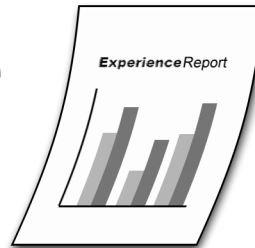


iSeries



Tuning prestart job entries

Experience Report



iSeries



Tuning prestart job entries

Contents

Tuning prestart job entries	1
Set the number of prestart jobs	1
Change job attributes for prestart jobs	4
References and resources	7
Disclaimer	9

Tuning prestart job entries

This document shows how to manage prestart jobs to improve overall system performance. Prestart jobs are jobs that start running before the work arrives. A prestart job entry in a subsystem description tells the system how many jobs to create and how to manage the prestart jobs. What you tell the system affects how well the system works. To get the best performance from your system, you need to tell the system what workload to expect.

The following sections contain the details:

“Set the number of prestart jobs”

This section uses the Display Active Prestart Jobs (DSPACTPJ) command and the Change Prestart Job Entry (CHGPJE) command to get the right number of jobs.

“Change job attributes for prestart jobs” on page 4

This section creates a new job description and sets the job message queue maximum size (JOBMSGQMX) and job message queue full action (JOBMSGQFL) job attributes.

Set the number of prestart jobs

You should have enough prestart jobs started by the subsystem so that work is handled as it arrives rather than waiting for new jobs to be started. While the system is handling its normal workload and information about the workload is available, take the following steps:

1. Use the Work with Subsystems (WRKSBS) command to get a list of all active subsystems. For each subsystem in the list of active subsystems, use option 5 to display the subsystem description.

On the **Display Subsystem Description** panel, use option 10 to display prestart job entries. If there are no prestart job entries for that subsystem description, continue with the next subsystem in the WRKSBS list.

2. On the **Display Prestart Job Entries** panel, use option 5 to display details for the prestart job entry. Make a note of the current settings for Initial number of jobs, Threshold, and Additional number of jobs.
3. For each prestart job entry in the subsystem description, enter a Display Active Prestart Jobs (DSPACTPJ) command. For example:

```
DSPACTPJ SBS(SUBSYSTEM) PGM(PJPGMLIB/PJPROGRAM)
```

If the DSPACTPJ command is not currently allowed, the prestart job entry is not active and does not need to be changed. Continue with the next prestart job entry or the next subsystem description.

4. Use the DSPACTPJ information to get an estimate of your workload. The DSPACTPJ command produces a display that looks like this:

```
-----  
                                Display Active Prestart Jobs                                SYSTEM  
                                08/06/03 07:35:00  
Subsystem . . . . . : SUBSYSTEM      Reset date . . . . . : 08/06/03  
Program . . . . . : PJPROGRAM        Reset time . . . . . : 07:23:03  
Library . . . . . : PJPGMLIB        Elapsed time . . . . . : 0000:11:57  
  
Prestart jobs:  
Current number . . . . . : 122  
Average number . . . . . : 21.4  
Peak number . . . . . : 122  
  
Prestart jobs in use:  
Current number . . . . . : 120  
Average number . . . . . : 17.7  
Peak number . . . . . : 120
```

More...

Press Enter to continue.

F3=Exit F5=Refresh F12=Cancel F13=Reset statistics

```

-----
Display Active Prestart Jobs                                SYSTEM
                                                            08/06/03 07:35:00
Subsystem . . . . . : SUBSYSTEM      Reset date . . . . . : 08/06/03
Program . . . . . : PJPROGRAM       Reset time . . . . . : 07:23:03
Library . . . . . : PJPGMLIB       Elapsed time . . . . . : 0000:11:57
Program start requests:
Current number waiting . . . . . : 0
Average number waiting . . . . . : .0
Peak number waiting . . . . . : 1
Average wait time . . . . . : 00:00:00.0
Number accepted . . . . . : 120
Number rejected . . . . . : 0
-----

```

Bottom

Press Enter to continue.

F3=Exit F5=Refresh F12=Cancel F13=Reset statistics

Find the *prestart jobs in use* section and the value for the *peak number*. In this example, the value is 120. This number is an estimate of your peak workload. Make a note of this value, it will be used in the following steps.

Find the *program start requests* section and the value for the *peak number waiting*. You may need to page down to see this field. In this example, the value is 1. This number tells you how well the system is handling the arrival of new work. Make a note of this value, it will be used in the following steps.

- If DSPACTPJ shows a zero (0) for the peak number of prestart jobs in use, the prestart job entry is not being used by your workload and therefore does not need to be changed. Continue with the next prestart job entry or the next subsystem description.
- Choose a value for the THRESHOLD parameter. When the pool of available jobs is reduced below this number, more jobs are started. Starting jobs takes time. Meanwhile, more requests for work may arrive. Set THRESHOLD to a value of at least one plus the number of requests that can arrive while new jobs are being started.

