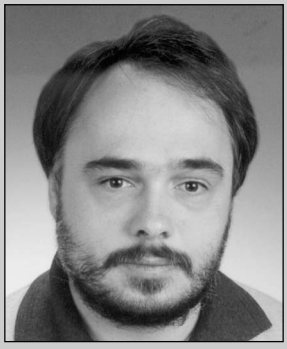
Optimizing, Monitoring, and Fine-Tuning a Newly Upgraded Release 4.x Spool Service

Uwe Krüger



Uwe Krüger studied Computer Science at the Technical University of Karlsruhe, Germany. He joined SAP AG in 1995 as a member of the CCMS (Computing Center Management System) group, where he is responsible for development in the output management area. Uwe can be reached at uwe.krueger@sap.com

A lot of processing takes place between the time an R/3 user hits the print button and the time that his print job actually shoots out of the target printer. First, the R/3 application generates a printer-independent output data stream, which gets stored as a spool request in the database. Then, an output request is created for this spool request and the desired output device.[1] These database entries must then be committed before the print processing can start on the spool server. Next, a message is sent to the target spool server, where the output request will eventually land in a dispatcher queue awaiting assignment to the first available spool work process. Once that job has been processed, it's off to the host spool system, which ultimately directs the job to the target printer.

Could you discern the key element that distinguishes this sequence of events from those that take place in a Release 3.x environment? Notice that I said the "first" available spool work process. As of Release 4.0, there can be more than one spool work process on a spool server. This was not the case with earlier releases. In the pre-Release 4.0 environments that many of you are running, a spool server can have only one spool work process, and that spool work process has to shoulder processing responsibility for all output requests and related management activities. Needless to say, processing bottlenecks can be a common occurrence. You will realize far better throughput and performance on a Release 4.x server that is running multiple spool work processes. Configuring multiple spool work processes for one spool server not only helps cope with rising output load on an instance, it also allows the processing of output requests to take place in parallel with the processing of administrative tasks.

[1] You can see these output requests using transaction SP01.

Just how many spool work processes should you set up on your Release 4.x spool server?

That's the question we will tackle in this article as I show you how to weigh the following factors in order to make that determination:

✔ What is the total output-request load on the server? For example, how many output devices[2] are assigned to the server, and how many and how large are the requests generated for these devices?

✔ Can all these output requests be processed in a parallel fashion, across multiple spool work processes? Or, do some require sequential processing?

✔ Is frontend printing supported on the server?

✔ Do you plan to "classify" spool work processes on the server such that some spool work processes will be dedicated to production or high-volume print jobs? Or, will all spool work processes be able to process any type of output request?

✔ Will you be setting aside spool work processes for management tasks such as polling of host spool systems and host spool system connection checks?

I will walk you through some simple scenarios to illustrate how you actually apply these factors in a real-world situation. Since Releases 4.0, 4.5, and 4.6, successively, introduce new features into the mix, I will be sure to point out release-specific concerns along the way.

[2] Don't confuse output devices with printers. In this article, the distinction between the two is very important. When I refer to an "output device" (or "device" for short), I'm talking about the entity that is the *logical* target for an SAP output request and that directs the request to a spool server, where it is processed by a spool work process. From there, it's off to a host spool system, which can direct the job to the target printer.

Six Steps for Determining the Right Number of Spool Work Processes

While your pre-upgrade configuration (which uses just one spool work process) will work just fine when you upgrade to any 4.x release, I strongly recommend that you increase the number of spool work processes so that a minimum of two are running on the server. This will offset the additional load introduced by the Release 4.x management tasks (see sidebar, "Spool Server Management Tasks — What You Need to Know About the Various 4.x Releases") and also obviously bolster performance of output request processing.

So, is two the right number of spool work processes? Probably not. That would be a configuration that's sure to beat the performance of just one spool work process, but it probably falls short of the correct number.

Is the correct number 20, then? Probably not. So many spool work processes would needlessly consume too many server resources, since each additional spool work process would require additional memory and processor time. Someone's got to pay the price for these resources, and in the case of an application server that has been configured as a spool server, that would be your dialog users.

Your best bet is to configure as many spool work processes as are required to meet your print processing requirements, but no more than that.

Just how many spool work processes should you set up on your Release 4.x spool server? Your best bet is to configure as many spool work processes as are required to meet your print processing requirements, but no more than that.

Spool Server Management Tasks — What You Need to Know About the Various 4.x Releases

The spool service of a Release 4.x spool server is responsible for more tasks than ever before.

Aside from processing print requests and polling host spool systems (the responsibilities it anchored in Release 3.x), a Release 4.x spool server also shoulders responsibility for redirecting print jobs to other spool servers, checking the availability of the host spool system, spool system reorganization, and activating print jobs.

Summarized here are the features that have been introduced in successive 4.x releases and how these new features impact the number of spool work processes you will want to set up on your spool server.

Management Task	Release	Additional Spool Work Processes*
Polling of output requests: This can now be done by a maximum of one spool work process.	4.0A	Add one spool work process if you use extensive polling; but none if you use access method E with callback, or if you switch off the polling by device configuration.**
Connection check: When a host spool system is unavailable, the Release 4.0B spool server performs periodic connection checks to determine if the unavailable service is up and running, and therefore ready to accept an R/3 output request. (Applicable to access methods S and U.)	4.0B	A connection check can be performed by all spool work processes. But if you expect a high probability of unavailable remote host spool systems, one extra, dedicated work process should be added. An extra spool work process is not required if you do not access remote host spool systems with access methods S or U.***
Support for frontend printing: In Release 3.x/4.0/4.5, desktop print requests were handled by the dialog service, not by the spool service.	4.6A	Since frontend printing is now a function of the spool service, you'll need to account for this additional load. I recommend you add at least one additional spool work process for all Release 4.6 servers for this purpose, not just to the traditional spool servers.
Global management tasks: • Redirection of output requests • Scanning for lost output requests	4.0A	In general, all global management tasks are executed by a maximum of one spool work process. So, only one additional spool work process is necessary.
• Spool system reorganization — Instead of using report RSPO0041 or RSPO1041 to delete obsolete spool requests, this can now be done directly by the spool service. • Activation of print jobs — it is now possible to create output requests with a starting date and time.	4.6A	

* Number of spool work processes you should add to your spool server, above and beyond what you've set aside for processing traditional print requests to support the specified management task.

** Access method E is the certifiable interface for external output management systems. It supports both polling and callbacks.

*** S and U are access methods used to connect to remote host spool systems. S is an SAP-specific extension used by SAPlpd and U is an implementation for RFC1179.

The optimal number can be found in the six-step process shown in **Figure 1**.

I'll now take you through each of these steps, and show you how you can determine the right number of spool work processes according to your print processing needs. Admittedly, this six-step process is baffling in the beginning. At times, I will ask you to revisit previous steps, and at others, you may be told you can skip a step or two. Just "hang in there," as the saying goes. I promise, all the suggestions that are presented in these six steps will come together and make sense to you as you see them being applied in the sample scenarios that follow.

Step 1: Accounting for Frontend Output Requests

If you are upgrading to Release 4.6A, ask yourself whether or not frontend print requests will be processed on the Release 4.6A server. As of this release, all frontend printing requests (access method F) are handled by the spool service; they are no longer handled by the dialog service. This places an additional load on the spool service, so you will need to set up at least one additional spool work process to keep the same processing power for your traditional output. By default, the maximum number of spool work processes that can be used for frontend printing is limited to 1.[3] If one additional spool work process is not sufficient, you can, and should, override this default value.

To determine the optimal number of additional spool work processes, look at the load being generated by users for frontend printing on this server. Can it be adequately served (i.e., at no more than 70 to 80 percent utilization) by just a single spool work process? Remember, one spool work process would mean that all frontend print jobs (generated on this server) would be processed sequentially. Or, could they be processed in parallel instead? If the answer to this question is yes, adjust the work process restriction (i.e., increment the value) for frontend printing.

If the number you calculate here is greater than 1, don't forget to increase the maximum limitation for the processing of frontend print requests.

Step 2: Accounting for Output Requests That Must Be Processed Sequentially

Certain applications may rely on a specific order for the generation and processing of print requests. Take, for example, a document and an attachment that have been created sequentially as two separate requests, yet need to be printed in this sequential order.[4] These types of output requests cannot be spread across multiple spool work processes, where they would be processed in parallel with no guarantee of output of the document preceding output of the attachment; they must be processed by a single spool work process to ensure the correct order. If a spool work process starts to process such a request, it will be reserved as long as there are more jobs for this device, thereby limiting the parallel job processing. So, if you have many such output requests, it may be useful to add more spool work processes for these types of output devices.

To ensure sequential generation of print jobs, we introduced a special reservation scheme between output devices and spool work processes. Things are set up in the spool server to tap into this scheme automatically. In fact, when performing an upgrade from Release 3.x to Release 4.x, all *existing* output devices are automatically assigned this behavior. This ensures that if sequential processing is needed, it will get done. After an upgrade, it's the *new* devices that you configure, because by default, this option is turned off, and therefore requires you to intervene if you want sequential processing of output requests.

[3] Even in situations where your frontend printing load is very low and there is enough capacity available on other spool work processes, for reasons that will become clear a bit later in the article, I would not leave the number of spool work processes at 0, even on your dialog servers.

[4] As of Release 4.6C, it will be possible to build compositions of spool requests as a new spool request. Such a request may contain different kinds of documents and will be submitted as a single job to the host spool system.

Figure 1 Six Steps for Determining the Optimal Number of Spool Work Processes for a Release 4.x Spool Server

