record, _
                                 max_age as Long, _
                                 show_closed as Boolean)
```

Parameters

Name	Description
rec_focus	The 'focus' record of the page. Can be a case record, a subcase record, or nothing. If a case or subcase record is specified, then the 'page_request' record will be relate to the specified record.
rec_emp	The employee being paged. If page_num is an empty string, then this employee's record will be used to determine the pager number.
max_age	The max_age will filter page requests by age (seconds). Any pages older than the specified age will be filtered out. If max_age is zero or negative, no age filtering will occur.
show_closed	The show_closed if True will show both 'open' and 'closed' page requests. If False, only 'open' page requests will be shown.

Return Code Values

Return Code	Description
	This function returns the page monitor form instance.

Description

This API will launch the Page Monitor form. This form provides a page request monitor GUI that can easily be plugged in to most Clarify forms or menu bars.

Example

Display the Page Monitor form to allow the monitoring of all open page requests.

```
Sub btn_show_monitor_form_Click()
  Dim emp_rec  As New Record
  Dim case_rec As New Record

  Set emp_rec.RecordType = "employee"
  Set case_rec.RecordType = "case"

  show_page_monitor_form(case_rec, emp_rec, 0, False)

End Sub
```