Over seinpalen.

Wij beschouwen een aantal onderling "zwak gekoppelde", in zichzelf sequentiele processen. Onder de "zwakke koppeling" versta ik, dat ze op bepaalde punten rekening met elkaar kunnen moeten houden. Als bv. een aantal processen af en toe wel eens van een of andere faciliteit gebruik wil maken, die maar een proces teglijk kan bedienen, dan betekent dit, dat de processen wel eens even op elkaar kunnen moeten wachten. Als het ene proces informatie verwerkt, dat door een ander proces geleverd moet worden, dan is het ook duidelijk, dat het eerste op het laatste kan moeten wachten. M.a.w. de processen moeten ten opzichte van elkaar in zekere mate gesynchroniseerd kunnen worden.

Om de processen in de gelegenheid te stellen onderling informatie uit te wisselen over elkaars staat van vordering, is er een gemeenschappelijk geheugen ingevoerd. De elementen van dit geheugen zijn non-negatieve integers, die we seinpalen noemen.

In de individuele processen zijn de voor de onderlinge synchronisatie belangrijke punten gemarkeerd, doordat daar is aangegeven, dat bij passering van zo'n punt op bepaalde seinpalen een bepaalde operatie moet worden uitgevoerd. De individuele processen hebben hier de keuze uit twee operatie's, de zg. V-operatie en de zg. P-operatie, welke wij hieronder zullen beschrijven.

In het volgende nemen wij aan, dat S1, S2, S3 etc. namen zijn van toegankelijke seinpalen (niet alle seinpalen hoeven in elk proces toegankelijk te zijn!); de V- en de P-operatie zullen wij schrijven als procedure statement.

De V-operatie ("Verhoog").

De V-operatie heeft betrekking op minstens 1 seinpaal, dus bv. "V(S1)" of "V(S1, S2, S3)". Als een van de individuele processen de V-operatie uitvoert, dan is het effect, dat alle er dit maal bij opgegeven seinpalen in één ondeelbare handeling met 1 worden verhoogd.

Opm.1 De toevoeging "in één ondeelbare handeling" bedoelt hetvolgende uit te drukken. Stel, dat de waarde van de seinpaal S1 = 3 is en dat dan bv. twee van de (simultaan werkende!) processen "tegelijkertijd" de operatie V(S1) willen uitvoeren. Tengevolge van de ondeelbaarheid van de handeling mogen we ons dan voorstellen, dat deze twee V-operatie's op dezelfde seinpaal in een overigens niet ter zake doende volgorde in successie uitgevoerd worden, zodat na afloop S1 = 5 is en niet een van de ophogingen bv. onder tafel is geraakt.

Opm.2 De V-operatie met meer dan 1 argument is logischerwijs niet noodzakelijk, maar wel elegant. In de statement "V(S1, S2)" wordt een simultane verhoging van beide seinpalen gevraagd; vervangen we dit in een van de individuele processen door "V(S1);V(S2)", dan vragen we heel expliciet om ophoging in een bepaalde volgorde. Het zou wel eens wat minder leuk kunnen zijn om daartoe gedwongen te zijn, als men liever "neutraal" een aantal seinpalen simultaan wil ophogen.

Opm.3 Als de V-operatie met meer dan 1 argument wel wordt opgenomen, dan zullen wij ons voorlopig beperken tot het geval, dat de er bij opgegeven seinpalen verschillend zijn.

De P-operatie ("Prolaag").

In een individueel proces markeert de P-operatie de tentatieve passering van dit punt. De P-operatie heeft betrekking op 1 of meer seinpalen, dus bv. "P(S1)" of "P(S1, S2, S3)". Als de P-operatie in een van de individuele processen geentameerd is,

dan is daarmee een operatie begonnen, die slechts beeindigd kan worden op een moment, dat alle er bij opgegeven seinpalen positief zijn. Beeindiging van een P-operatie impliceert, dat alle er bij opgegeven seinpalen met 1 verlaagd worden en ook deze beeindiging geldt als één ondeelbare handeling. Ook voor de P-operatie beperken we ons tot het geval, dat de er bij opgegeven seinpalen onderling verschillend zijn.

Opm.1 De P-operatie met meer dan 1 argument is logisch wel noodzakelijk.

Opm.2 Vele seinpalen nemen slechts de waarden 0 en 1 aan. In dat geval fungeert de V-operatie als "baanvak vrijgeven"; de P-operatie, de tentatieve passering, kan slechts voltooid worden, als de betrokken seinpaal (of seinpalen) op veilig staat en passering impliceert dan een op onveilig zetten.

