Divisie Wiskunde & Informatica
Vrije Universiteit

Toets Bedrijfssystemen
31.03.2005

1a Een bedrijfssysteem can gezien worden als een virtuele machine of als een resource manager. Leg het verschil uit. *5pt*

1b UNIX bedrijfssystemen representeren een harde schijf als een zogeheten *block special file*. Leg uit wat voor soort bestand dit is. *5pt*

1c Er is geen DELETE *file system call* in MINIX. Hoe wordt dan een bestand vewijderd? *5pt*

2a Wat wordt bedoeld met de context van een process? *5pt*

2b Als er een hardware interrupt optreedt, is er een moment waarop de software de besturing overneemt van de hardware. Leg uit waneer. *5pt*

2c Leg uit wat de *test-set-lock* instructie (TSL) doet, en hoe die gebruikt kan worden om een kritieke sectie te beschermen. *5pt*

2d Wat doet de procedure MINI_SEND in MINIX (zie ook de code op de andere pagina)? *5pt*

3a Wat zijn de noodzakelijk en voldoende voorwaarden voor het ontstaan van een deadlock? *5pt*

3b Wat is het fundamentele verschil tussen I/O tasks en processen in MINIX? *5pt*

DEZE TOETS BESLAAT TWEE PAGINA'S

```
0001  PRIVATE int mini_send(caller_ptr, dest, m_ptr)
0002  register struct proc *caller_ptr;
0003  int dest;
0004  message *m_ptr;
0005  {
0006    register struct proc *dest_ptr, *next_ptr;
0007    vir_bytes vb;
0008    vir_clicks vlo, vhi;
0009
0010    if (isuserp(caller_ptr) && !issysentn(dest)) return(E_BAD_DEST);
0011    dest_ptr = proc_addr(dest);
0012    if (dest_ptr->p_flags & P_SLOT_FREE) return(E_BAD_DEST);
0013
0014    vb = (vir_bytes) m_ptr;
0015    vlo = vb >> CLICK_SHIFT;
0016    vhi = (vb + MESS_SIZE - 1) >> CLICK_SHIFT;
0017    if (vhi < vlo ||
0018        vhi - caller_ptr->p_map[D].mem_vir >= caller_ptr->p_map[D].mem_len)
0019      return(EFAULT);
0020
0021    if (dest_ptr->p_flags & SENDING) {
0022      next_ptr = proc_addr(dest_ptr->p_sendto);
0023      while (TRUE) {
0024          if (next_ptr == caller_ptr) return(ELOCKED);
0025          if (next_ptr->p_flags & SENDING)
0026              next_ptr = proc_addr(next_ptr->p_sendto);
0027          else
0028              break;
0029      }
0030    }
0031
0032    if ( (dest_ptr->p_flags & (RECEIVING | SENDING)) == RECEIVING &&
0033         (dest_ptr->p_getfrom == ANY ||
0034          dest_ptr->p_getfrom == proc_number(caller_ptr))) {
0035      CopyMess(proc_number(caller_ptr), caller_ptr, m_ptr, dest_ptr,
0036           dest_ptr->p_messbuf);
0037      dest_ptr->p_flags &= ~RECEIVING;
0038      if (dest_ptr->p_flags == 0) ready(dest_ptr);
0039    } else {
0040      caller_ptr->p_messbuf = m_ptr;
0041      if (caller_ptr->p_flags == 0) unready(caller_ptr);
0042      caller_ptr->p_flags |= SENDING;
0043      caller_ptr->p_sendto= dest;
0044
0045      if ( (next_ptr = dest_ptr->p_callerq) == NIL_PROC)
0046          dest_ptr->p_callerq = caller_ptr;
0047      else {
0048          while (next_ptr->p_sendlink != NIL_PROC)
0049              next_ptr = next_ptr->p_sendlink;
0050          next_ptr->p_sendlink = caller_ptr;
0051      }
0052      caller_ptr->p_sendlink = NIL_PROC;
0053    }
0054    return(OK);
0055  }
```

Cijferbepaling: *Het cijfer MT wordt bepaald door de punten die per onderdeel behaald zijn, bij elkaar op te tellen (totaal maximaal 45 punten), en vervolgens daar 5 extra punten bij te tellen. Er zijn dus totaal 50 punten te behalen. Het eindtentamen bestaat uit twee delen. Deel 1 beslaat dezelfde stof als de toets. Laat $P1$ het aantal behaalde punten voor deel 1 zijn; $P2$ dat voor deel 2 (elk maximaal 50 punten). Het aantal punten E voor het eindcijfer is dan $E = \max\{MT, P1\} + P2$. De toets telt alleen mee voor het eerste volledige tentamen.*