print | close

Run Your Watch Definitions and Create a Watch Event Exit Program

iPro Developer
Carsten Flensburg
Carsten Flensburg
Thu, 06/28/2012 - 6:00am
APIs by Example

Click here to download the code bundle.
To report code errors, email ***iPro Developer***

In the previous installment of APIs by Example, I discussed the concept of watches introduced with release 5.4. Watches let you define a set of conditions identifying a specific system event, in turn causing a watch event exit program to be called. In a user-written exit program, you can optionally perform the actions prompted by the watch event, such as forwarding an email reporting the event, collecting event data, and running appropriate system commands. Alternatively, you can specify the QPDETWCH system watch program, which takes the information supplied to the watch exit program, generates an XML service request, and places that service request on the Service Monitor queue.

Last time, I presented my new Add Watch Definition (ADDWCHDFN) and Remove Watch Definition (RMVWCHDFN) commands. Creating a watch definition lets you save the watch attributes in a table and easily restart the watch following an IPL or an issued End Watch (ENDWCH) command. Here, I offer the Start Watch Definition (STRWCHDFN) command used to activate a previously created watch definition. I also present examples of how to write and take advantage of watch event exit programs, and include a skeleton exit program that you can use as a starting point for writing your own watch event exit programs for system-monitoring or message-forwarding purposes (see "How to Compile," below, for a list of sources included in the downloadable code bundle).

The STRWCHDFN Command

Figure 1 shows the fully prompted STRWCHDFN command. You'll find information about the command and its parameters in the associated help text panel group. Note that if you specify the special value *WCHDFN for the command's *User profile* parameter, the user profile that created the watch definition will run the Start Watch (STRWCH) command. For the involved profile swap to work correctly, your user profile must have *SECADM special authority.

Figure 1: Start Watch Definition (STRWCHDFN) command prompt

```
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -


                       Start Watch Definition (STRWCHDFN)

 Type choices, press Enter.

 Session ID . . . . . . . . . . .                    Name, generic*, *ALL
 User profile . . . . . . . . . .   *CURRENT         *CURRENT, *WCHDFN
 Verify watch program owner . . .   *YES             *YES, *NO



- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

For security reasons, if you specify USRPRF(*WCHDFN), the *Verify watch program owner* parameter will ensure that the watch program is owned by the user profile that created the watch definition. This option prevents a potential misuse of a privileged user profile by disallowing an unauthorized update directly against the watch definition table. If this integrity check is enforced and a user profile mismatch is found, a diagnostic message will be issued to the STRWCHDFN command caller, but the start watch definition processing continues with the remaining watch definitions.

You can place the STRWCHDFN command in the system startup program (QSTRUPPGM) or in a subsystem autostart job to automatically restart your watch definitions following a system IPL. Once active, a watch will call the specified watch event exit program whenever the watch criteria are met. Generally, you define three types of watch events, depending on the watch attributes you specify for the watch event domain you want to monitor:

- a message ID or a range of message IDs in a message queue or a job log
- LIC log entry in the Licensed Internal Code (LIC) log
- Product activity log entry in the Product Activity Log (PAL)

In addition to the above three watch event types, the STRWCH and ADDWCHDFN commands let you call the watch event exit program at the point where the watch starts and/or ends. This provides an option for you in the exit program to communicate or register whenever a watch starts or ends. The watch event exit program must implement the interface in Figure 2.

Figure 2: Watch event exit program parameters

```
Watch for Event Exit Program

  Required Parameter Group:

  1  Watch option setting              Input      Char(10)
  2  Session ID                        Input      Char(10)
  3  Error detected                    Output     Char(10)
  4  Event data                        Input      Char(*)