In this example, the value chosen is 10. This is an estimate of arrival of work requests, a guess based on the peak number of jobs in use. This is not an accurate analysis of hard-to-get measurements.

Refer to the notes you took in an earlier step. If the current setting for THRESHOLD is high enough, the *peak number waiting* will be zero. If the peak number waiting is not zero, add this number to your current THRESHOLD value and compare the result to the estimated value based on arrivals. Use the larger value. The sample DSPACTPJ information shows a value of 1 which means the current value for THRESHOLD is too low. The current setting plus one is less than the estimate of 10. For this example, we use the value 10.

- Choose a value for the initial number of jobs (INLJOBS) parameter. INLJOBS specifies the number of jobs that are started when the subsystem is started. Also, INLJOBS is part of what the subsystem uses to decide if there are too many prestart jobs waiting for work.

Refer to the notes you took in an earlier step. Use the *peak number of prestart jobs in use* as the estimate for peak workload, add the value for THRESHOLD, and use the result as the new value for INLJOBS. The DSPACTPJ information shows a peak of 120 prestart jobs in use and we have already chosen a THRESHOLD of 10, so the new value chosen for INLJOBS is 130.

- Choose a value for the additional number of jobs (ADLJOBS) parameter. ADLJOBS specifies the additional number of prestart jobs that are started when the number of available prestart jobs drops below the value specified on the Threshold (THRESHOLD) parameter.

When INLJOBS and THRESHOLD are high enough to avoid causing requests to wait, ADLJOBS can be fairly low. If INLJOBS is far below peak workload, ADLJOBS may need to be as high as THRESHOLD. In this example, the chosen value is 5.

Try to avoid large numbers. If you specify a large value for ADLJOBS, the subsystem will start a large number of jobs all at once. This can adversely affect system performance and it delays the subsystem's handling of other work.

9. Compare the newly chosen values with the values configured in the prestart job entry. To be sure to have enough prestart jobs, use the larger value for each parameter. Change the configured values by using the Change Prestart Job Entry (CHGPJE) command.

```
CHGPJE SBSD(SBSLIB/SUBSYSTEM) PGM(PJPGMLIB/PJPROGRAM)
      INLJOBS(130) THRESHOLD(10) ADLJOBS(5)
```

10. Continue with the next prestart job entry or the next subsystem description.

Details

Some additional details may help you make good decisions when following this procedure.

- If the THRESHOLD value is too small, work waits for new jobs to be started. In some cases, errors will occur because requests time out.

Consider an example where THRESHOLD is 2 and there are only two jobs waiting for work. When the next work request arrives, that request is given to one of the waiting jobs and additional jobs are started. In this example, two more requests arrive before the new jobs are ready. The first request is handled by a waiting job. The second request waits for one of the new jobs to become ready. For the example workload, THRESHOLD should be set to at least 3: one to trigger the creation of more jobs plus two for the number of requests that arrive while new jobs are being started.

- Because the subsystem starts jobs when they are needed, the subsystem also ends jobs when they are not needed. ➤ This happens for prestart job entries that specify a maximum number of uses (MAXUSE) greater than one. ⏪ The value for the INLJOBS parameter tells the subsystem how many jobs are needed. You need to get INLJOBS set correctly to prevent the subsystem from ending too many jobs.

If the INLJOBS value is too small, the subsystem will periodically start jobs because there are too few and end jobs because there are too many. Moreover, the system incurs the cost of starting new jobs at the time when the system is most busy.

- In the sample output from the DSPACTPJ command, the peak number of prestart jobs in use is 120 while the average number of prestart jobs in use is 17.7. This is not a high peak. This is a low average. By default, DSPACTPJ shows what has happened since the subsystem started. The average includes periods when the workload is zero.

Even when you use **F13** to reset the statistics and even when you carefully control the sample interval, the average number of prestart jobs in use is likely to be lower than the number you should tune to. A workload may have an average somewhere between 40 and 60 jobs and yet have lots of peaks between 100 and 120 jobs.

When INLJOBS is set to the estimated peak workload plus THRESHOLD, the subsystem does not have to start additional jobs unless the actual workload exceeds the estimated peak workload. If your workload has peaks that are relatively high and relatively infrequent, you may wish to set INLJOBS to a lower number.

- The procedure given in this report assumes that the peak load on a typical day is a typical peak load. If you gather more data, you might be able to produce a better estimate of your workload.

You can use the List Job (QUSLJOB) API or the Open List of Jobs (QGYOLJOB) API to periodically sample your workload. For some workloads, it helps to graph the results.

You do not need a perfect prediction for the number of prestart jobs. You only need to be close enough to avoid delays and timeouts.

- If THRESHOLD and INLJOBS are too large, there will be active jobs in the subsystem that are not needed. Starting and ending extra jobs takes more time when starting or ending the subsystem or when starting or ending the prestart job entry.

