Package 'jobqueue'

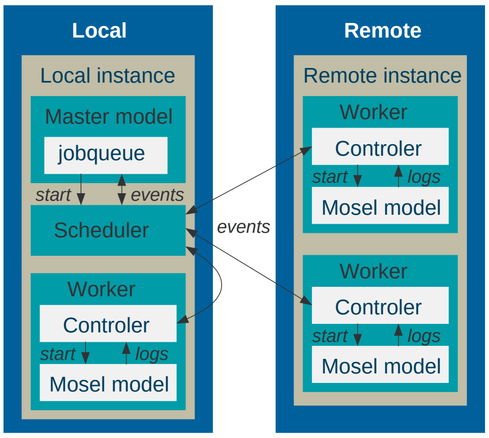
Concurrent remote execution of submodels

Y. Colombani, S. Heipcke

Xpress Optimization, FICO
`http://www.fico.com/xpress`

Package *jobqueue*: Overview

- Configuration of *computation jobs* (models+other required files) and node lists
- Management of *task queues*
- Supervision of nodes and model execution on workers via *controler programs*
- *Reporting* model output, errors, status and host information
- Package requires Mosel 5, but remote (worker) instances can use older releases



Package *jobqueue*: Usage

1. Configuration of queue(s)
   - initial status check, connections are opened when needed

```
public declarations
  queue: integer
  nd1,nd2,nd3,nd4: integer
end-declarations

queue:=queuenew                       ! Create a new queue

nd1:=queueaddnode(queue,"*",2)        ! Use same instance as current (no 'connect')
nd2:=queueaddnode(queue,"",3)         ! New instance (via rcmd) on current machine
nd3:=queueaddnode(queue,"somemach")   ! New instance (xprmsrv) on remote machine
nd4:=queueaddnode(queue,"localhost")  ! New instance via xprmsrv on current
```

2. Definition of jobs

```
public declarations
  job,job2,job3: jq_job
end-declarations

jobinit(job,"simple.bim")           ! Model is already compiled

jobinit(job2,"simplewdata.mos")     ! Model gets compiled by 'jobqueue'
jobadd(job2,"simpledata.txt")       ! Add one (or more) data files for this model
jobsetresult(job2,"results.txt")    ! Specify the result file (single file)

jobinit(job3,"simple.mos",2)        ! 2 attempts to run model if server failure
```

3. Turn jobs into computation tasks by adding them to a queue
   - submodel execution starts automatically

```
public declarations
  job,job2: jq_job
  tasks: list of integer
  queue: integer
  mpar: text
end-declarations

forall(i in 1..4) do               ! Queue some jobs with runtime parameters
  setmodpar(mpar, "WAIT_IN_SUBMODEL", i)
  setmodpar(mpar, "IMPORTANT_PARAM", text("Hello World ") + (i^2))
  tasks+=[queueaddjob(queue, job, mpar)]
end-do
tasks+=[queueaddjob(queue, job2)]  ! Queue another job
```

4. Wait for termination
   - can perform conditional wait (time limit)
   - explicit deletion of tasks or queues if not busy/being processed

```
public declarations
  queue, rti: integer
  rtb: boolean
end-declarations

rti:=queuewait(queue)           ! Wait for all tasks to terminate
writeln("End of wait: ",rti)    ! Number of tasks executing or waiting

! Alternative forms:
rti:=queuewaitnext(queue)       ! Wait for next task termination
rti:=queuewait(queue,10)        ! Wait for 10 seconds

! Optional: explicit termination/deletion
queuedeltasks(queue)            ! Delete all pending+terminated tasks
rtb:=queuereset(queue)          ! Del. pending + disconnect nodes (if not busy)
rtb:=queuedel(queue)            ! Delete a queue (only if not busy)
```

Package *jobqueue*: Reporting

- Reporting functionality
  - display queue information

```
public declarations
  queue: integer
  jqi: jq_qinfo
end-declarations

queuegetinfo(queue,jqi)             ! Queue info: workers, pending+running tasks
writeln("Queue info:", jqi)
writeln(queuepending(queue))        ! Display pending tasks
```

- Reporting functionality
  - submodel status and output (default files)

```
public declarations
  status, code: integer
  tasks: list of integer
end-declarations

forall(t in tasks) do
  initializations from taskstatfile(t)    ! Model execution status file
    status code
  end-initializations
  writeln("Status of task ", t, ": ", status, "/", code)
  writeln("Host info task ", t, ":")
  fcopy(taskhostfile(t),0,"",0)           ! Host (node) information file
  if status=0 then
    writeln("Output of task ", t, ":")
    fcopy(taskoutfile(t),0,"",0)          ! Output log file
  else
    writeln("Errors of task ", t, ":")
    fcopy(taskerrfile(t),0,"",0)          ! Error log file
  end-if
end-do
```

- Reporting functionality

- result file (only if specified via `jobsetresult`)

```
public declarations
  tasks: list of integer
  Sol: dynamic array(range) of real
  L: list of text
  val: real
end-declarations

with t=tasks.last do          ! This task has a result output file
  initializations from taskresfile(t)
    Sol L val
  end-initializations
  writeln("**Result values: Sol=", Sol, " L=", L, " val=", val)
end-do
```

Package *jobqueue*: Documentation and download

- Package documentation:
  - run `make.mos` in the jobqueue/source folder with setting `DOC=true` to generate the online documentation via *moseldoc*:

```
    mosel make DOC=true
```

- Distribution via public Git repository:
  https://github.com/fico-xpress/mosel