The purpose of this document is to help in troubleshooting job issues.

I. General

When troubleshooting job issues the following flow of sessions/questions/actions might be helpful in finding the cause of the problem:

- Review "Print Job History" (ttaad5411m000)
- Review "Print Job History Messages" (ttaad5412m000)
- Review the $BSE/log files (especially log.bshell)

- Does the problem occur in 1 company or in all companies?
- Does it concern a specific job or all jobs?
- Does it concern a specific user or all users?
- Split the job concerned into different new jobs and check the behaviour of those newly created jobs, i.e. if you have 3 sessions in the job, create 3 new jobs each containing one of the sessions concerned
- If the splitted jobs all run properly:
  Check whether there could have been a conflicting situation when the different session ran together in the job.
- If 1 of the splitted jobs does not run properly, you know the session that is causing the job to hang.
  - Does the specific session run properly outside the job?
  - Check whether the job always stops at the same point in time i.e. in a subsession: this can be be determined with the bshcmd executable (more info attached)

If the specific session (subsession) has been traced, further analysis is needed to find out what the reason is that the job was not ended properly for this specific session.

Notes:

- Also review section IV of this document on the use of bshcmd.
- In "Maintain sessions by job" (ttaad5101s000) you can set the "Action on error" field to "Interrupt", "Continue" or "Restart Session".
  Consider using the "Continue" option when a specific session is often hanging within a job, thus preventing that other sessions within the same job are not being executed (mind the implications).
- If you set the "max duration" of a job, please consider the consequences:
  if the max duration has been reached, the session that was running at the reached max duration time will be completed, however the sessions within that job that were not started yet will not be started anymore

II. Jobdaemon issues

A separate jobdaemon needs to be started per company:

This can be done by creating a rc.startjobdm.xxx script in $BSE/etc as a copy of rc.startjobdm, and adapted for the USER part. UNIX user root should start this script. (xxx specifies the company)

Example of edited script part in rc.startjobdm.000:
```
# If you want to use several companies you can create several users for e.g.
# root who probably is running this script. If so, copy this script and use
# the next line to define the USER variable (<root100> is just an example).
#USER=<root100>         ;export USER
USER=mfg000;export USER
```

Requirements for the baanuser that starts the jobdaemon:

The baanuser (mfg000 in above mentioned example):
- should be created (session "Maintain User Data" (ttaad2100m000))
- should be a "Super user" and connected to "System Login: root"
- have the applicable developer authorizations (session "Maintain General Developer Authorizations" (ttadv0142m000))
- be assigned to a current package VRC (session "Change current package VRC of User" (ttadv0140m000))

Is jobdaemon running?

You can check at UNIX level whether or not a jobdaemon is running via:
```
  ps -ef|grep aad5206
```
With this command you can also see under which bshell process the jobs will be executed.

Example of output:
```
# ps -ef|grep aad5206
  root 10894 13278   0 14:53:52  pts/1  0:00 grep aad5206
  root 16414         1   0 14:53:38  pts/1  0:00 ba6.1 ttaad5206m000
  root 18574 23106   0 14:52:21      -   0:00 bshell (mfg752[mariag]:23106/PIP
  root 21214 16414   0 14:53:38      -   0:00 bshell (mfg000[mariag]:16414/PIP
  root 23106         1   0 14:52:21  pts/1  0:00 ba6.1 ttaad5206m000
```
This means that 2 jobdaemons are running under 2 different bshell processes.

Stop the jobdaemon:

Every rc.startjobdm.xxx needs its own rc.stopjobdm.xxx to stop the jobdaemon for the specific company: then all processes concerned will be stopped properly and the jobdaemon can be restarted again.
The rc.stopjobdm.xxx script can be created in $BSE/etc as a copy of rc.stopjobdm, and adapted for the USER part (similar as in rc.startjobdm.xxx).
UNIX user root should start this script.

