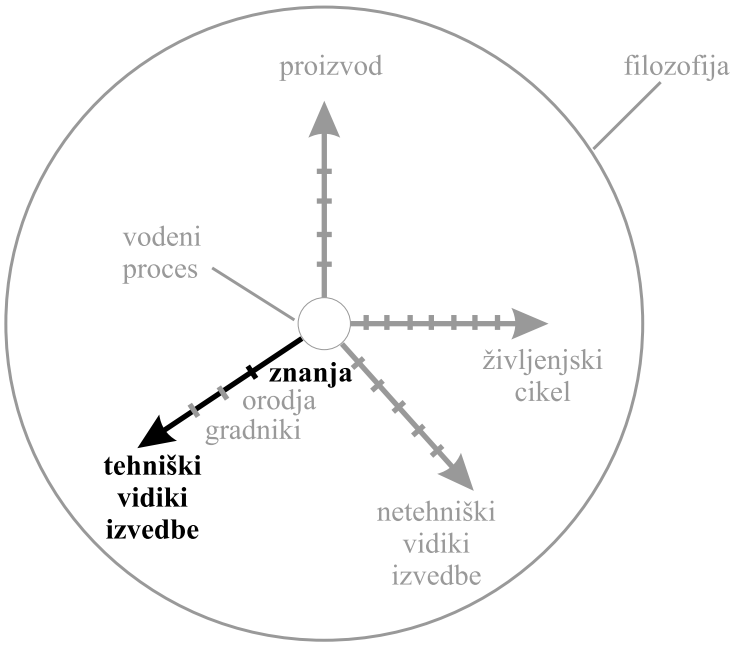
7. Tehniški vidiki izvedbe

Znanja in veščine



7.1 Uvod

Pričujoče poglavje je prvo v vrsti treh, ki se nanašajo na tehniške vidike izvedbe sistemov za vodenje. Ideja strukture: znanja in veščine, orodja in gradniki se nam zdi pomembna zaradi tega, ker so v praksi, posebej pa pri nepoznavalcih in začetnikih stvari pogosto nejasne. Posebej problematično je enačenje metod in orodij, ki pripelje do utvare, da je probleme možno reševati že, če imaš orodje, ni pa ti treba imeti ustreznih znanj in veščin. (Če imaš zidarsko žlico in zidak še ne pomeni, da znaš zidati.) Zato bi bilo seveda idealno, če bi lahko potegnili jasno ločnico med znanji in veščinami, orodji ter gradniki. Seveda pa take jasne ločnice ni in so stvari precej medsebojno povezane in prepletene. Nekateri postopki in metode so močno vezani na specifična orodja (žaganje si npr. težko predstavljamo brez žage, ali pa programiranje brez programskega jezika), spet drugi pa manj (npr. zidanje, matematično modeliranje). Nekatera orodja so bolj prilagojena metodam, druga pa bolj gradnikom in podobno.

V tem poglavju smo skušali zajeti reprezentativni izbor metod, postopkov in tehnik, ki so pomembne pri načrtovanju, izgradnji in uporabi sistemov za vodenje. Večji del poglavja je namenjen splošnim metodam kot so metode matematičnega modeliranja in simulacije ter metodam in postopkom, ki omogočajo realizacijo posameznih funkcij vodenja. Posebni del se nanaša na metode umetne inteligence, ki so v zadnjem desetletju na področju vodenja precej popularne. Zadnji del poglavja pa je namenjen problematiki razvoja programske opreme ter obdelave signalov in komunikacijam. Metode in postopki, ki se nanašajo na ta del namreč predstavljajo osnovo, brez katere se praktično sploh ni mogoče lotiti realizacije konkretnega sistema za vodenje.