Enige voorbeeldjes voor het gebruik van seinpalen.

Voorbeeld 1. Als wij een klasje machines (alias processen) Xi hebben (dwz. X0, X1, X2, etc....) en in elk proces komt een critische sectie voor, critisch in die zin, dat geen twee kritische secties tegelijkertijd onder behandeling mogen zijn, dan kunnen we dit bereiken met een seinpaal, zeg SX, die in dit simpele geval slechts tweewaardig zal zijn;

SX = 0 zal betekenen: een van de machines Xi is aan zijn critische sectie bezig
SX = 1 zal betekenen: geen van de machines Xi is aan zijn critische sectie bezig.

De omschrijving van alle processen is nu gelijkluidend:

"LXi: P(SX); TXi; V(SX); rest proces Xi; goto LXi"

Als we alle processen Xi op hun gelabelde punt (dwz. "aan het begin van de regel") zouden starten met de beginwaarde SX = 2, dan zouden we verwezenlijkt hebben, dat ongeacht de omvang van het klasje processen Xi en nooit meer dan twee simultaan aan hun critische sectie bezig zouden zijn. Dit is kennelijk een generalisatie van de probleemstelling van wederzijdse uitsluiting. (Het is precies de situatie van bv. n tape decks aan twee kanalen.)

Wij vestigen er de aandacht op, dat de formulering van de individuele processen Xi onafhankelijk is van de omvang van het klasje Xi, iets wat met het oog op de dynamische variatie van deze omvang wel hoogst gewenst is. Ook de maximaal toegestane simultaneiteit voor de critische secties dringt niet tot de formulering van de individuele processen door.

Voorbeeld 2. Nu beschouwen we een groepje machines Xi en een groepje machines Yj, elk met hun critische sectie TXi resp. TYj. Uitvoering van een critische sectie dient uitvoering van alle andere critische secties uit te sluiten, maar tevens eisen we , dat de uitvoering van een TX-sectie en een TY-sectie alternerende gebeurtenissen zijn.

We kunnen dit bereiken met twee tweewaardige seinpalen, zeg. SX en SY.

SX = 1 betekent, dat nu eerst een TX-sectie aan de beurt is
SY = 1 betekent, dat nu eerst een TY-sectie aan de beurt is.

De programma's voor de machines luiden:

"LXi: P(SX); TXi; V(SY); proces Xi; goto LXi" en

"LYj: P(SY); TYj; V(SX); proces Yj; goto LYj" .

Als de processen alle "aan het begin van de regel" gestart worden moet SX = 1 en SY = 0 zijn of andersom.

Voorbeeld 3. Tenslotte beschouwen we een klasje machines Xi, die informatie-eenheden willen lozen in een cyclische buffer met een capaciteit van N informatie-eenheden; voorts een klasje machines Yj, die informatieeenheden uit deze buffer willen verwerken.

Omdat vullen van de buffer administratieve maatregelen met de "vulwijzer" impliceert -en voor legen mutatis mutandis hetzelfde geldt- eisen we bovendien, dat vullen slechts door 1 machine Xi tegelijkertijd kan geschieden en evenzo, dat legen slechts door 1 machine Yj tegelijkertijd kan geschieden.

We voeren hiervoor in vier seinpalen:

SX1 = aantal vrije plaatsen in de buffer (aanvankelijk = N)
SX2 = 0 als een van de machines Xi aan het vullen is, anders = 1
SY1 = aantal gevulde plaatsen in de buffer (aanvankelijk = 0)
SY2 = 0 als een van de machines Yj aan het legen is (anders = 1).

Er zal gelden $N - 1 \leq SX1 + SY1 \leq N$.

De machines zijn nu:

"LXi: P(SX1,SX2); vul volgende plaats van de buffer; V(SY1,SX2);.....; goto LXi"

"LYj: P(SY1,SY2); leeg volgende plaats van de buffer; V(SX1,SY2);....; goto LYj"

Hardware voorzieningen.

Onze cyclische processen gaan we nu in twee groepen onderverdelen. Die van de ene groep heten " de concrete machines" , die van de andere groep heten " de abstracte machines".

De "concrete machines" zijn de transput-apparaten. Dit zijn alle (cyclische) processen, die met een eigen tijdsbewustzijn gedurende een zekere periode autonoom hun werk doen. Onder "autonoom" verstaan wij ni dit verband onafhankelijk van de besturing van de centrale computer. Behalve transporten van en naar de buitenwereld (paper tape, printer, ponskaarten etc.) valt hieronder ook transporteren tussen kerngeheugen enerzijds en trommel of magneetband anderzijds. Het zijn die processen, die, mits eenmaal door de X8 in gang gezet, simultaan met de werkende centrale computer plaatsvinden.