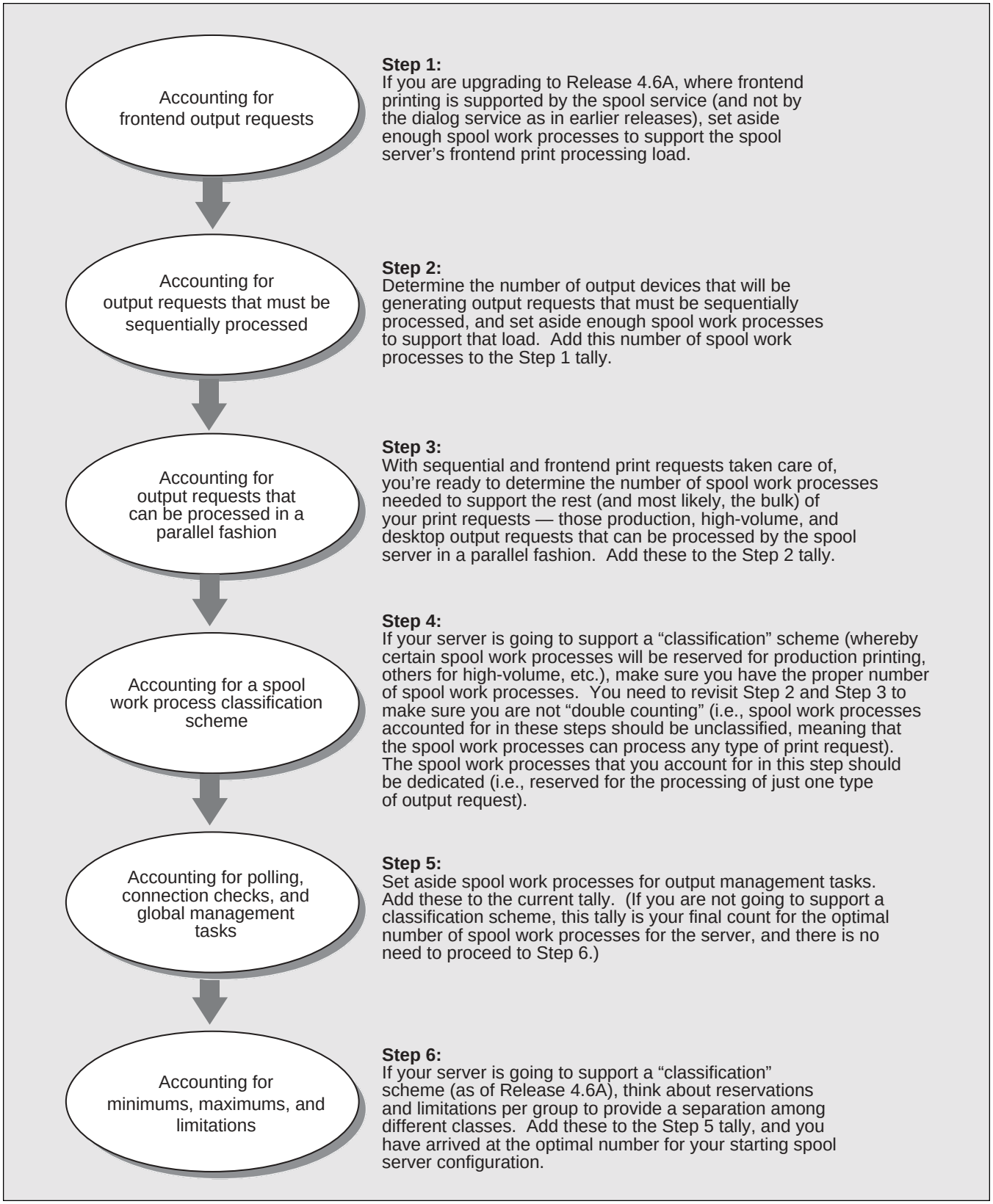
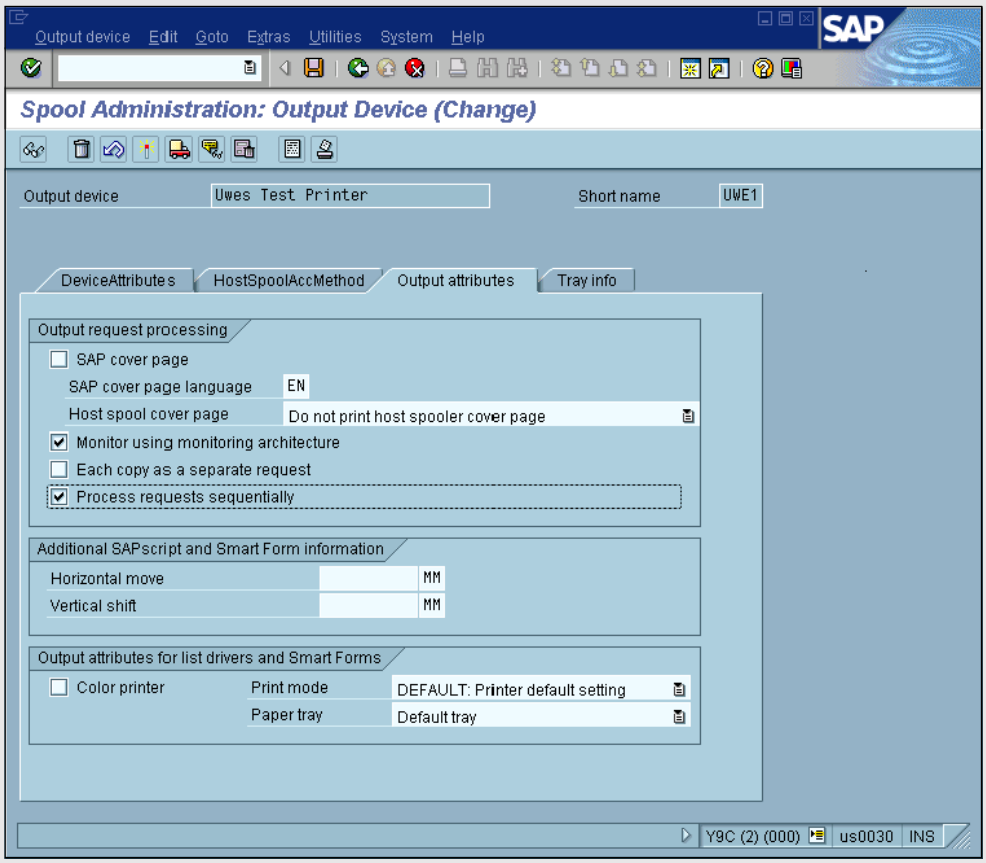


Figure 2 Configuring the Sequential Processing Option



To override the default Release 4.x spool server behavior, which assumes that output requests *can* be processed in a parallel fashion, go to the configuration screen and configure the output device for sequential request processing by checking "Process requests sequentially," as shown in **Figure 2**.

On the other hand, if sequential processing is not required, you will get better performance for your existing devices by reconfiguring them *not* to use this feature. After an upgrade, you should revisit the configuration of all existing devices, and check whether it is possible to switch off this feature. Please note, if your servers only have one spool work process, it does not matter whether this option is on or off.[5]

[5] If your Release 4.x spool server has just one spool work process (a configuration I do *not* recommend), this becomes a moot point.

Now we're ready to tackle the question, "How many spool work processes should I set aside for devices that require sequential processing of output requests?" The answer to this question is a function of how many devices will be making demands on the spool server at the same time. If three out of four output devices will be hitting the server up for sequential processing requests at the same time, set aside three spool work processes. If just one device will be issuing print requests at any given time, the number of required spool work processes is 1.

But, we're not done just yet! One more thing you need to consider is the load for a single output device. If it's too heavy, such that it cannot be handled by one spool work process, you will have to use multiple (identical) devices for different applications, instead

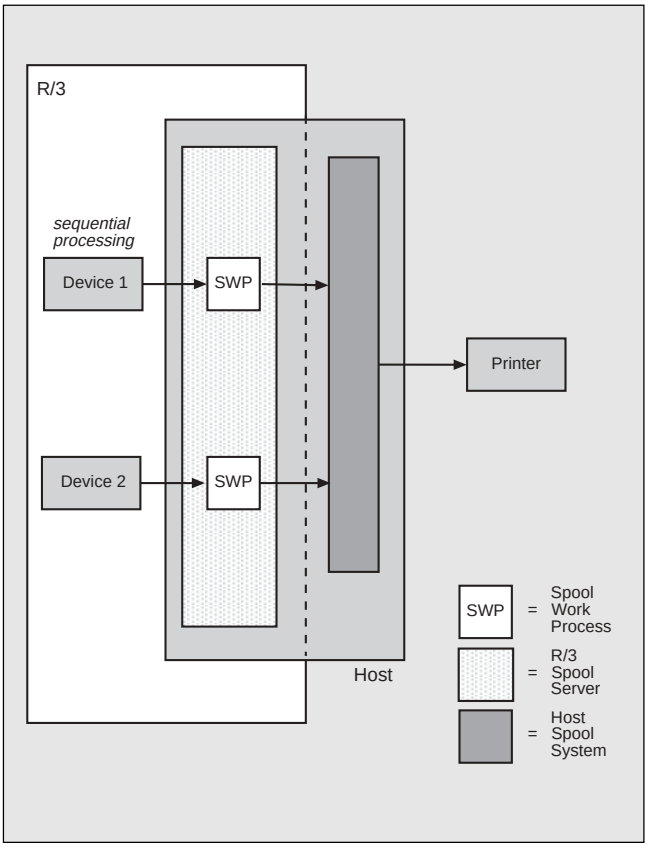
of only increasing the number of spool work processes, as shown in **Figure 3**.

You cannot simply increase the number of spool work processes used for a sequential device. That would defeat the purpose of establishing the exclusive processing of requests from a sequential device by one (and only one) spool work process. So, if you need more processing power for the requests of a single device, you must use multiple identical devices that are mapped to the same printer. Different applications can then use different devices to achieve parallel request processing.

You also can use a sequential device together with a nonsequential one (i.e., Device 1 in Figure 3 would be configured for sequential processing; Device 2 would not). Only applications requiring sequential processing would use the sequential device; all other output would be sent to the nonsequential device, thus allowing for parallel processing. But there's a catch: If both devices are used at the same time, the jobs generated for the different devices may get mixed up.

Alternatively, you could remove the sequential processing option (if possible) to allow for parallel processing of the requests for these devices. Whatever setup you decide upon, don't forget to take these additional devices into consideration when calculating the number of spool work processes in this step.

Figure 3 Assigning Multiple R/3 Devices to a Printer



Step 3: Accounting for Output Requests That Can Be Processed in a Parallel Fashion

With sequential and frontend print requests taken care of, you're ready to turn your attention to the rest (and most likely, the bulk) of your print requests — those production, high-volume, and desktop print requests that can be processed in parallel across multiple spool work processes. Again, the question is, how many spool work processes constitute the optimal number for the server. (Note that in this step, our focus is on unclassified spool work processes — those that can process any type of print request. In Step 4, we'll factor "exclusivity" into our calculations.)

To determine this number, you have to take into account two different criteria:

1. The first is to determine how many of the devices that are assigned to the server will be used at the *same* time. If there are 100 devices assigned to this server and you expect 10 of them to be used at the same time, you'll need 10 spool work processes. This number ensures that there will always be an available spool work process for end-user output requests. (A user printing on a device expects the request to be printed if no other requests are pending for this device, and does not know about the requests submitted by other users on other devices.)

2. The second is to estimate the size and number of output requests that are going to be generated by each device. The sum gives you an estimation of the total load that has to be handled by the spool service. Ideally, you want to achieve 80 percent utilization of a spool work process, leaving 20 percent of its bandwidth available for processing during peak periods of utilization.[6] You can use transaction RZ20 to see the idle time of your spool work processes. If you divide the total load by the load that can be handled by one spool work process, the result will be the minimum number of spool work processes required for your requests.

Step 4: Accounting for a Spool Work Process Classification Scheme

In my last article, I introduced you to the notion of spool server classification, whereby you reserve different spool servers exclusively for different types of print processing activities — production printing, high-volume printing, and desktop printing — then map the output devices to the appropriate servers.[7] With this type of spool server landscape, a production print job that goes to a server that handles only production print requests can't be delayed by a high-volume print job hogging the server.

Release 4.6B drives this idea down to the spool work process level. With this release, you can reserve spool work processes exclusively for just one type of output class.[8] (If you will not be "classifying" spool work processes, and instead decide to allow any spool work process to process any type of print request, skip this step and go directly to Step 5.)

If you plan to exercise this classification feature, you'll need to revisit Steps 2 and 3 to make sure that you are not double-counting spool work processes. In Step 2 and Step 3, make sure you account for unclassified spool work processes. In Step 4, factor classified spool work processes into the overall tally. You also need to make sure that you have enough spool work processes to support each type of print request.