QSYSINC/H member name: ESCWCHT
```

The *Watch option setting* parameter defines the type of watch event that caused the exit program to be called, as discussed earlier. The *Session ID* contains the watch session identifier specified when the watch started. *Error detected* is an output parameter that the watch event exit program uses to communicate whether the exit program contains an error, while processing the watch event. If an error condition is met, the exit program will specify the special value *ERROR for this parameter and, as a consequence, the watch terminates. Otherwise, the exit program should return blanks, indicating that processing completed normally.

In fact, the watch runtime considers returning anything other than blanks an error. Note that if you rely on the watch event exit program being called whenever the watch ends, terminating the watch by indicating an error will not cause the exit program to be called again for *End watch* processing. You therefore need to conduct any actions that your exit program requires at the end of a watch session, immediately before an error condition is returned.

The final exit program parameter defines the information describing the watch event that caused the exit program to be called. The actual data provided in the *Event data* parameter varies, depending on which of the five types of watch events described earlier applies. The simplest form relates to a start or end watch event, in which case, only four bytes defining the length of the provided watch information are returned. At this point, it doesn't seem that useful, but sticking to this convention would let IBM add more event information in the future.

The message ID, LIC log, and PAL entries are all described in the detail required to process, register, and communicate the watch event adequately and independently of other resources. Please refer to the watch event exit program documentation for a thorough discussion of this parameter (see the link in Find Out More, below). To help you get started writing your own watch event exit programs, I've included one general sample exit program and two exit program examples each designed to handle one specific watch event.

As for the latter, the first example forwards an email message to the watch event user profile (the one that started the watch), in case one of two specific messages is sent to the QSYSOPR message queue. Release 7.1 introduced the two messages in question via a PTF delivering a change in system behavior whenever a job exceeds its CPU or storage limit. Previously, such jobs were immediately terminated. But if you installed the mentioned PTF, those jobs are now instead held and, therefore, require speedy operator intervention to cope with the situation at hand.

The other exit program example is called whenever a CPF4131 level check message is issued to any job's job log. The message is then forwarded to the LOGLVLCHK message queue together with another message, identifying the job in which the CPF4131 exception was generated. This lets you investigate such exceptions retrospectively, without having to retain and locate the job log. Feel free to adapt and change the example exit programs as you see fit to meet your shop's specific requirements.

Two Potential Issues

When testing the watch event exit programs included with this article, I encountered a couple of issues in the watch implementation on the IBM i. One problem caused a session ID of all nulls to be passed to the exit program for the *End watch* event. Another, and potentially much more critical, issue surfaced when I tested an exit program forwarding all messages sent to the *Critical system messages* message queue QSYSMSG to an email account. Starting a watch against the QSYSMSG message queue specifying a watch message ID of *ALL caused a never-ending stream of QSCWCHPS prestart jobs servicing the watch to be started and then immediately ended, as long as the watch remained active.

As a result, this could lead to the system job tables filling up, severely degrading the performance of your system, which ultimately might require an IPL to recover. Preventing the prestart jobs from being cycled by specifying MAXUSE(*NOMAX) for the QSCWCHPS prestart job entry could, however, negatively impact your system's performance. IBM is currently looking into providing an improvement to the current implementation but has made no promises at this point. I'll report back to you in an upcoming issue of APIs by Example about the ongoing investigation's outcome. I'll also include any PTF numbers that will become available as a result of IBM's efforts.

How to Compile

Below, you'll find instructions about how to create the Start Watch Definition command and the watch event exit programs. The following sources are included with the code download associated with this article:

CBX249—RPGLE—Start Watch Definition–CPP

CBX249H—PNLGRP—Start Watch Definition–Help

CBX249V—RPGLE—Start Watch Definition–VCP

CBX249X—CMD–Start Watch Definition

CBX250—RPGLE—Watch Event Exit Program–Skeleton

CBX2501—RPGLE—Watch Event Exit Program–CPI112D/CBX112E Msg IDs

CBX2502—RPGLE—Watch Event Exit Program–CPF4131 Message ID

CBX249M—CLP—Start Watch Definition–Build command

To create all above command objects, compile and run the CBX249M CL program, following the instructions in the source header. Note that due to the requirement of authority adoption, the CBX249M program must be run by a user profile with *ALLOBJ special authority. For more information, please refer to the CBX249M source header.

You'll also find compilation instructions in the respective source headers of the individual sources, as well as the associated add watch definition commands in the exit program source headers. The code included with this article takes advantage of the most recent watch enhancements, and therefore must be adapted to run on pre-7.1 releases.

Find Out More

An Example Watch Program to Use for the STRWCH Command for LIC Logs

Configuring and Benefiting From IBM i Watches

IBM i 7.1: Jobs Exceeding Their CPU or Storage Limits are now Held

STRWCH - Watch Exit Programs Explained with CL Example

What's New in Watches?

IBM I 7.1 Information Center documentation

Monitoring APIs

Retrieve Watch Information (QSCRWCHI) API

Service APIs

Start Watch Command or API Exit Program (QPDETWCH) API

Watch for Event Exit Program

Articles at iProDeveloper.com:

"APIs by Example: Watch Definitions and the Start Watch (QSCSWCH) API"

"Sending Messages from RPG to a Syslog Server"

"Using the Create and Send MIME Email API"

Source URL: http://iprodeveloper.com/rpg-programming/run-your-watch-definitions-and-create-watch-event-exit-program