Example of edited script part in rc.stopjobdm.000:
If you want to use several companies you can create several users for e.g.
root who probably is running this script. If so, copy this script and use
the next line to define the USER variable (<root100> is just an example).
#USER=<root100> ;export USER
USER=mfg000;export USER

If jobdaemon cannot be stopped via rc.stopjobdm.xxx:

- Kill the bshell process concerned.
- Remove the application lock on the jobdaemon (session "Remove Application Locks" (ttstpdellocks), else the jobdaemon cannot be restarted properly anymore.

Example:
kill -15 18574 (PID of bshell run by user mfg752 -see above-)
ps -ef|grep aad5206
  root 16414 1 0 14:53:38 pts/1 0:00 ba6.1 ttaad5206m000
  root 21214 16414 0 14:53:38 - 0:00 bshell (mfg000[mariag]:16414/PIP
  root 23110 13278 0 15:11:53 pts/1 0:00 grep aad5206

Note:
Mind that with this "kill - 15" of the bshell process, also the related processes (i.e. ba6.1 and the database driver process) are closed properly (this can take some time).

1 jobdaemon can execute different jobs at the same time:

Every 60 seconds the jobdaemon reviews the jobs that need to be executed (status: in queue) at a certain time e.g. 10:00, and starts the applicable processes.
When the number of jobs to be executed at 10:00 could not all be started within the 60 seconds, the remaining jobs are started the next time the daemon is checking again.

Jobs are not executed by the jobdaemon:

- If a job that is supposed to be run via the jobdaemon has a status like "Runtime Error", the job will not be started. The status needs to be reset to "In queue" again via "Maintain Job Data" (ttaad5100m000)
- When 1 of the running jobs is completely blocking the bshell, under which all jobs are supposed to run, the jobdaemon does not start any job anymore. Try to locate the reason that is causing the blocking of the bshell via the above mentioned troubleshooting flow.

  To restart the jobdaemon:
  - First stop the jobdaemon: execute the rc.stopjobdm.xxx script
    If this doesn't work properly, use "kill -15 <bshell PID>"
    If that doesn't work properly, the only way is to use "kill -9" for all processes Concerned.
    Don't forget to remove the application lock on the jobdaemon when you had to use the kill command.
  - Restart the jobdaemon: execute the rc.startjobdm.xxx script
    Check with “ps -ef|grep aad5206” whether the daemon runs and stays running.

III. Issues on tools sessions in a job

- Tools sessions can only be added to jobs in company 000.

- Queries in a job:
  - Create a query with session ttadv3180m000 Maintain query data.
    Take care that you add a condition for ._compnr, else the job will execute the query in company 000.
    Example of query:

```
select   tccom000.ncmp
from    tccom000
where  tccom000._compnr = 001 <==
```

    Generate the report and check if the query is executed properly.
  - Execute: Make job.
    At the question 'Consider the difference between entered date and system date?' you can answer either Yes or No. This is dependent on whether you would like to work with a relative date or an absolute date.
    Add the session to a job of company 000 (you can either choose an existing job or create a new one).
  - Activate the job (in company 000) or have the job executed via the jobdaemon of company 000

  Note: if you get the error "Report Not Present" please refer to solution 106822

IV. Use of bshcmd

The bshcmd tool can be used to review the processes running within the bshell.
This can be handy to check whether there is still bshell activity or not, and if not in which process the bshell activity stopped.
It also can be used to kill processes within the bshell, or to kill the complete bshell.

Usage:

bshcmd6.1 [options] <bshell_pid>

options:

-v	: Print version
-p	: Show process list
-m	: Show memory usage
-d <dbglvl>	: Set DEBUG_LEVEL to (octal) <dbglvl>
-k <pid>	: Kill bshell process id <pid>
-e	: Kill all bshell processes
-M "message"	: Send "message" to bshell
-W <sec>	: Wait until the previous issued command is executed After this the previous command will be overwritten
-w <sec>	: Wait <sec> seconds for bshell to execute command
-u <sec>	: Send SIGUSR1 to the bshell (wakeup). Only to be used in combination with -w option. Waits <sec> seconds to see if the command is executed