De "abstracte machines" zijn de onderling asynchrone programma's. Aangezien de centrale computer de facto steeds maar met 1 programma bezig is, kunnen wij stellen, dat van de abstracte machines er op elk moment hoogstens eentje werkt. (Het is mogelijk, dat geen van deze programma's werkt: de centrale computer "zit dan in de coordinator". De coordinator wordt niet als een van de abstracte machines beschouwd.)

Zolang seinpalen louter door abstracte machines geselecteerd worden, zijn hiervoor geen speciale hardware voorzieningen nodig (behalve de doofheid): elke geheugenplaats kan als geprogrammeerde seinpaal fungeren

Een seinpaal, die louter door de concrete machines geselecteerd wordt (ik laat even in het midden, of die er zullen zijn, het is heel goed denkbaar) is een aangelegenheid, die zich geheel buiten de X8 afspeelt en hoeft ons op het ogenblik dus ook niet te interesseren.

Een seinpaal, die zowel door een concrete als door een abstracte machine geselecteerd moet kunnen worden, vraagt echter onze speciale aandacht. Het is duidelijk, dat hiervoor wel speciale hardware voorzieningen en rigoureuze conventies nodig zijn.

In principe is elke "hardware seinpaal" een (per installatie) vaste geheugen-

plaats; door het transput-apparaat kan daar 1 van worden afgetrokken (externe P-operatie) of 1 bij opgeteld worden (externe V-operatie).

Als de centrale computer deze seinpalen moet ophogen of aflagen, dan moet dit gebeuren met behulp van de additieve uitopdracht. In dit geval is nl. interferentie tussen beide additieve operaties (nl. een interne en een externe) niet mogelijk.

Een complicatie is, dat aan elke hardware seinpaal een flip-flop geassocieerd is, die aangeeft of de bijbehorende seinpaal positief is. (Voor de noodzaak van deze flip-flop, zie later.)

We kunnen nu twee soorten hardware seinpaal onderscheiden (als regel heeft elk transput-apparaat er van beide soorten eentje):

1) Een startopdrachtentelling. In dit geval verricht de centrale computer de V-operatie, en het transputmechanisme verricht de P-operatie. De geassocieerde flip-flop heet in zo'n geval een Acflop.

2) Een ingrepentelling. In dit geval verricht de centrale computer de P-operatie en het transput-apparaat de V-operatie. De geassocieerde flip-flop heet dan een Inflop.

In elke installatie zijn de concrete machines genummerd. Uit het nummer is dan af te leiden, welke geheugenplaatsen voor startopdrachten- en ingrepentelling gereserveerd zijn; de bijbehorende Acflop en Inflop zijn onder opgave van het apparaatnummer bereikbaar.

Wij laten nu zien, hoe de centrale computer bij wijziging van een hardware seinpaal de bijbehorende flip-flop zo nodig kan aanpassen. (Aanpassing van zo'n flip-flop ten gevolge van wijziging door de transput-apparatuur is niet de zorg van de centrale computer, maar van de bouwers.)

De V-operatie door de centrale computer.

Zij t de integer, die in het kerngeheugen de seinpaal representeert; de geassocieerde flip-flop noemen we Acflop. Het transput-apparaat zal een volgende startopdracht accepteren, als die er is en het er aantoe is. Het verricht in de voltooiing van de P-operatie op deze seinpaal de handeling, die beschreven kan worden door

" if Acflop then begin t:= t - 1; if t ≤ 0 then Acflop:= false".

Deze P-operatie, die beschouwd moet worden als onderdeel van het consumeren van de volgende startopdracht, heeft dus als mogelijk gevolg, dat het opnemen van volgende startopdrachten tot nader order niet meer zal plaats vinden; dit nl. als tengevolge van uitputting van de voorraad Acflop op false gezet is.

Additionele spelregels zijn:
a) dat deze handeling vanuit de computer bezien moet worden als ondeelbare handeling -dit schept zijn voordelen- maar dat anderzijds de computer niet over de mogelijkheid zal beschikken om deze handeling over een aantal opdrachten te verbieden -en dit schept zijn problemen.
b) dat bij beinvloeding van t en Acflop door de computer het pertinent verboden is, als hoe kort ook ten onrechte Acflop true zou zijn. Op dat moment zou nl. het transput-apparaat meteen tenonrechte kunnen gaan werken.