> ✔ ***Tip***
> *Steps 2 and 3 can be skipped altogether if you take advantage of a Release 4.6A classification group called Reg (for regular, unclassified jobs). Using the Reg group lets you configure minimum and maximum limits for the number of spool work processes that can be used to service unclassified output requests.*

> ✔ ***Tip***
> *Step 1 sets aside the spool work processes you'll need for frontend printing. "Fro" is the name of the Release 4.6A group that separates your frontend requests from your traditional output. If you do not want this, just set the maximum number of allowed spool work processes for this group to the total number of spool work processes of the server.*

Step 5: Accounting for Polling, Connection Checks, and Global Management Tasks

Once a spool service has processed a print request and sends it off to the host spool system, that print job is no longer under the control of R/3. In order to track its status, the spool service has to send queries to the host spool system. Or, maybe the host spool system is not available and cannot accept jobs —

---

[6] Depending upon the processing power of your hardware, it's possible that you may be able to accommodate as many as 100 pages per minute per spool work process.

[7] "Achieving a More Manageable and Reliable R/3 Spool Server Landscape Using Release 4 Output Classifications, Logical Servers, and Alternate Servers," *SAP Professional Journal*, November/December 1999.

[8] By all appearances, an administrator does "reserve" spool work processes for the processing of one specific type of print job. In reality, any spool work process can be used for any task. The assignment of print processing tasks to a spool work process is actually done dynamically. If a dedicated task has to be performed, the configurable restrictions (i.e., minimum and maximum number of spool work processes that have been set aside for a particular printing class) are evaluated before each new task assignment. If assigning the new task to a free spool work process would violate these restrictions, the task stays unassigned and is delayed until its processing matches the restrictions.

what then? An R/3 spool service has to perform periodic connection checks and be ready to hand off the print job once the failed host spool system is up and running again. Outdated jobs need to be deleted. Delayed jobs need to be activated. Tables need to be scanned for lost jobs. Queues need to be maintained. All these administrative tasks constitute "overhead," and overhead requires processing time.

On a pre-Release 4.x spool server, these activities, as well as the output jobs themselves, all vie for just one spool work process. So, if a large print job, one that can consume several minutes of processing time, is at the head of the queue, you can be sure that no other print request will be processed — nor will any administrative tasks be performed until the large request is done.

As of Release 4.0A, management tasks no longer have to compete with print requests for processing time. Polling, connection checks, and global management tasks can all be relegated to separate spool work processes. How does this affect the number of spool work processes you need to set up? Let's take a look:

- If you are using access methods that poll the host spool system in order to track the status of a print job once it has left the R/3 system, add one additional spool work process. (If you don't use polling at all, you don't need to add one.) Polling is used automatically for all traditional access methods except F (frontend printing). If you use the OMS interface (access method E), depending on the partner product, you can choose between polling and callbacks for the job tracking. Whenever polling is used, you can switch it off at the device level. In this case, the request will always be done after a successful submission to the host spool system. You only need to add an extra spool work process for this task if you have many printers with polling or slow host spool systems.

- If you are using access method S or U on your R/3 spool server and you expect frequent unavailability of your remote host spool systems, add another spool work process to make enough resources available for the connection check task (this task is available as of Release 4.0B). Once it has been determined that a remote host spool system is unavailable (e.g., a desktop PC), this task will periodically check that system to see if it has become reachable again.

- As of Release 4.6A, two global tasks can be configured manually: *spool reorganization* and *activation of output requests with a starting time*. If you switch on these features[9] and at least one of those tasks has a short scheduling interval *and* you suspect that there will not be adequate bandwidth available on the server's other spool work processes to support the extra load generated by a high-frequency management task, add one additional spool work process. Do not worry whether other global tasks are required (such as the *request redirection in case of a server failure*); they are only scheduled if required and typically do not require an extra spool work process.

  In total, only one additional spool work process makes sense for global management tasks. Why? Because these tasks are sequentialized on a spool server, so that no more than one spool work process can be occupied by global management tasks, even if more than one task is pending at the same time.

Step 6: Accounting for Minimums, Maximums, and Limitations

If you opted *not* to employ a spool work process classification scheme, you're done. The tally you arrived at in Step 5 (having skipped Step 4) is the one you want to use as the best estimate of the number of spool work processes needed for your spool server. If you are going to use the Release 4.6B classification scheme, note that you can reserve a minimum number of spool work processes for a class, as well as a maximum number:

[9] To switch on these features, use the configuration screen for the spool system in transaction SPAD.

- The *minimum* number specifies how many spool work processes will be reserved for a dedicated class. You may, for example, want to set things up to ensure that there will always be a minimum of three spool work processes available for production output requests. (There's an implicit maximum limitation here for all other classes; it should be no more than the difference between the minimum number and the total number of spool work processes that can be used for all the other classes together.)

- The *maximum* number specifies a limitation on the use of spool work processes for a dedicated class. If load conditions warrant it, for example, production print requests can be allowed to spill over to up to three of the spool work processes configured on the spool server, but no more than three. All further spool work processes will be available for other kinds of output. (Here, there's an implicit minimum reservation for all other classes; the difference between this number and the total number of spool work processes are always available for all other classes together.)

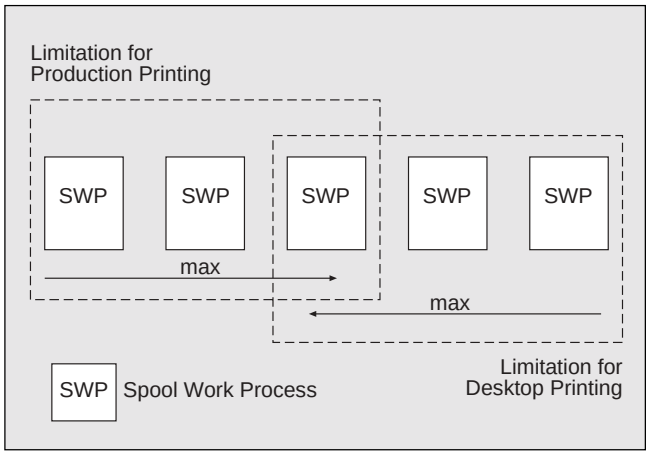
When establishing minimum and/or maximum limits among the different types of spool work processes, you will need to adhere to the following rules:

✔ You must leave a minimum of three spool work processes unreserved. So, for example, if you set up the spool server with five spool work processes, only two can be reserved for a specific print class. If you set it up with 10 spool work processes, up to seven can be reserved for specific print classes. If you fail to do this, the numbers will be decreased automatically during the server startup phase.

✔ You should check whether the sum of your exclusive reservations plus three is equal to or less than the tally (in Steps 1 through 5). If not, add the necessary number of spool work processes to your tally.

✔ Check the sum of the maximum limitations. You may find that the sum is larger than the total number of spool work processes, and that's okay. In this case, some of the spool work processes can be shared by multiple classes, depending on the load. For example, if you have a total of five spool work processes, and a maximum limitation of three for production and three for desktop printing, you would arrive at a sum of six. This means you have (6-5=1) 1 spool work process that can be used for both classes, as shown in **Figure 4**.

Figure 4 Using One Spool Work Process for Two Classes



✔ If the sum of the maximum limitations plus three is less than the total number of spool work processes, then there are spool work processes that never can be used (they will only be used if there is an overflow of the spool system processing queue).[10] In this case, you should increase some maximum values.

Now, tally the total number of spool work processes derived in Steps 1 through 6, and you're finally finished — you've figured out the optimal number of spool work processes you need for your upgraded server.

[10] Where does the number 3 comes from? In Release 4.6B, there may be a maximum of three management tasks that can be executed in parallel (polling, host spool checking, and the global management tasks). Since execution of these management tasks should always be possible, if you don't have more than three spool work processes, you won't be able to reserve one for a specific print class.

Some Simple Scenarios

I realize that it's one thing to *read* about how to derive the optimal number of spool work processes, and it's another to actually *do* it! So, before asking you to go off and try this six-step exercise on your own, I thought it would be helpful to walk you through a few sample scenarios. Bear in mind that I selected the print load estimates you are about to see because they are good for instructional purposes. These values are not representative of common customer output environments.

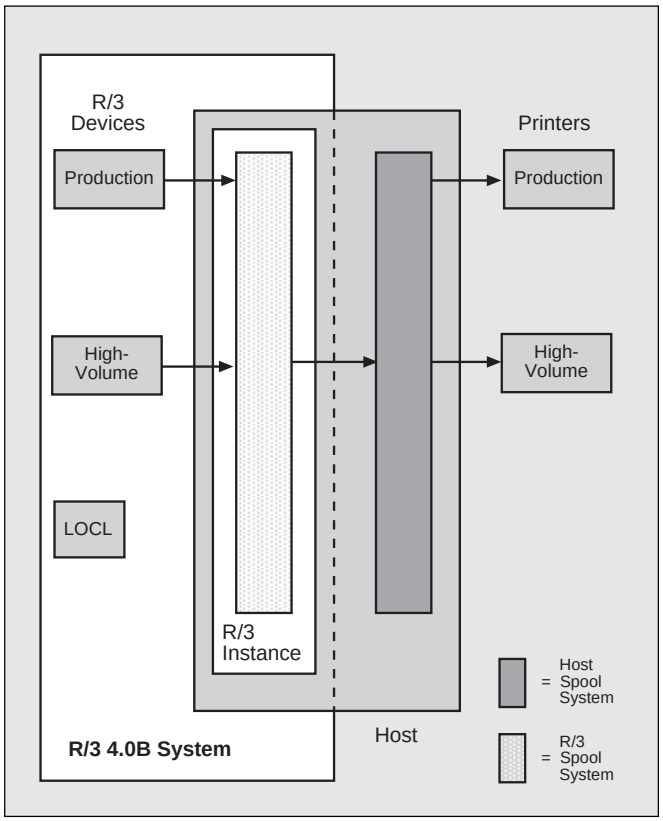
Scenario 1

In our first scenario (shown in **Figure 5**), we look at a Release 4.0B system with three output devices and one R/3 application/spool server. The first device is used for production printing. The second is used for high-volume printing. The third, LOCL, is a frontend printing (access method F) device. In this scenario, there are very few requests for high-volume print jobs during the day, but there are extensive requests for production printing.

Before we go any further with this exercise, do you see the major problem that is inherent in this setup? With just one application server, configured with just one spool work process, even though there may not be many high-volume output requests, when one does come along, it's going to prevent processing of the production print jobs.

What can you do to remedy this situation? Well, you *cannot* reserve a spool work process for processing production print requests or limit the number of work processes for high-volume printing, because that feature is only available in Release 4.6B, and this scenario centers around a Release 4.0B system. With just one high-volume output device at work here, however, there is a very effective workaround. You can simply declare the device as a *sequential output device*. This signals to R/3 that all its output requests must be processed by a single spool work process. It doesn't matter which spool work process fields the jobs for this device. So, "you can have your cake and eat it too," as the saying goes.

Figure 5 A Sample Release 4.0B System



Effective as this workaround may be, it doesn't answer the question, "How many spool work processes should be configured for this system?" Let's go through the steps….