It is better to use values that are slightly higher than what is needed than to use values that are lower than what is needed. Having a few extra jobs is not a problem because these jobs are waiting for work and do not compete for memory or processors.

- Because prestart jobs were first used with communications devices, a request for work is called a program start request and the prestart job shows a status of **PSRW** (waiting for program start request) when it is waiting for work.

Change job attributes for prestart jobs

You should avoid having a very large number of very large job message queues. Large job message queues can consume storage, can lead to large joblogs which also consume storage, and can cause IPL performance problems when job message queues need recovery or cleanup during an IPL. This example shows how to change the job message queue full action (JOBMSGQFL) and job message queue maximum size (JOBMSGQMX) values for prestart jobs.

➤ **Note:** The QDFTSVR job description was introduced in release V5R3M0 to do some of this for you. ⚡

To limit the size of job message queues for prestart jobs without affecting other jobs, follow these steps:

1. Create a new job description to be used by the prestart job entries that currently specify USER(QUSER) and JOB(*USRPRF). You can use the Create Job Description (CRTJOB) command, but in this example we make a copy of the job description that is currently being used. Use the Display User Profile (DSPUSRPRF) command to determine which job description is currently being used. The default configurations use job description QDFTJOB in library QGPL.

```
DSPUSRPRF USRPRF(QUSER)
```

To avoid confusion with IBM^(R)-supplied objects, avoid names starting with the letter 'Q'. This example uses the name PJJOB as the name of the job description for the prestart job entries. Use the Create Duplicate Object (CRTDUPOBJ) command to make a copy of the job description currently used by the QUSER user profile.

```
CRTDUPOBJ OBJ(QDFTJOB) FROMLIB(QGPL) OBJTYPE(*JOB)
          TOLIB(QGPL) NEWOBJ(PJJOB)
```

2. Since QDFTJOB is owned by QPGMR, change the new job description to be owned by QPGMR. Use the Change Object Owner (CHGOBJOWN) command and the Grant Object Authority (GRTOBJAUT) command to get the object ownership and public authority set correctly.

```
CHGOBJOWN OBJ(QGPL/PJJOB) OBJTYPE(*JOB) NEWOWN(QPGMR)
```

```
GRTOBJAUT OBJ(QGPL/PJJOB) OBJTYPE(*JOB) USER(*PUBLIC) AUT(*USE)
```

3. Use the Change Job Description (CHGJOB) command to customize the job attributes. In this example, we use a value of 8 megabytes for the job message queue maximum size. Other values would also work as long as the limit is far less than 64 megabytes.

```
CHGJOB JOB(QGPL/PJJOB) JOBMSGQMX(8) JOBMSGQFL(*WRAP)
      TEXT('Job attributes for prestart job entries')
```

4. Look through all of the prestart job entries that are active on your system. Use the Work with Subsystems (WRKSBS) command to get a list of all active subsystems. Use option 5 to display the subsystem description. Use option 10 to display prestart job entries and use option 5 to display details for the prestart job entry.

If the prestart job entry specifies USER(QUSER) and JOB(*USRPRF), use the Change Prestart Job Entry (CHGPJE) command to specify the new job description.

```
CHGPJE SBS(SBSLIB/SUBSYSTEM) PGM(PJPGMLIB/PJPROGRAM)
      JOB(QGPL/PJJOB)
```

If the prestart job entry specifies a job description, use the Change Job Description (CHGJOB) command to change the JOBMSGQMX and JOBMSGQFL values in that job description.

```
CHGJOB JOB(JOBDLIB/JOBNAME) JOBMSGQMX(8) JOBMSGQFL(*WRAP)
```

Details

The QDFTJOB job description is used by many prestart job entries and is used in many other places in the system. This example creates a single new job description named PJJOB. The new job description is used by many prestart job entries but it is not used elsewhere. To use different values for different prestart job entries, use a different job description for each entry. Some prestart job entries already have unique job descriptions.

Some job attributes for prestart jobs cannot be changed using this procedure because they do not come from the job description that is used when starting the job. Many servers that use prestart jobs will swap user profiles and then use the Change Job (QWTCHGJB) API to change a subset of the job attributes. The changed job attributes come from the job description used by the user profile to which the prestart job has swapped. Refer to the JOBC0300 format of the Change Job API for more information.



For some job attributes, the job description may indicate that the value is to be taken from a system value. When you change the system value, the change affects all jobs that get their job attribute from the system value. Changing the value in the job description affects only those jobs that get their job attributes from that job description.

References and resources

iSeries^(TM) Information Center

- Work Management
- System Values

Manuals

- V4R5 Work Management 
- Job Scheduler for OS/400 

Disclaimer

Information is provided "AS IS" without warranty of any kind. Mention or reference to non-IBM products is for informational purposes only and does not constitute an endorsement of such products by IBM.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.



Printed in USA