Als de centrale computer -ter melding dat de volgende startinformatie is klaargezet- op deze seinpaal de V-operatie moet uitvoeren, geschiedde dit door het volgende stukje programma:

```
"t:= t + 1;
 if  non Acflop then
     begin  if 0 < t
            then Acflop:= true  end"
```

De rechtvaardiging van dit stukje programma is als volgt. Na de ophoging wordt Acflop getest; als Acflop true was, dan was dus de telling t positief en kan deze door de ophoging alleen maar positiever geworden zijn en eventueel aanpassen van Acflop ligt dus tot nader order op de weg van de transputmachine. Maar als we XX de Acflop false aantreffen, dan kan de transputmachine de P-operatie niet meer uitvoeren, zodat van die kant t en Acflop ongemoeid gelaten worden en de centrale computer rustig de test kan uitvoeren.

De P-operatie door de centrale computer.

Weer zullen we de seinpaal aanduiden met t; de bijbehorende flip-flop noemen we nu Inflop. Inflop geve aan, of t positief is. De V-operatie, die door de transputmachine uitgevoerd wordt, kan beschrven worden door

```
"t:= t + 1;  if 0 < t then Inflop:= true"   ;
```

van de centrale computer bezien is dit een ondeelbare, niet tegen te houden handeling.

De voltooiing van de P-operatie zal de centrale computer alleen XXXXX uitvoeren als Inflop true is; bij gratie van doofheid -waarbij de waarde van Inflop geen effect heeft, zie later- kan de computer zich permitteren om Inflop tijdelijk false te zetten.

De voltooiing van de P-operatie door de computer bestaat nu uit het volgende stukje programma:

```
" t:= XXXXX t - 1;
  Inflop:= false;
  if  0 < t  then
             Inflop:= true"
```

Essentieel is hierbij aangenomen
a) dat het geen kwaad kan, als X Inflop ten onrechte een tijdje false is
b) dat als transput-machine en computer "tegelijkertijd" proberen om de assignmaat "Inflop:= true" uit te voeren, dat dan na XX afloop zeker Inflop true zal zijn.

De rechtvaardiging van dit stukje programma is nu als volgt. Elke V-operatie voor de assignment "Inflop:= false" uit bovenstaand programma'tje doet niet ter zake, omdat daarna in elk geval Inflop false gezet wordt. Als tijdens de assignment "inflop:= false" de telling t positief is, dan zal de computer uiteindelijk de statement XXX "Inflop:= true" uitvoeren. Door tussengeschoven operatie's V kan de telling t alleen maar nog groter worden en dus wordt Inflop correct achter gelaten. Als tijdens de assignment X "Inflop:= false" de telling t daarentegen non-positief is, moeten we twee gevallen onderscheiden. In het geval, dat de telling t niet door tussengeschoven V-operaties positief wordt, dan zullen zowel de transput-machine als de computer de Inflop verder ongemoeid laten en Inflop blijft dus terecht = false achter. Als door ondergeschoven V-operaties daarentegen de telling positief wordt, dan zal in elk geval door de transput-machine op het moment van omslag de assignment "Inflop:= true" uitgevoerd worden (en mogelijk ten overvloede ook nog een keer door de computer, nl. als ten tijde van de test de telling t al positief is). In dit laatste geval wordt Inflop dus gegarandeerd correct op true achtergelaten.

Opm. De uitleesbaarheid van de Acflops en de tweezijdigheid van de zetbaarheid van de Inflops zijn in de X8 geintroduceerd speciaal om deze programmatjes mogelijk te maken. Deze hardware is in zekere zin de extra prijs die we betalen moeten voor de ononderdrukbaarheid van de autonome operatie's op de seinpalen.

De hardware voor de ingreep.

In het volgende zullen we Inflop true ook door =1, en false door =0 representeren.

Zoals gezegd zijnde Inflops genummerd. In een installatie met minder dan 27 Inflops zijn deze gezamenlijk als bits van het zg. (1ste) Inflopwoord uitleesbaar met behulp van een speciale communicatieopdracht. (Komen er meer Inflops, dan wordt een 2de Inflopwoord geintroduceerd, etc.etc. Ik verwacht, dat er per Inflopwoord hoogstens 26 bits gebruikt zullen worden en dat het tekenbit van een Inflopwoord altijd = 0 is met het oog op normering.)