- **Step 1:** In this scenario, frontend printing is not a factor. (Remember, in a Release 4.0B spool server, frontend print requests are handled by the dialog service, not by the spool server.) This step does not advance our tally, so our calculation remains at 0.
- **Step 2:** Yes, this spool server is going to support sequential processing. Since there is just one device whose jobs will be processed in this fashion, we set aside one spool work process. We then ask ourselves, "Can that one spool work process adequately handle the load for that one output device?" For this scenario, because there are so few high-volume print requests, the answer is yes. So, our calculation for this step remains at 1.

- **Step 3:** In this simple three-device scenario, we're left now with just the production device, and of the three, it's the only output device that will be in use all day long. How many spool work processes do we need to support this device? For starters, since we're talking about just one device, we set the count at 1.

  But then we ask ourselves, "Can that one spool work process adequately handle the load for that one output device?" This time, the answer is no. The production device, for argument's sake, is generating lots and lots of complex SAPscript delivery notes, which are being directed to a very fast printer. With just one spool work process, processing would be too slow for the printer. We will therefore declare that two spool work processes are required to adequately handle the load for this device. So, we add these two spool work processes to the tally, and the count now stands at 3.

- **Step 4:** As I explained earlier, a Release 4.0B server does not support a spool work process classification scheme. So, we skip this step.

- **Step 5:** Yes, we are using polling for our production and high-volume printers. (The access method F, frontend printing, device does not support status feedback.) For polling purposes, we set aside one additional spool work process. This brings our tally to 4: two spool work processes for the production device, one for the sequential output device, and one for polling.

  No, we are not using connection checks. We only have access method L and F devices, so the host spool systems in questions will always be accessible.[11] Therefore, no spool work processes have to be set aside for connection checks. And there's no reason to believe that four spool work processes will not be able to support the remaining global tasks. So, our tally remains at 4.

[11] Access method L is a command line interface for accessing a host spool system running on the same host where the R/3 spool service is running.

- **Step 6:** There are no reservations in Release 4.0B. Our tally stands at 4.

Let's now make things a bit more interesting by adding 10 desktop devices.... The users of these devices will produce just one output request at a time, so there will be never be more than one request to be processed at the same time.

Is it clear which steps need to be revisited? To accommodate the additional load, you will need to increment your count by 1, in Step 3. You will also need to revisit Step 5, because with desktop printers in the fray, we *will* employ connection checks. So, the number of spool work processes associated with Step 5 will need to be incremented by 1, as well. This increases the total number of spool work processes for this spool server from 4 to 6.

Would this calculation be different had we upgraded the system to Release 4.6A, rather than to Release 4.0B? Yes. If we add 10 desktop devices with Release 4.6A, we'd need 7 spool work processes. Let's go through the steps once more:

Step 1: In this scenario, we'd need one spool work process for frontend printing. Remember, as of Release 4.6A, frontend printing is done by the spool service. Our count is 1.

Step 2: We'd need one spool work process for the high-volume/sequential device. The tally becomes 2.

Step 3: Here, we'd need three spool work processes: two for the production device, and one for the desktop devices. Our tally is now 5.

Step 4: The general classification scheme is available with Release 4.6A, and we've opted not to use the Fro/Reg classification scheme beyond Step 1. Remember, the default maximum limitation for frontend printing is 1 (as calculated in Step 1), so the limitation does not need to be changed. So, our tally remains at 5.

Step 5: We'd need two spool work processes to accommodate polling and connection checks. This brings our count to 7.

Step 6: With no reservations here, our tally remains at 7.

Scenario 2

In this scenario, shown in **Figure 6**, let's assume we have a Release 4.0B system with:

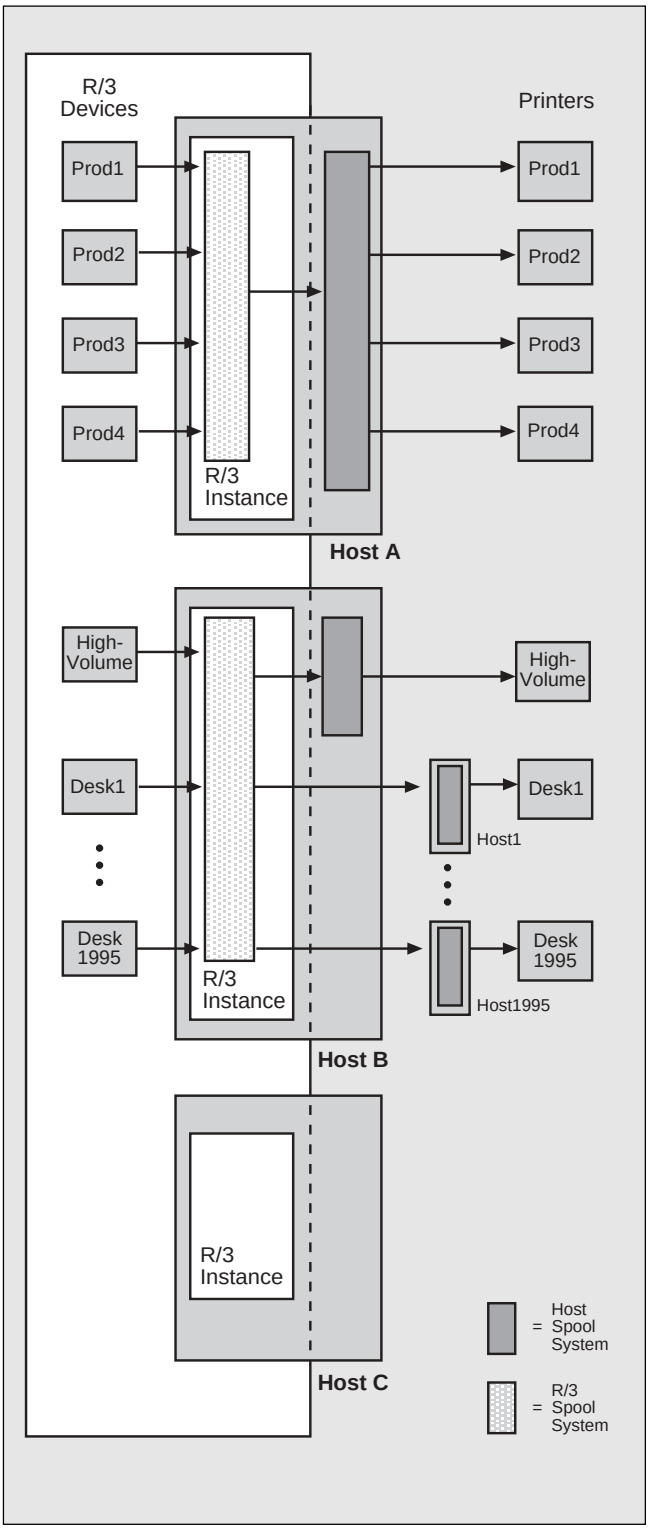
- 2,000 output devices:
  - 4 production devices
  - 1 high-volume device
  - 1,995 desktop devices
- 1 spool server (Host A) for handling production output requests
- 1 spool server (Host B) for handling high-volume output requests
- 1 application server (Host C), which is not configured as a spool server

In this environment:

- One server (Host B) handles all the high-volume print jobs, and another (Host A) handles all the production print jobs.
- The desktop devices are all assigned to the high-volume server (Host B).

Let's first examine the requirements of the production server. Here, the four production devices are always in use, but are not generating requests faster than they can be serviced by a single spool work process:

Figure 6 A Release 4.0B System with 2,000 Output Devices



- **Step 1** and **Step 2:** Neither sequential processing nor frontend printing[12] is a relevant factor here, so neither Step 1 nor Step 2 requires us to ratchet up our spool work process count. For now, the count stays at 0.

- **Step 3:** This step yields a value of 4, because all four production devices are constantly in use. Since processing is keeping pace with the rate of the devices, there's no need to advance this number beyond 4. Our tally, therefore, is 4.

- **Step 4:** Classification of spool work processes is only applicable in Release 4.6 environments. The tally, therefore, remains at 4.

- **Step 5:** Polling is used in this scenario to track print jobs released to the target printer's host spool system. So, we set aside one spool work process for this activity, and advance the tally by 1, bringing the total number of spool work processes to 5. (Only access method L is used, so connection checks are not applicable.) As before, we'll assume that five spool work processes can adequately handle management tasks and that we do not need to add another spool work process for this purpose.

- **Step 6:** No classification is used, so the total sum for our production server remains at 5.

Now we turn our attention to the high-volume printing server:

- **Step 1** and **Step 2:** Neither frontend printing nor sequential processing is a relevant factor here, so neither Step 1 nor Step 2 requires us to ratchet up our spool work process count. The count stays at 0 for now.

- **Step 3:** The desktop printers are only used on an intermittent basis. There is only one high-volume request per day (sometimes in the evening), and the desktop devices are used to print an average of only one or two pages per day. The load is so low, in fact, that for the 1,995 desktop devices and the high-volume device combined, we only need to assign two spool work processes. Our count is now 2.

- **Step 4:** Classification of spool work processes is only applicable in Release 4.6 environments. The tally, therefore, remains at 2.

- **Step 5:** Polling is used to track the print jobs that are released to the target printer's host spool system, so set aside a spool work process for this activity, and advance the tally by 1, bringing the total number of spool work processes to 3. With desktop printers in use, connection checks will also be used, so advance the tally again by 1, bringing the total to 4.

  We will assume that 4 spool work processes can adequately handle the global management tasks, and that there is no need to add another spool work process for this purpose. Since the management tasks are typically quite short, they can usually fit between two other requests. And if longer requests are being processed, it does not matter if the execution of the management tasks is a bit delayed.

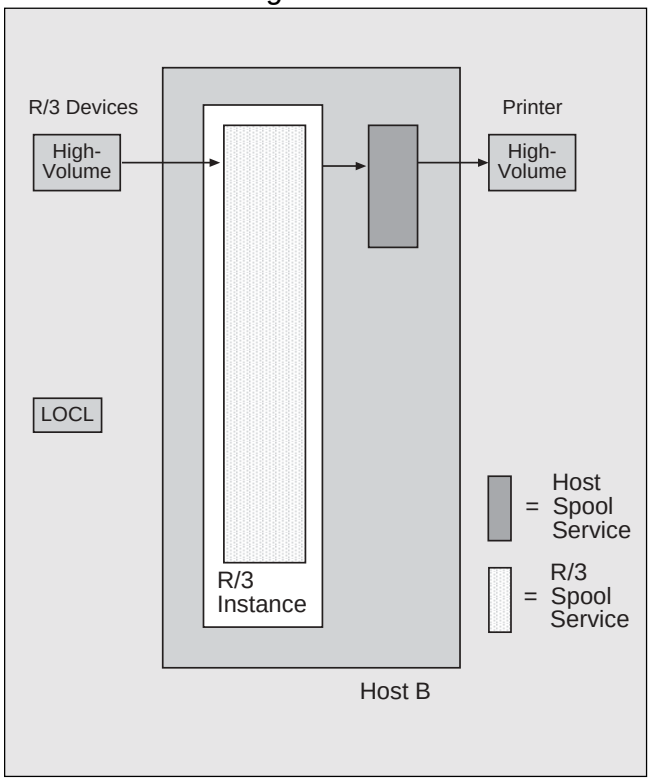
- **Step 6:** Again, no classification is used. So, the tally for the high-volume server remains at 4.

