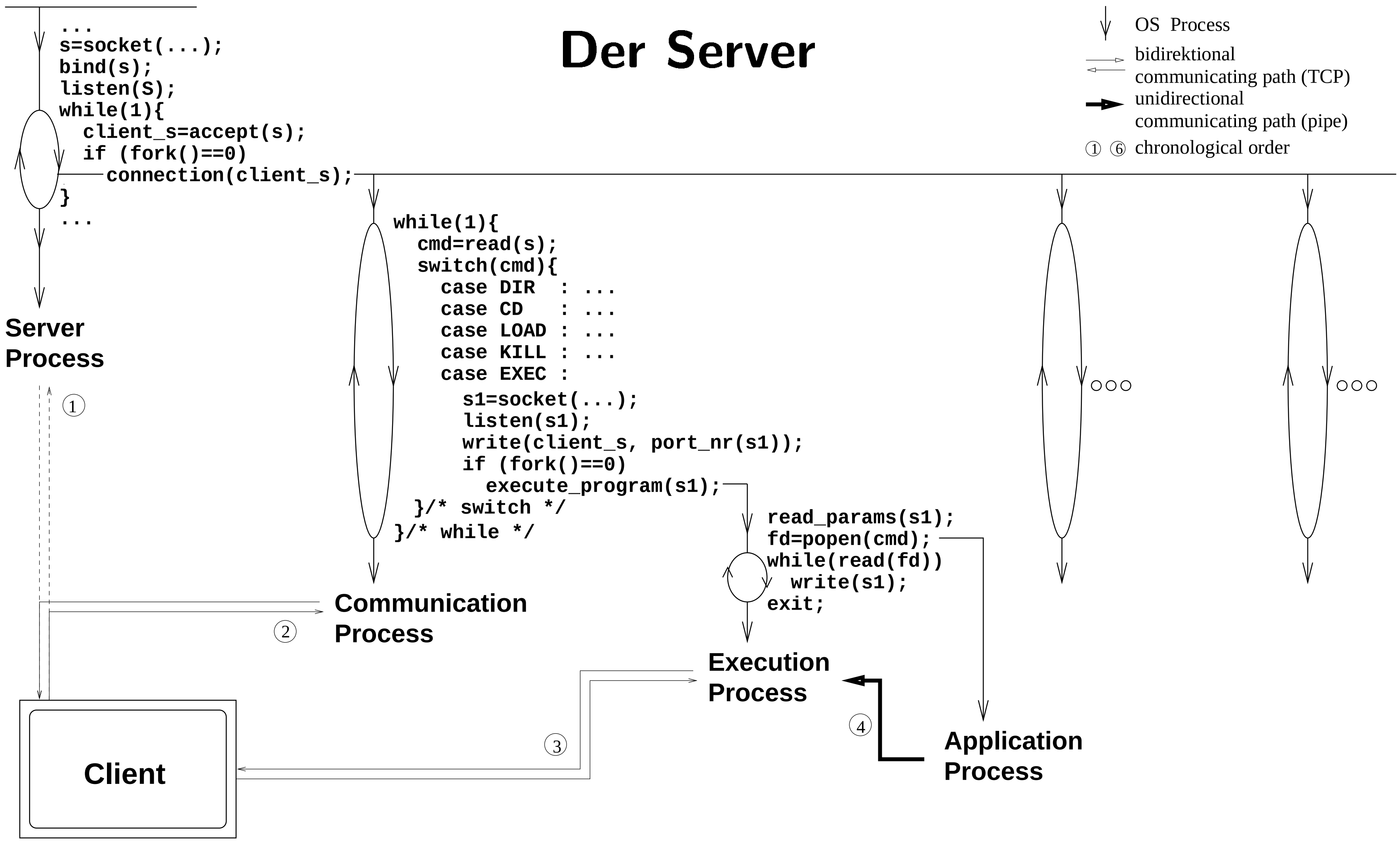
Der Server


...
s=socket(...);
bind(s);
listen(S);
while(1){
client_s=accept(s);
if (fork()==0)
connection(client_s);
}
...
Server Process
while(1){
cmd=read(s);
switch(cmd){
case DIR : ...
case CD : ...
case LOAD : ...
case KILL : ...
case EXEC :
s1=socket(...);
listen(s1);
write(client_s, port_nr(s1));
if (fork()==0)
execute_program(s1);
}/* switch */
}/* while */
Communication Process
read_params(s1);
fd=popen(cmd);
while(read(fd))
write(s1);
exit;
Execution Process
Application Process
Client
1
2
3
4
OS Process
bidirektional communicating path (TCP)
unidirectional communicating path (pipe)
1 6 chronological order

Der Client

Java Thread

bidirectional communication path (TCP)

① ② chronological order

class GbGui

```
. . .
try{
  sock=new Socket(appletHost,sreverPort);
  in=new DataInputStream(sock);
  out=new PrintStream(sock);
}catch(IOException){
  url=new URL("http://"+appletBase+"/start_server");
  url.println("Start server");
  if (url.getLine()=="OK"){
    sock=new Socket(appletHost,sreverPort);
    in=new DataInputStream(sock);
    out=new PrintStream(sock);
  } else error();
}
animationThread = new AnimationThread();
. . .
handleEvent(..) {
  CD ..
  DIR ..
  LOAD ..
  KILL ..
  EXEC : new ExecThread(cmd, options)
}
. . .
```

main client Thread

①

Server

②

class ExecThread

```
animationThread.resume();
sock=new Socket(appletHost,sreverPort);
in=new DataInputStream(sock);
out=new PrintStream(sock);
out.println(cmd + options);
while(in.getLine() != EOF)
  showResult(...);
animationThread.stop()
```

execution Thread

class AnimThrad

```
animationThread.run();
```

animation Thread