Opm. Welke poditie de verschillende Inflops in het Inflopwoord innemen is mij niet bekend; hier kon nog wel eens een beslissing over genomen moeten worden.

Bij elke Inflop hoort een luisterbit; ook dit is een XXX flip-flop, die door de X8 individueel in beide richtingen zetbaar is. Ook de luisterbits zijn via een (of zo nodig meer) zg. "luisterwoord" uitleesbaar. Een luisterbit neemt in het luisterwoord dezelfde positie in als de overeenkomstige Inflop in het Inflopwoord.

Tenslotte is er een flip-flop, die aangeeft of de X8 "doof" dan wel "horend" is. Een ingreep vindt plaats als

a) het collatieresultaat van Inflopwoord(en) en luisterwoord(en) niet enkel nullen bevat
b) en tevens de machine horend is.

Opm. De doofmakende opdrachten van de X8 werken "instantaan"; het is dus niet mogelijk, dat na de opdracht "maak doof" nog net een interruptie plaatsvindt (zoals dit bij de X1 het geval was).

Pro memorie: Dit moet gecontroleerd worden voor de herstellende sprong, die de overgang horend → doof kan bewerkstelligen;
nagevraagd moet worden of ook de horend makende opdrachten instantaan werken.

De ingreep bestaat daarui, dat:

a) er een speciale subroutinesprong wordt uitgevoerd (met een voor dit doel gereserveerde labda)
b) in de uitvoering van deze subroutinesprong de normale ophoging van de opdrachtteller onderdrukt wordt en de machine onmiddellijk doof gemaakt wordt.

De ingreepsprong verwijst de besturing naar de "ingreeproutine"; deze begint met registerinhouden e.d. te redden om te zorgen dat het nu onderbroken programma later voortgezet kan worden alsof er niets gebeurd is. Vervolgens zal het de Inflops moeten analyseren om te ontdekken welk extern apparaat zo nodig iets te melden had en wat, etc.

De coordinator.

Een belangrijk onderdeel van de coordinator is de ingreeproutine. Ik verwacht dat de ingreeproutine zozeer met de coordinator verweven zal zijn, dat een gescheiden behandeling nauwelijks mogelijk is

De coordinator-wachtlijst bevat alle programma's in de machine, onderverdeeld in twee categorieen, de geblokkeerden en de uitvoerbaren.

Geblokkeerde programma's zijn programma's, die niet verder kunnen, doordat ze op een seinpaal staan te wachten, op de voltooiing van een voor hun voortzetting noodzakelijke gebeurtenis.

Uitvoerbare programma's zijn programma's, die wel verder kunnen, ware het niet dat de X8 maar 1 van hen verder kan helpen. (Ik neem aan, dat het programma, dat door de X8 uitgevoerd wordt, onder de uitvoerbare gerangschikt blijft: de term "wachtlijst" is dan wat merkwaardig, maar laten we er maar in berusten. Als de besturing echt in de coordinator zit, dan is de term in orde.)

Het effect van de P-operatie kan zijn, dat een programma van uitvoerbaar nu geblokkeerd wordt, het effect van een V-operatie kan zijn, dat een (ander) programma uit de groep der geblokkeerde naar de uitvoerbare wordt overgeheveld. Een programma, dat in actie door een interruptie onderbroken wordt, blijft gerangschikt onder de uitvoerbare!

We kunnen het ook zo zien: als in een programma een P-operatie geinitieerd wordt, dan was op dat moment het programma in actie, dus uitvoerbaar. Nu zijn er twee gevallen. Of de heersende waarden van de betroffen seinpalen vormen geen beletsel, of zij doen dit wel. In het eerste geval worden zij afgelaagd, de P-operatie is daarmee voltooid en het programma blijft uitvoerbaar. (of het ook in actie blijft is een heel ander chapiter!) In het tweede geval wordt het programma uit de groep der uitvoerbaren gehaald. De groep der geblokkeerde bevat dus alleen alle programma's, die in een geinitieerde maar niet voltooide P-operatie zijn blijven hangen.