Suppose the administrator should now decide to replace the desktop printing with frontend printing. This will reduce the total number of R/3 output devices supported by the two R/3 spool servers to 6: four production devices on Host A, one high-volume device on Host B, and one frontend device without an assigned spool server, because frontend printing is still handled by the dialog service. There is no longer any need to have additional devices if the number of desktops is extended, since they all map to the LOCL device.[13] This means a drastic reduction of the configuration work.

[12] Remember, frontend printing is not supported by spool work processes until Release 4.6. This is a Release 4.0 scenario!

[13] For more information on frontend printing, see Stefan Fuchs' article, "An Introduction to SAP's New and Improved Frontend Printing," in the November/December 1999 issue of the *SAP Professional Journal*.

Figure 7 The Host B Spool Server Configuration



What happens to the spool work processes count? We can now remove the spool work processes for the desktop printing, because the frontend printing in this release will be supported by dialog processes. The count for the production printing server remains the same. When you add the LOCL output device to this server (to support frontend printing), the tally is unaffected, because in a Release 4.0B system, frontend printing is supported by the dialog service. The tally for the high-volume printing server, however, *is* affected. **(Figure 7** shows the configuration for the spool server supported in Host B.) The desktop devices are not required anymore, which means we can eliminate the two spool work processes that had been previously set aside to support this load. We can also eliminate connection checks, since they are only required by desktop print processing, and not frontend print processing. Lastly, we can do away with the dedicated "polling" spool work process — since in this scenario so few requests are being made for the high-volume printer, they can be handled on the same spool work process that handles the output requests. The end result? We need just one spool work process for the high-volume server.

Let's add another wrinkle into this scenario and assume that we're ready to take on a Release 4.6B upgrade. Now frontend printing *is* handled by the spool service. We have to assume that this will have consequences for all *three* application servers in this scenario, even Host C, which is not configured as a spool server. Frontend printing impacts all three servers because there is no way of knowing which server will be used for user logons. (A user's logon server is the one that will support his/her frontend print processing requirements.) So, we have to be prepared to support the full frontend print processing load on all three servers.[14]

Because the load for frontend printing is so low, it requires just two spool work processes. How does this requirement affect each of the three servers?

- Surprisingly, even the application server, Host C, which is not configured as a spool server, requires these additional two spool work processes!

- The production server, Host A, which had previously needed 5 spool work processes, now needs 7. Why? Well, in our original calculation for this server, the desktop devices were not an issue, because they were supported on the high-volume server. Since the production server now has to potentially support the full frontend printing load, we have to revisit Step 1 and account for the two spool work processes needed to support the frontend printing load. Adding these two spool work processes to the count brings the tally to 7.

- The high-volume server, Host B, whose spool work process count we just calculated to be 1, also requires two spool work processes for the

[14] Logon groups may offer an exception to this rule. If you are using logon groups, for example, and have one server that will never be a logon server, a spool service won't be necessary.

additional frontend print processing load. This server's tally therefore goes to 3.

You should note here that because of the two spool work processes needed to support the frontend printing load in Step 1, you must also adjust the maximum limitation for the frontend processing group![15]

If you think that using this frontend method is more taxing in terms of overall server resources because of the additional work processes, think again. The print processing load already existed; it was just being handled by the dialog service, which might have hampered the regular dialog sessions. Now you have a strict separation between print processing and dialog tasks. Perhaps, because of the missing printing load, you can now reduce the number of dialog work processes.

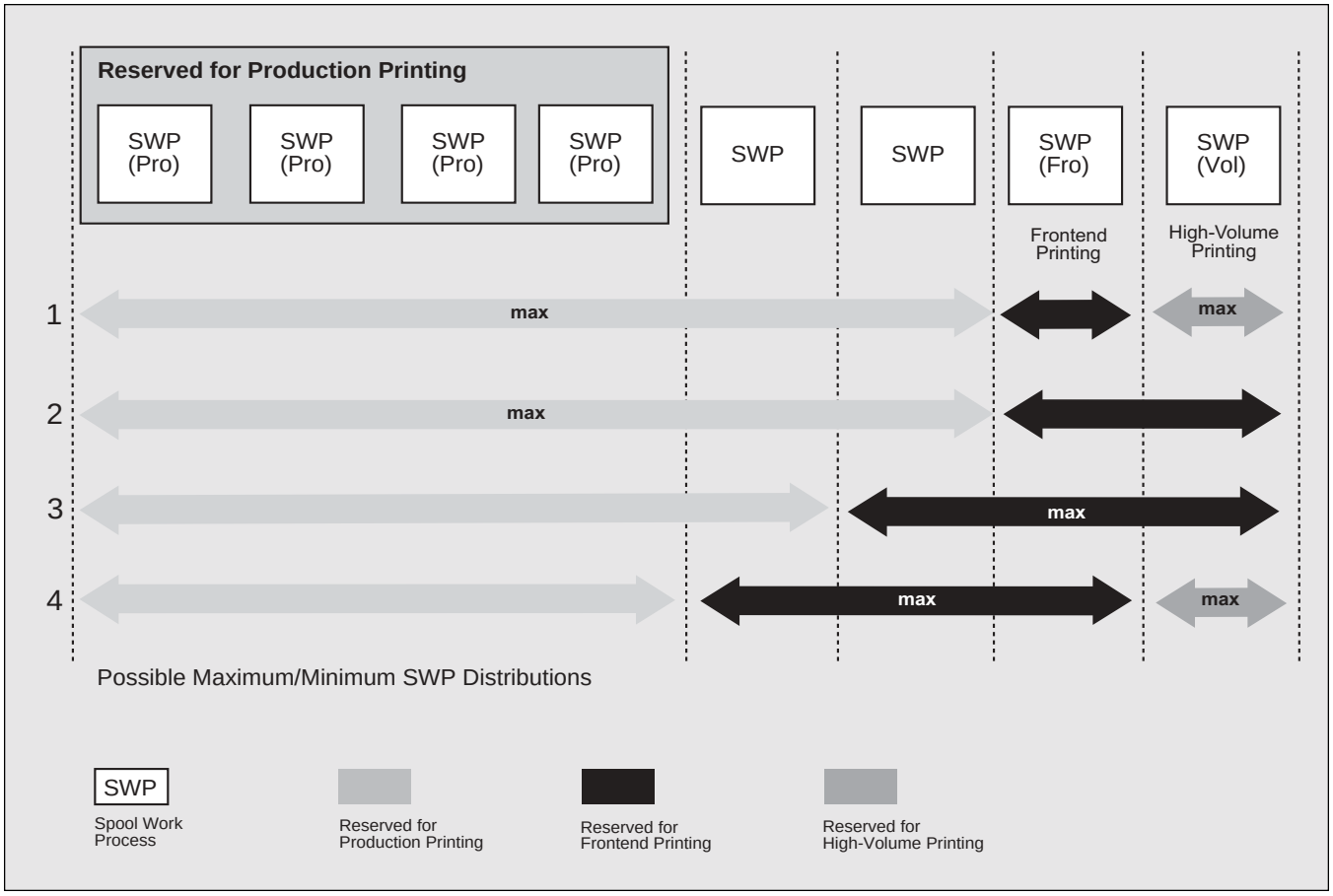
One last twist … Having successfully upgraded the server to Release 4.6B, we are now ready to leverage a spool work process classification scheme. At present, this separation exists at the server level. We now want to drive this down to the spool work process level, making it possible to migrate the spool service from the high-volume server to the production server.

- **Steps 1-3:** Since there are no sequential devices, and since we will only utilize the Release 4.6B Fro, Pro, and Vol groups, we forgo Steps 1-3 and skip ahead to Step 4.
- **Step 4:**
  - First, we look at group Fro (for frontend printing) and, based on our previous assessment, assume an amount of 2 spool work processes.
  - Second, we consider group Pro (for the production devices). We take the numbers from the old Steps 2 and 3, which gives us the value of 4 spool work processes.
  - Third, we plan to move the high-volume printer to Host A, so we have to consider group Vol. Here we take the number of the old Steps 2 and 3 from our calculation for Host B — that being the value of 1.

  This gives us a tally of 7 (2+4+1=7) spool work processes for Step 4.
- **Step 5:** We add one spool work process for polling (just as we did before) to give us a total of 8 spool work processes. (We don't need any connection checks because no remote access method is used.)
- **Step 6:** To be sure that one kind of output can never hamper another type, we now make use of the following reservation and maximum limitation feature:
  - We limit the *maximum* number of spool work processes that can be processing frontend print requests to 3. This is 1 more than we calculated in Step 4 for this group. It will be used as an extra process in the event of a short-time boost of frontend output requests. This limit prevents the frontend printing from hampering the production printing.
  - Because the production printing is our most important type of printing, we decide to reserve the 4 calculated spool work processes for production printing. This means there will always be a *minimum* of four spool work processes available for production printing.
  - To ensure that frontend print requests can always be processed, we set the *maximum* number of spool work processes that can process production output requests at 6. So, even if all six spool work processes are churning out production print requests, that still leaves two spool work processes free for frontend and high-volume printing. (Remember, we've calculated a total number of 8 spool work processes for the production server.)

[15] See Figure 10 for the profile parameters used for the classification scheme.

Figure 8 Reservation Scheme for Migrating the Spool Service from the High-Volume Server to the Production Server



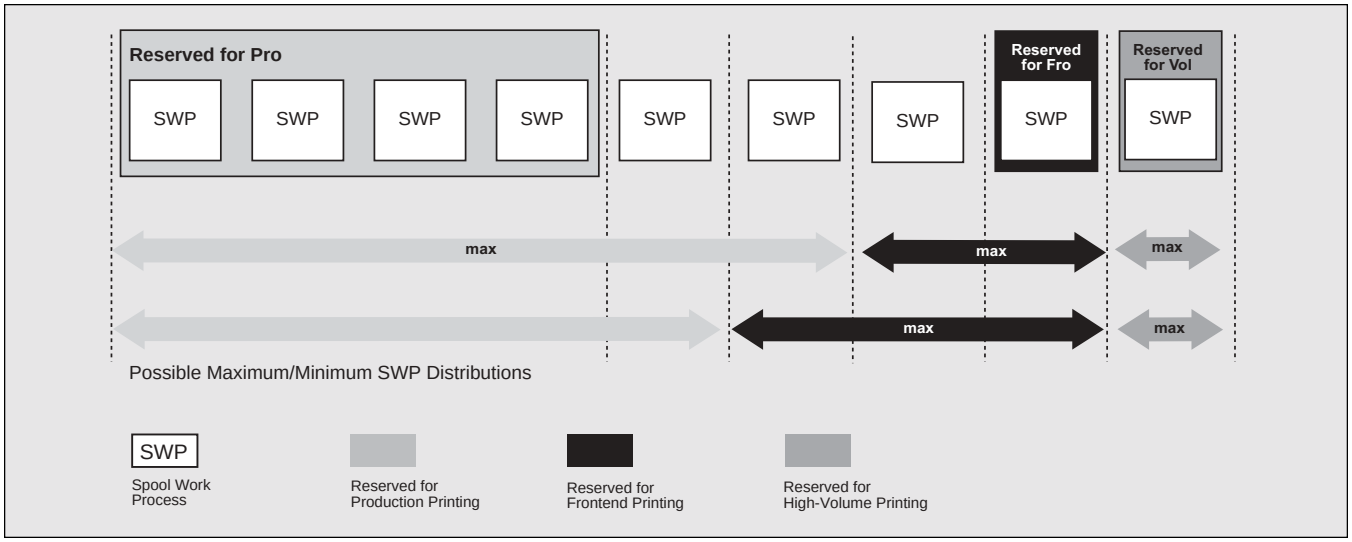
- To ensure that high-volume printing does not hamper production or frontend printing, we restrict the *maximum* number of spool work processes used for high-volume printing to 1. We do not, however, reserve a *minimum* spool work process for this purpose, because most of the time there is no high-volume printing, and we don't want to needlessly tie up a spool work process exclusively for this purpose.