-s : Show entire contents of logfile (if accessible)
-l : Print logfile name of bshell (for later examination)
-T "cmdstr" : Modify BDB_DEBUG or TT_SQL_TRACE (bshell) and DBSLOG, TT_SQL_TRACE, {DBMS}PROF or {DBMS}STAT (drivers) tracing Variables
"cmdstr" may contain multiple commands of the form:
<trace variable>=<value>: set variable to value
<trace variable>+<value>: add bits to variable
<trace variable>-<value>: remove bits from variable

Remarks:
- Output will be stored in $BSE_TMP/bshell.<pid> (default)
- Only one command can be active (new commands will overwrite previous ones)
- Check the return value to see if a command has been processed (use the -w option)

Example of usage:

The following example is meant to provoke a job to "hang".
Create periodical job in company 000. Add session “Print Report Defaults” (ttaad3405m000) to job and direct it to a Windows Printer device.
Put the job to 'In queue' and check whether jobdaemon is running for company 000.

→ At the applicable "Next execution time" the job status changes from "In queue" to "running" and it stays in status "running".
Session "Print History" (ttaad5411m000) and "Print History Messages" (ttaad5412m000) do not have any data yet, for the jobstatus is still "running". Also the log.bshell does not give any message yet.

Now you want to check the activity of the bshell concerned:

- Check what the <bshell_pid> for the jobdaemon is:
  # ps -ef|grep aad5206
  root 15134 14634 1 12:17:14 pts/1 0:00 grep aad5206
  root 17944 1 0 11:10:31 pts/1 0:00 ba6.1 ttaad5206m000
  root 18300 17944 0 11:10:31 - 0:03 bshell (mfg000[mariag]:17944/PIP
  → <bshell_pid> is 18300

- Execute the following:
  bshcmd6.1 -s -p -w 10 18300
  → if output is generated, it will be written to ${BSE}/tmp/bshell.18300

- Review the output.
  # more ${BSE}/tmp/bshell.18300
  Command 1 (arg ") issued by user root

PID	PPID	PGRP	SESSION	TICKS	FLAGS	S	CMP	MEM
1	0	1	ttaad5206m000	6299	00000010	S	000	213544

  object: ttaad5206m000
  oic:

```
   ottaad5206
   ottstp_stddll
2    0      2      ottstpstdlib      2463    00020000  S   000    239684
object: ottstpstdlib
oic:
   ottstpstdlib
3    1      3      ttaad5203m000     25019   00000010  S   000    332444
object: ttaad5203m000
oic:
   ottaad5203
   ottstp_stddll
   ottstpstandard
forms:
   fttaad5203m0001*
4    3      3      ttaad3405m000     17591   00000010  S   000    299384
object: ttaad3405m000
oic:
   ottaad3405
   ottstp_stddll
   ottstpstandard
forms:
   fttaad3405m0001*
7    4      3      ttstpsplclose     291     00000030  B   000    221712
object: ttstpsplclose
oic:
   ottstpsplclose
   ottstp_stddll
   ottdllbw
```

- Analyse the output:
  Process ended in session ttaad3405m000 in ottdllbw.
  Reviewing the jobdefinition results in:
  The printer that was defined for the session to generate its output to, was a Windows Printer. However no bw is active when job runs in background, and the windows printer could not be accessed.
  → appropriate action can be taken to solve the issue.

V. Documentation

- On line help documentation.

- Solution 100668:
  contains the information on jobmanagement, retrieved from document 7782 US Baan IVTools Administration

- Links to useful documents:
  ftp://ftp.support.baan.com/updates/B40/bw40c/Baan_Job_Daemon_Service_on_Windows_NT.doc
  ftp://ftp.support.baan.com/updates/B40/bw40c/Using_Task_Scheduler_to_schedule_jobs.doc

VI. Solutions

Mind that you always upgrade to the latest available job specific solutions.
The standardprogram has also an important role in the jobmechanism.
The general solution numbers for retrieving the latest version of the standardprogram
Have been listed below.

Standardprogram solutions:

These solutions are updated (bi)weekly.

10077	General solution for standard program B40a
10111	General solution for standard program B40b
10923	General solution for the Standard program version B40c
15555	General solution for the Standard program version B50b
73609	General solution for standard program version 7.0a
100278	General solution for the Standard Program Version 71a