De P-operatie's van de abstracte machines worden dus beslist niet als een permanent observeren van de betrokken seinpalen gespeeld, in tegendeel: de abstracte machines, die geblokkeerd zijn, sluimeren. Nu de abstracte machines, eenmaal geblokkeerd zijnde, niet meer gevoelig zijn voor het positief zijn van seinpalen, moet de coordinator gevoelig zijn voor het positief worden van seinpalen. Maw. als 1 of meer seinpalen positief worden, heeft de coordinator de plicht om vast te stellen, of er nu geblokkeerde uit hun blokkade geholpen kunnen worden en zo ja, dan dientde coordinator dit te doen. (Bij de V-operatie met meer dan 1 argument kunnen er dus twee abstracte machines naar de uitvoerbare overgeheveld moeten worden.) Hier ligt een vrij stringente plicht voor de coordinator: er moet immers vermeden worden, dat een abstracte machine geblokkeerd is door een geprogrammeerde seinpaal, die inmiddels "onopgemerkt" positief is geworden. De abstracte machine kijkt niet meer, de seinpaal seint niet meer en de boel zit vast!

Het is om bovenstaande redenen, dat wij voor de V-operatie, die tot nog toe als ophoging beschreven is, beslist niet zullen volstaan met een simpele additieve uitopdracht. De geprogrammeerde V-operatie wordt een aanroep van een coordinatorroutine: deze zal de meegegeven seinpalen verhogen en tevens kijken, wat de aan deze verhoging te verbinden consequenties zijn!

Hier zien we inmiddels de verweving van de ingreeproutine en de coordinator. De ingreeproutine sec doet niet veel meer, dan vaststellen op welke (hardware) seinpaal de V-operatie is uitgevoerd en ook dit zal in de regel tot gevolg hebben, dat een abstracte machine gedeblokkeerd wordt.

De geblokkeerde programma's bevinden zich in de coordinatorwachtlijst onder opgave van de seinpalen, die bij de voltooiing der P-operatie afgelaagd moeten worden.

Ik stel mij voor, dat de onderverdeling van de wachtlijst geschieden zal niet door verplaatsing maar door kettingrijgen. Ik neem aan dat elke abstracte machine bij creatie een nummer toegekend krijgt -het laagste vrije nummer- en dit nummer gedurende zijn bestaan zal blijven behouden. Ik neem voorts aan, dat onder dit nummer in de wachtlijst direct geselecteerd kan worden.

Ik wou alle uitvoerbare machines met een ketting aan elkaar rijgen, verder wou ik bij elke non-positieve seinpaal de beginschakel van een mogelijk niet lege blokkadeketting maken.

Als nu een van de abstracte machines de P-operatie wil uitvoeren, dan worden de meegegeven seinpalen onderzocht. Als een van de seinpalen non-positief bevonden wordt, dan worden de andere seinpalen helemaal niet meer bekeken, want deze abstracte machine wordt meteen aan de blokkadeketting van deze seinpaal gehangen. De opmerking is nl. dat voordat deze seinpaal positief is, de P-operatie in elk geval niet voor voltooiing in aanmerking komt.

Als nu een V-operatie op een seinpaal wordt uitgevoerd, dan kan de coordinator nu op vrij snelle wijze de mogelijke consequenties hiervan nagaan. Was de blokkade-ketting leeg, dan stond er niets op deze V-operatie te wachten en geen van de geblokkeerde machines komt dus voor deblokkade in aanmerking. Als de blokkadeket-ting van deze seinpaal niet leeg is, dan onderzoekt men een abstracte machine in de ketting: als de overige seinpalen geen beletsel vormen, dan kan een P-operatie voltooid worden en gaat deze abstracte machine over naar de uitvoerbare. Als een andere seinpaal wel een beletsel vormt, dan gaat de abstracte machine over naar de blokkadeketting van die laatste seinpaal. Repetatur, totdat of de seinpaal weer nul is (succesvolle deblokkade) of de ketting leeg is (of beide). Het is vanzelfsprekend, dat door het dusdanig afwerken van de V-operatie het uitgesloten is dat een positieve seinpaal met non-lege blokkadeketting zou overblijven.

Salvo errore et omissione zou de coordinator naar aanleiding van een V-operatie dankzij deze kettingen nu vrij snel moeten kunnen vaststellen, of hieraan verdere consequenties verbonden zijn en zo ja, welke.

In dit schema passen de luisterbits voortreffelijk: men maakt de luisterbit van een hardware seinpaal = 1, als zijn blokkadeketting gevuld is, anders = 0. Immers: als de blokkadeketting leeg is, dan staat er in eerste instantie niets op deze seinpaal te wachten. Wat zullen we dan een ingreep introduceren, als deze seinpaal positief wordt? Zo komen de luisterbits naar voren als middel om onnodige ingrepen te onderdrukken. Of dit veel zoden aan de dijk zet, valt nog ernstig te bezien, het zou best eens zo kunnen zijn, dat er haast altijd op een ingreep gewacht wordt.