We finally check the reservation sum. Because we've done reservations (e.g., set a *minimum* number of spool work processes) to be used exclusively for production printing, the sum is 4. Since 4 plus 3 equals 7 (which is less than 8), no additional work processes are required.

This gives us a final tally of 8 spool work processes. The reservation scheme is shown in **Figure 8**.

As you can see from this figure, if more than four work processes are used for production printing and the rest for frontend printing (lines 2 and 3), even if the polling task is not taken into consideration, it is possible that there won't be a free spool work process available for high-volume printing. But then, when you additionally consider the polling task, even frontend printing may be temporarily unavailable if six processes are busy with production printing, one process with high-volume printing, and the single remaining process with executing the polling task.

Figure 9 The Adjusted Reservation Scheme



One way to solve this problem is by adding three work processes. Why three? Well, if the maximum number of spool work processes is used for production (6), frontend (3), and high-volume printing (1), and you add the polling task (1), you'll get a sum of 11 (6+3+1+1), three more than before. (To fall short by even one spool work process means that there is a possibility that a task cannot be processed when it is scheduled.)

Another way to solve this problem is by using additional reservations for the other groups. We will go this route, rather than adding more spool work processes, because we want to avoid the extra resources the additional spool work processes would require. Now, the sum of the reservations is 6 (4+1+1) spool work processes — we reserve a minimum of four for production printing, one for frontend printing, and one for high-volume printing. Unfortunately, since 9 (6+3) is greater than 8 (our total number of spool work processes), this reservation scheme is not valid. So, we'll have to add at least 1 (9-8) more spool work process.

Our tally is now 9 work processess, which, in terms of server resource comsumption, is better than adding three processes without further reservations. The final result can be seen in **Figure 9**.

The sum of the maximum limitations is 10 (6+3+1), and we have 1 (10-9) spool work process that can be shared between production printing and frontend printing. Our polling process can also be used for production or frontend printing if no polling needs to be executed.

As a last example, we will leverage the new manually configurable Release 4.6 global management tasks: the spool reorganization and the activation of output requests with a starting time. We remove the batch job for the reorganization, replace it with the new spool system option, and use the start feature. We set the period for the reorg to 2 hours, and the activation period to 5 minutes (using transaction SPAD).

We know that with such a short interval for the activation period, administrative tasks will affect the processing load of this server. So, we add one more spool work process in Step 5. This brings the tally to 9. Now, if we've already added this work process to solve our reservation problem, everything is just fine. Otherwise, since we have to increase the number to 9 anyway, we could add the additional reservation for free.

Figure 10 lists the parameters you use to actually go in to set and/or override the default values that determine the number of spool work processes that will run on an R/3 spool server and that enables reservation schemes.

Figure 10 Parameters Used to Configure a Release 4.x Spool Server

Parameter	Default Number of Spool Work Processes	Description
rdisp/wp_no_spo	0	Sets the total number of spool work processes of an R/3 application server. If this number is greater than 0, output requests can be handled by the server. This parameter can be found on all Release 4.x application servers. I recommend you set this parameter to a minimum of 2 if you use a server as a spool server. If only frontend printing should be handled (Release 4.6x), there should at least be a minimum of 1 for all servers.
rdisp/wp_no_spo_**<GRP>**_max Dsk Fro Pro Reg Tst Vol *(Only applicable to Release 4.6 servers)*	1 (for Fro)	Sets the maximum number of spool work processes that can be used to process a particular class of print jobs. By default, the maximum number of processes that can be used for frontend printing (Fro) is 1, but this can be changed. The maximum default for all other groups (Pro, Vol, Dsk, Tst, and Reg) is the value of rdisp/wp_no_spo.
rdisp/wp_no_spo_**<GRP>**_min Dsk Fro Pro Reg Tst Vol *(Only applicable to Release 4.6 servers)*	0	Reserves a minimum number of work processes exclusively for the processing of jobs for a particular group. These spool work processes will not be used to process other tasks.

Monitoring and Fine-Tuning Spool Server Performance

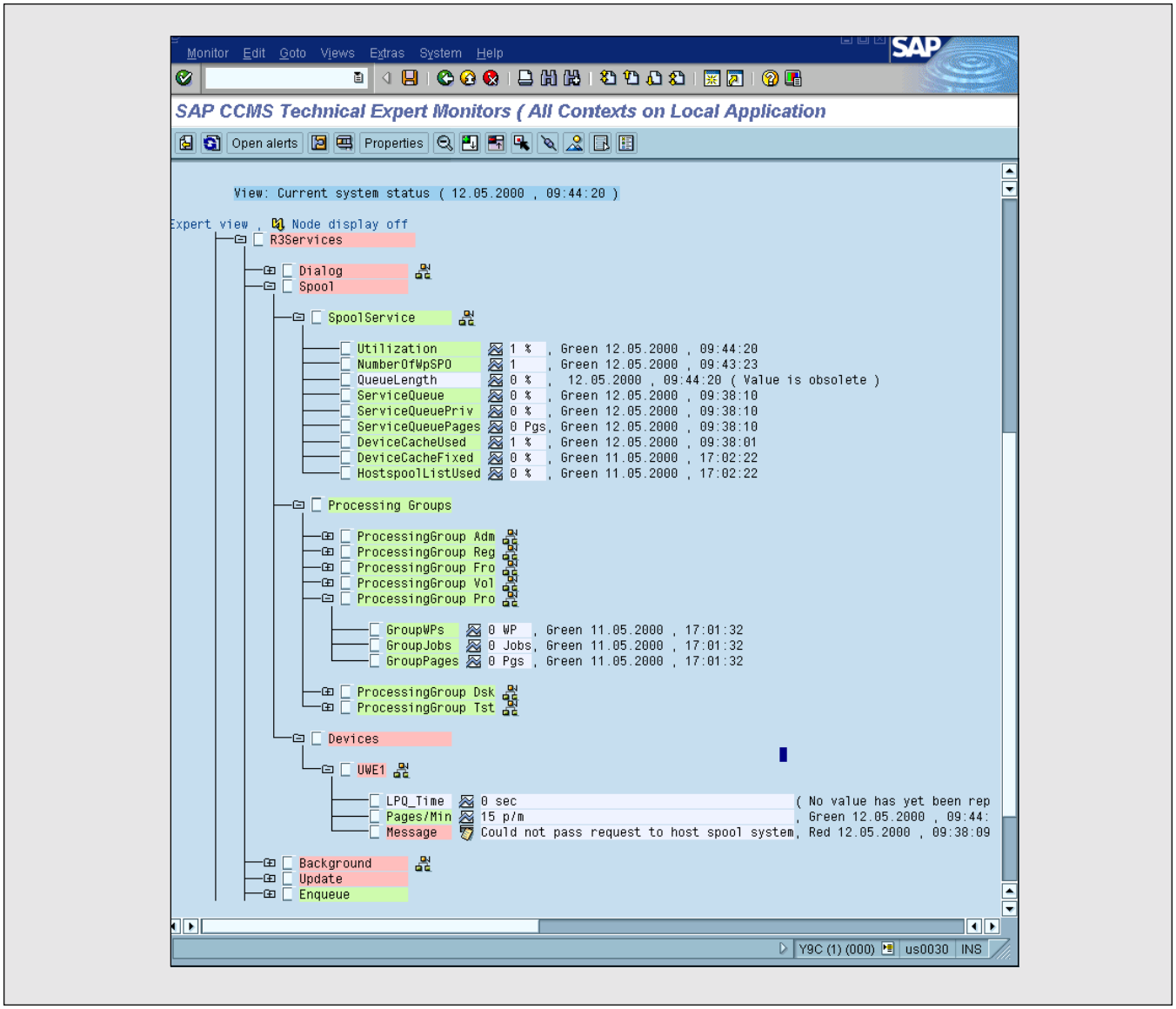
One thing that every good administrator knows is that change is inevitable. Whatever configuration you arrive at today will need to be fine-tuned tomorrow. The number of spool work processes you derive provides only a best guess. To make certain that this "guess" is holding up to your environment's real-world requirements, I strongly recommend that you monitor the behavior of your initial configuration.

Many of you may be familiar with transaction SP01, which, in addition to enabling end users to track the status of their active jobs, can be used by administrators to see the *status* of users' jobs on a system-wide level. You can also use transaction SPOV (via the overview button in transaction SPAD). These transactions, however, will not help you monitor the *performance* of your spool configuration.

For performance information, turn to transaction RZ20.[16] It not only provides snapshot information, but historical data as well. You can dynamically select the performance information you are interested in and configure your own monitoring screens. You can even configure screens that let you monitor the performance of multiple R/3 systems.

[16] Starting with Release 4.0A, the spool system is connected to the SAP monitoring architecture.

Figure 11 Tree Elements for Monitoring the Spool System



In **Figure 11**, you see performance information for a spool server that I configured with one spool work process. (You will find this spool system information under node `R3Services-Spool` for all R/3 application servers.) The first entry you see under "SpoolService" is "Utilization,"[17] a value that summarizes the average load of all spool work processes.

[17] "Utilization" is the only entry I discuss in this article.

The average utilization should never be higher than 80 or 90 percent. If it is too large, this is a sure sign that there are too many requests for the spool service.

The average load across the single spool work process in my test system is just 1 percent. The reason for this is that there is virtually no printing. If this is a typical value in your system, it is a good bet that you've got too many spool work processes.

(But remember, never remove the last spool work process, even if its utilization is quite low!)

Because Figure 11 is a screen capture of a Release 4.6B screen, you see the additional node `Processing Groups`, where you can find information about the spool server's load according to its spool work process classification scheme. The `Reg` processing group summarizes pending jobs for unclassified devices and therefore also for spool work processes assigned to this group. `Fro` summarizes jobs for the spool work processes that constitute the frontend printing group. `Vol` summarizes jobs for the spool work processes that constitute the high-volume printing group. `Pro` summarizes jobs for the spool work processes that constitute the production printing group.

For each group, you can see information about the number of used spool work processes (GroupWPs), the number of pending jobs (GroupJobs), and pages[18] (GroupPages). If the number of pending jobs gets quite large, or if the number of spool work processes in use is near the maximum configured for this group, it's a sure sign that there is not enough processing power and that you need to add a new spool work process to this group. If the total load for the server (i.e., its "Utilization" value) is less than 70 to 80 percent, you can simply adjust the maximum value for this group. If the total load for the server is approaching 100 percent, an additional new spool work process might be in order.

What do you think it means if the value you see in GroupWPs never approaches the maximum that has been set for this group? For example, what if the server has been set up with a maximum of 4 spool work processes for production printing, but the value in GroupWPs never exceeds 2? It suggests that there might be too many work processes for this group.

[18] "Pages" refers to the sum of the pages in all pending output requests.

Does this mean you need to decrease that maximum value? Not necessarily. If the maximum limitations of your reservation scheme allows for it, the under-utilized production spool work processes will be used for the other groups automatically. You don't need to take any action. If, however, your reservation scheme sets things up in a way that *only* allows production print jobs to be processed by these spool work processes, you should decrease the number of reserved processes.

If the total load is quite small all the time (i.e., less than 10 or 20 percent), it might be possible to reduce the total number of spool work processes. Just be sure that you've got enough horsepower to cover your peak periods of print processing activity. Let's assume you have a system where a large number of users all want to print out jobs around 5:00 in the afternoon, right before they leave for the day. In this situation, it's quite possible that the average load (its utilization) throughout the day would be very small, but the important peak performance utilization would be very high. So, when reviewing your server's utilization values for the purpose of determining an optimal number of spool work processes, you need to take into account the behavior of your end users!

If the total load is quite small all the time (i.e., less than 10 or 20 percent), it might be possible to reduce the total number of spool work processes. Just be sure that you've got enough horsepower to cover your peak periods of print processing activity.

Management Tasks Take a Toll on Performance, Too!

Up till now, I've been discussing performance as it relates to the processing of print jobs. But we can't forget about management tasks. They impact a spool server's performance, too. Take a look now at

Figure 12 Status Overview of the Spool System

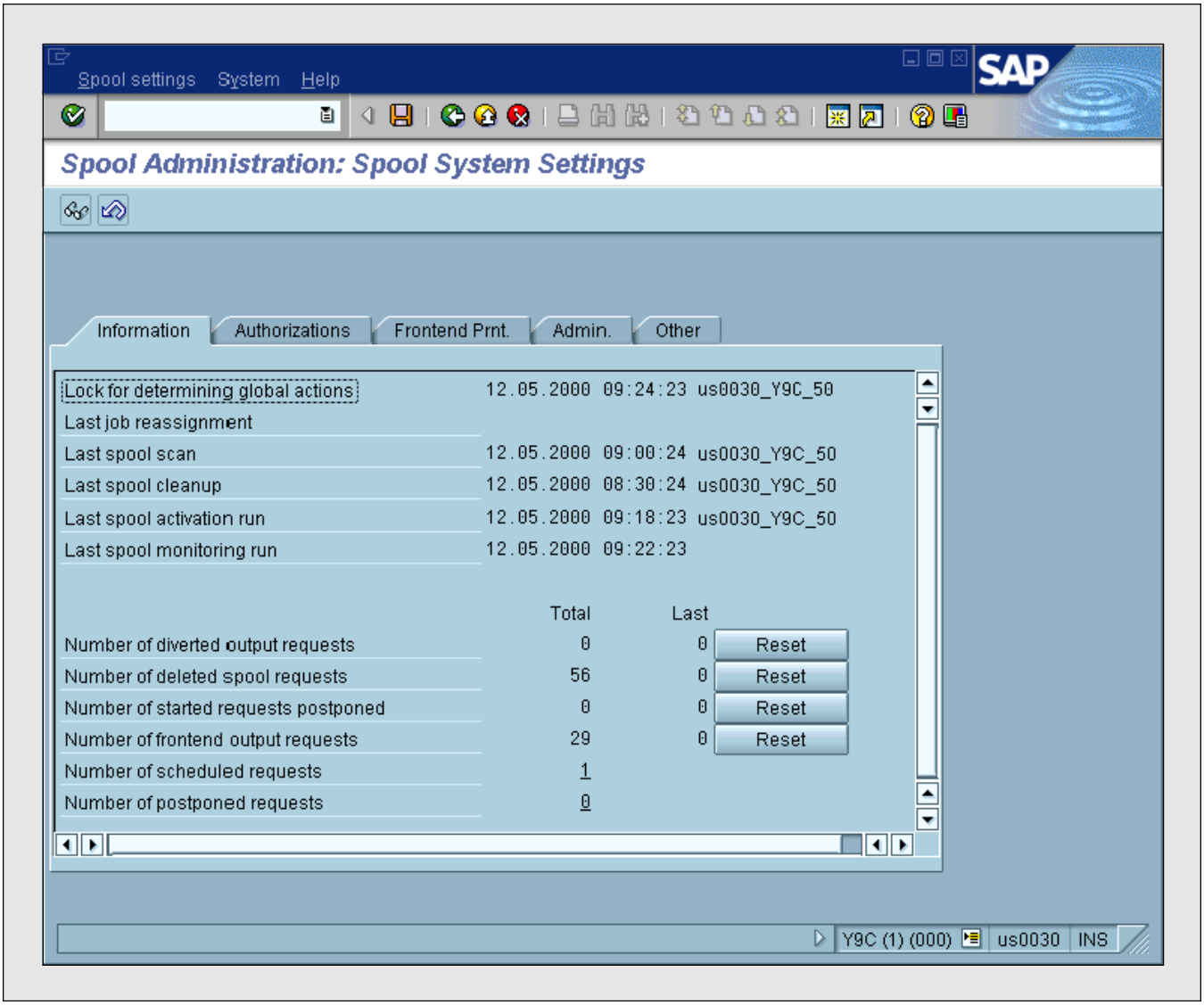


Figure 12. Here, you find status information about administrative tasks. You can see the last execution time, the server(s) of the different global administrative tasks, and the number of jobs affected. If a line is highlighted, then the last execution took place too long ago, a signal that there is a problem with scheduling the global administrative tasks.

On the Admin screen (shown in **Figure 13**), it is possible to select some global spool options, such as activation of automatic spool reorganization and activation of jobs with a starting time together with the desired intervals. The default period for the reorg is 24 hours (in Figure 13 we have changed the default to every hour); and for the job activations, every 5 minutes.

Figure 14 summarizes the queue and cache parameters that affect spool server performance.

Figure 15 summarizes the server options that affect spool server performance.

Figure 13 Configuring Optional Global Management Tasks

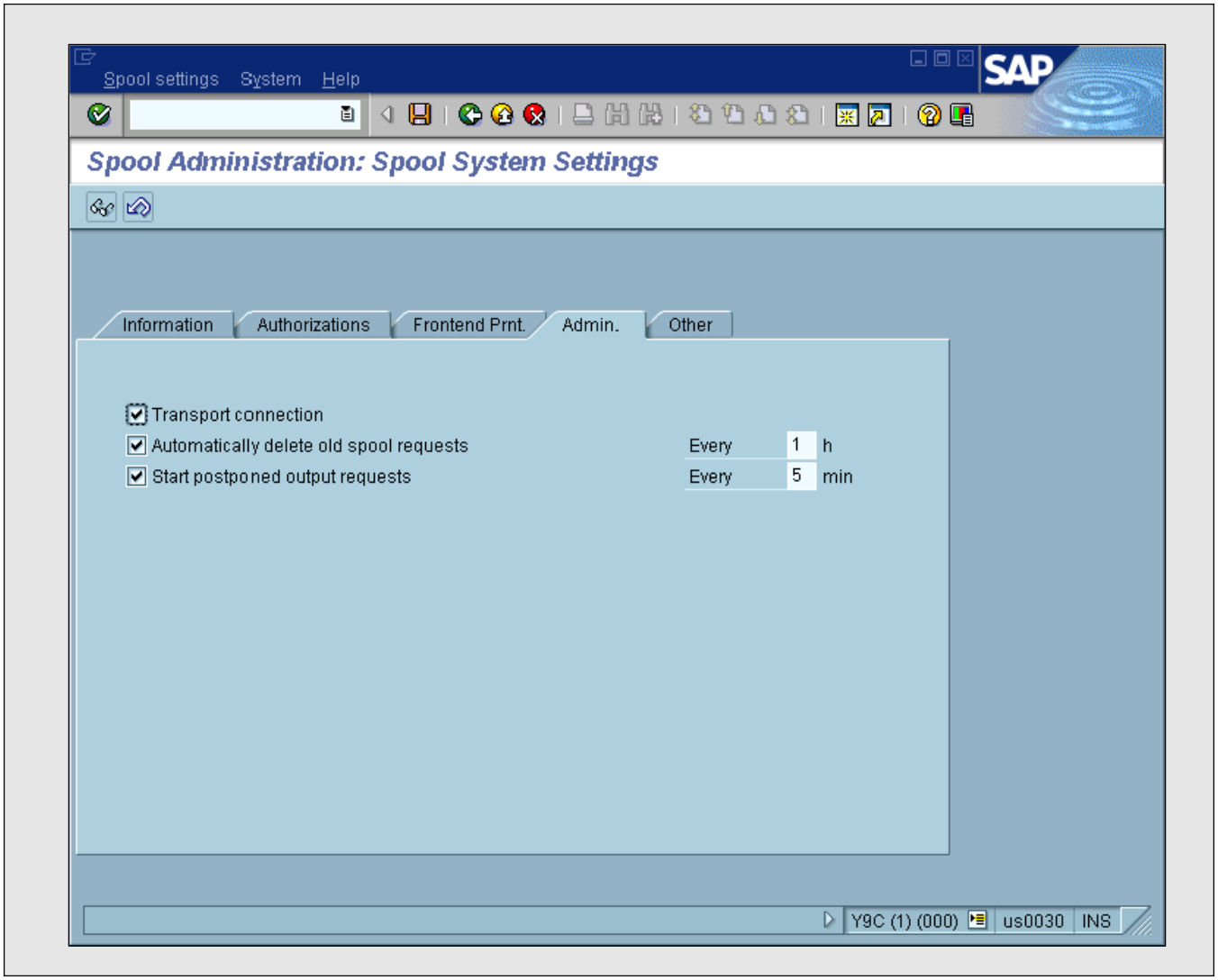


Figure 14 Queue and Cache Parameters That Affect Spool Server Performance

Parameter	Default Value	Description
rdisp/max_com_msg	50	Relevant for all servers where output requests are generated. It limits the number of print messages that can be handled between two commits. If a transaction or report generates more output requests without a COMMIT WORK, downstream jobs will be stored in the database, but will not be sent to a spool service. This means that the first few jobs will be printed at once, but the rest will lag by 20-30 minutes (after their generation), until the time when they are found by the global job scan that is done as a periodic global administrative task.

(continued on next page)

Figure 14 (continued)

Parameter	Default Value	Description
rspo/global_shm/job_list	50	The spool service queue, or "job list," is used to queue jobs that need to be processed. The queue absorbs short-term overloads, output requests that are created faster than they can be processed. The queue should therefore be large enough to handle typical short-term (temporary) overload situations in the R/3 system. If there is an overload for a longer period and the spool service queue does not have sufficient entries, the requests are stored in the dispatcher queue. However, since the size of the dispatcher queue is also limited, there cannot be an unrestricted number of requests kept in the queue. If there is a general (not temporary) overload situation on a spool server, enlarging these two queues is not the answer to the problem. This is useful only if temporarily more output requests were created than can be processed, and if there is enough time for processing between such peaks. For a general overload situation, an improvement of the throughput could be possible by reconfiguring the R/3 system in the following manner: • Increasing the number of spool work processes for the spool server affected • Better distribution of output devices to spool servers • Increasing the number of spool servers • Distributing requests among several spool servers • Too many requests for a sequential output device
rspo/global_shm/printer_list	150	The device cache, or "printer list," was introduced prior to Release 4.x. In earlier releases, it was only used for the spool work process and had to contain all the output devices the spool server was responsible for. Only requests for those output devices could be processed that could be stored in this buffer. If the cache was too small, output devices could not be served. As of Release 4.x, the device cache has a new role. It serves as a cache for device definitions and server assignments for all work processes (and also for other servers than spool servers). Entries are recorded in the cache when they are needed. However, they can be pushed out if there are no free entries left. For this usage, the cache size does not depend on the number of configured output devices. For spool servers, there is an additional usage for the cache. In order to do specific querying at the host spool level, the spool service has to register for which output devices requests exist in the host spool system that have not been reported as processed. Also, it has to register for what devices connection problems occur. For these purposes, entries in the device cache are used. However, these entries will be set and pinned until all the requests of a device have been reported as finished and no further connection problem has occurred. The number of entries in the cache must be at least as large as the number of devices that are maximally used at the same time. Even if a spool service is potentially responsible

(continued on next page)

Figure 14 (continued)

Parameter	Default Value	Description
		for a very large number of devices, a smaller cache can be used because this number of devices will never be used at the same time (solely for performance reasons). If the cache is too small, requests of other devices cannot be queried at the host spool system as long as there are still old devices pinned in the cache. However, no errors will occur, and instead, these requests are automatically reported as finished after they have been successfully transferred to the host spool system. No explicit querying is necessary anymore. This emergency operation continues until entries in the cache can be displaced. The chosen value determines the maximum number of cache entries that can be pinned. Since the device cache is also used by other work processes while output devices are accessed when spool requests are created, not all entries can be pinned so that the spool service can still operate smoothly. That is why the minimum value is added to the selected size. However, in order to maintain the cache functionality, these additional entries cannot be pinned. For example, if you select a value of 200, you will get a cache with a total of 350 entries (200 fixable entries and 150 additional entries).
rspo/global_shm/hostspool_list	10 times the size of the service queue (minimum 500)	Every spool work process can transfer requests to a host spool system and is able to query the status of requests. The list of those requests that are to be queried, but not yet reported as finished, has to be kept in shared memory in order to reduce database access on status information. The host spool cache is used for this purpose. The number of requests in the host spool system that can be administered by the spool service is limited by the size of the host spool list. The status of other requests that are transferred to a host spool system cannot be queried. These requests are reported as finished after they have been successfully transferred to the host spool system within R/3 so that no entry in the host spool cache is needed.
rspo/global_shm/server_list	100	To map server names to active servers efficiently, a server cache is used. This minimizes selects on the database. It stores relations between individual entries according to the server mapping and the current server configuration of the R/3 system. If there are no free entries left, cache entries are added to the shared memory. The cache size is configured according to the configuration in the R/3 system. The maximum cache size is limited by the size of the underlying shared memory area.
rspo/global_shm/memory	100000	A shared memory area used for server list entries and security parameters for SAPlpd devices.

Figure 15 Server Options That Impact Spool Server Performance

Parameter	Default Value	Description
rspo/local_print/server	Not set	Available as of Release 4.6A. The value of this parameter specifies the server that should be used to handle frontend output requests. If it is empty or not set by default, the local server will be used if it has a spool service. Otherwise, the requests are routed to an arbitrary spool server (with the lowest load).
rspo/rspoget2/rescantime	30	Default time between two global job scans, if not overridden by the spool system setting in transaction SPAD. This parameter should be set in the default profile and not in an instance profile.
rspo/host_spool/checking	1	Available as of Release 4.0B. A value of 0 switches the behavior of connection checks back to its earlier behavior for access methods S and U. By default the new host spool connection check *is* enabled. It is possible now to use unreliable remote print servers. If a job cannot be passed to a remote host spool system, the job and all other jobs to that host spool system are delayed without further connection retries for all jobs. Earlier there were separate processing retries for all jobs. After some delay, the jobs were reprocessed. If the connection still failed, they were delayed again. Now there will be no time-consuming reprocessing for delayed jobs until the host spool system is reachable again. Instead of multiple connection retries, there is a coordinated connection check that will automatically be paused if the down-time is too long. It will automatically be resumed if a new job comes in for this host spool system. If no further job is sent after turning on the remote system again, at least one of the devices has to be reactivated manually using transaction SPAD. But remember, to determine a host spool system failure, at least one job has to be processed, including a connection try and the TCP time-out.
rspo/rspoget2/check_retries	3	Establishes how many retries should be done to check the availability of a remote host spool system after a connection problem.
rspo/tcp/retries	3	Establishes how many retries should be done for job processing if a host spool system is not available. If the checking mode is switched off, all jobs enter the error state if no connection is possible after these retries. If the checking mode is switched on, they stay in a delayed state until a new job for this host spool system is generated or the checking is restarted manually by transaction SPAD.
rspo/tcp/retrytime	5	The time, in minutes, between two checks.
rdisp/spooltime	60	Period for periodic messages for the spool service administration tasks.
rspo/rspoget2/infomessages	1	Available as of Release 4.6A. Send broadcast messages to propagate spool server load for load balancing. If set to 0, instead of sending messages, a database table will be used. If you really want to change the default, do this in the default profile, not in the instance profiles.

Tips for Bolstering Spool Server Performance

✔ Take advantage of the spool work process classification scheme introduced in Release 4.6B. This new capability lets you simulate multiple spool servers for different classes with just one server.

✔ Even if you have enough servers to separate the different output classes at the server level, the spool work process classification feature can still be very helpful. You can use it in conjunction with your alternate server configuration to help achieve a more reliable output environment. Alternate servers operate as follows: if a server is unavailable, print jobs get redirected to the alternate server. But what happens, for example, if you use the production server as an alternate server for your high-volume printing server? When that high-volume server fails, its print jobs get redirected to the production print server, and those high-volume print jobs will start to hamper your production output. This can be avoided if you use device classification together with a maximum restriction for the class that will be handled by a server that's acting as an alternative to another one. In this example, you could restrict the processing of high-volume requests on your production server to one spool work process. This might hamper the processing of *high-volume* requests in the event the high-volume server fails, but your production output will remain intact.

✔ Don't be too quick to configure an output device for sequential request processing. Exercise this option only when and where it is truly needed. The whole idea behind this concept is to inhibit parallel processing. Once you do that, you impede performance.

✔ If there is a need to configure an output device for sequential processing, consider defining two R/3 output devices for the same physical printer — one for sequential processing, and the other one for parallel processing. In this way, only the applications that really need this kind of processing have to use these particular R/3 devices.

✔ Additional applications that require a sequential stream of output should each use a separate R/3 device for sequential processing. Through this setup, you achieve parallel processing of requests generated by different applications. (Other users and applications can still use those devices.)

✔ If you have a limited number of high-volume print requests, you don't need to set up an additional spool server to process them or even set up a high-volume processing group on your spool server (if it's a Release 4.6B server). Think back to Scenario 1, where requests for output devices with an enabled sequential processing option are handled by only one work process. So, if you add one spool work process to your server, it can handle an additional high-volume printer without hampering your production output. Because an extra spool server requires at least three work processes (two dialogs, one spool), it is cheaper (in terms of resource consumption) to use three additional spool work processes for up to three high-volume printers. This has the added advantage that these processes can be used for other output requests as well, provided no high-volume request is in need of processing.

✔ The spool system processes print requests according to their priorities. High-priority requests always take precedence over lower-priority requests. You can use priorities higher than 3 (or 5, as of Release 4.6A) for important output in your production environment. But take care when assigning higher priorities to large requests, because there is no displacement of output requests. Once a processor starts processing a request, it won't switch gears, even for a higher priority request, until processing of the request it has started on is complete. So if there is a large request that has the attention of the processor, it may still hog the spool service of an R/3 spool server.

✔ By default, frontend print requests will be handled by the spool service of the server where the user is logged on. If this server does not have a spool service, another server (with a spool service) will be used. To avoid routing print requests to other servers, where the additional load may start to present problems, you should have at least one spool work process

configured on all your Release 4.6B servers, even if they are not used for traditional printing (i.e., they are meant to support only dialog processes).

✔ An alternative to configuring all your application servers with at least one spool work process is to select a dedicated frontend request server by adding the profile parameter `rspo/local_print/server` to your instance profile. All frontend requests created on this particular instance would then be redirected to the spool server specified by the parameter value.[19] If you do this, be sure there are enough spool work processes on that specified server! If you are using extensive frontend printing, you should increase the number of spool work processes *and* the maximum restriction for processing group `Fro`.

✔ Increasing the number of spool work processes and the maximum restriction for processing group `Fro` is good for any server that supports users who generate extensive frontend printing demands.

✔ If you don't need to track jobs once they've cut over to the host spool, switch off the polling feature in your device configurations. Note that only access method E provides reliable feedback.

✔ You can save the processing overhead associated with polling if you can use a certified output management system supporting the callback feature.

✔ The host spool checking mode that was introduced in Release 4.5B works like this[20]: if a host spool system cannot be reached, all jobs for that system are delayed without further processing until a check of that host spool system shows that it is reachable again. At this point, the jobs will be printed without the need for manual intervention. Connection checks can be very time consuming because checking the availability of unavailable host spool systems requires long timeouts. To save time, use as few different remote host spool systems as possible.

✔ The spool reorganization should be done periodically. You could use a batch job (RSPO0041 or RSPO1041) for this task, but you would have to bother with batch jobs, variants, and batch servers, and you would have to ensure the job really *runs* in your batch environment (i.e., it is not merely planned). As of Release 4.6A, just switch on the reorganization checkbox in the spool system settings. This enforces the spool expiration date of spool requests, as long as there is at least one application server with a spool service in your R/3 system.

Conclusion

The availability of multiple spool work processes per R/3 server in Release 4.x solves many of the urgent performance problems of earlier releases. You no longer have to add additional R/3 servers just to get more spool work processes. Even in small R/3 systems, it is now possible to handle very high-intensity printing loads simply by adding more spool work processes. Another huge improvement in Release 4.6 is the introduction of the new frontend printing and dynamic work process grouping. Together with the alternate server concept (which I described in my last article), it is now possible to configure a fault tolerant, highly scalable output environment.

The design for handling multiple spool work processes provides a self-organizing mechanism for assigning jobs to processing groups and spool work processes. An administrator does not need to become involved in the details of parallel processing — the administrator just specifies limitations and possibilities; the rest is done automatically. Still, there is ample room for misconfiguration. For example, since all available resources must be configured by the system administrator, it is still a demanding task for every administrator to achieve a configuration that is both adequate and does not waste resources. I hope this article has given you the insights and rules you need to achieve this goal.

---

[19] Again, for more detail on failover, load balancing, and redirection of print jobs from one spool server to another, see my last article, "Achieving a More Manageable and Reliable R/3 Spool Server Landscape Using Release 4 Output Classifications, Logical Servers, and Alternate Servers," *SAP Professional Journal*, November/December 1999.

[20] There is also a patch for Release 4.0B that provides this functionality.