7.2 Modeliranje in simulacija

Med metodami in postopki, ki jih je treba obvladovati, da bi lahko uspešno reševali zahtevnejše probleme vodenja in gradili učinkovite in kvalitetne računalniško podprte sisteme za vodenje procesov, so nekatere precej specifične, druge pa so bolj splošne in univerzalne ter se uporabljajo tudi na drugih področjih tehnike pa tudi znanosti. Med slednje spadata prav gotovo matematično modeliranje in simulacija, ki pa pri načrtovanju sistemov vodenja igrata še prav posebej pomembno vlogo.

Pričujoče podpoglavje je namenjeno osnovnim definicijam, ki se tičejo matematičnega modeliranja in simulacije, opredelitvi različnih vrst parametričnih in neparametričnih dinamičnih modelov, ki so seveda osnovni predmet našega zanimanja, predstavitvi različnih pristopov k modeliranju ter metodam za simulacijo zveznih sistemov.

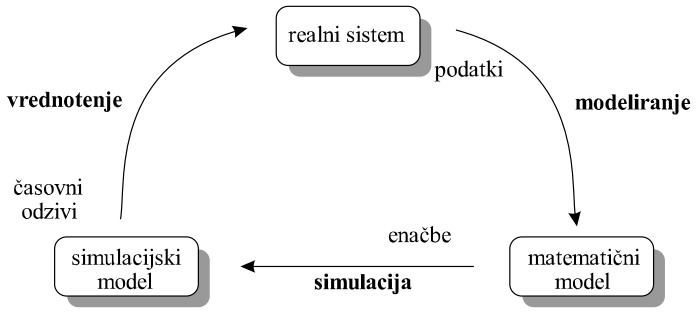
7.2.1 Matematično modeliranje procesov

Modeliranje in simulacija sta dva neločljiva postopka, ki vsebujeta kompleksne aktivnosti v zvezi s konstrukcijo modelov in eksperimentiranje z modeli v smislu pridobivanja podatkov o obnašanju modeliranega procesa (Matko in soavtorji, 1992).

Pri tem je modeliranje vezano predvsem na relacije med realnim procesom in njegovimi modeli, simulacija se ukvarja s povezavo med matematičnim in simulacijskim modelom, slednji pa tvori kot svoj izhod časovne odzive, ki jih vrednotimo glede na obravnavani proces, kar nekako zaključi krog prikazan na Sl. 7.1.

Namen študija sistemov s pomočjo modeliranja in simulacije je doseganje različnih ciljev, ne da bi morali ekperimentirati na realnem objektu. Pristop je uporaben celo v primeru, ko obravnavani sistem še ne obstaja. Najpogostejši cilji so:

- poboljšati poznavanje in razumevanje nekaterih mehanizmov delovanja obravnavanega sistema;
- napovedovati obnašanje sistema v različnih situacijah, kjer kakršnikoli nivo predkicije predstavlja koristno informacijo;
- omogočiti načrtovanje sistemov vodenja in njihovo vrednotenje;
- oceniti parametre procesa, ki niso direktno merljivi;
- preizkušati občutljivost sistemskih parametrov;
- optimirati obnašanje sistema;
- omogočiti učinkovito odkrivanje napak v sistemu;
- omogočiti raziskavo primerov, ki bi bili v realnem svetu dragi, tvegani ali problematični, kar je pomembno tudi pri simulatorjih za učenje operaterjev.



Sl. 7.1. Odnos med sistemom in modeliranjem ter simulacijo

Čeprav obstaja mnogo tehnik modeliranja in je na voljo precej različnih simulacijskih orodij, se moramo zavedati, da niti model niti računalnik ne moreta popolnoma nadomestiti človeških odločitev, presoje, intuicije in izkušenj, ki še vedno igrajo odločilno vlogo pri določanju vrednosti in uporabnosti modelov v praktičnih aplikacijah. Pri tem je modeliranje bolj problemsko orientirano, medtem ko je simulacija relativno neodvisna od obravnavanega primera.

Splošno o modelih

Skušajmo sedaj našteti nekaj definicij modela, ki naj bi prikazale njegove osnovne značilnosti:

- ***model je nadomestek nekega konkretnega sistema ali opreme;***

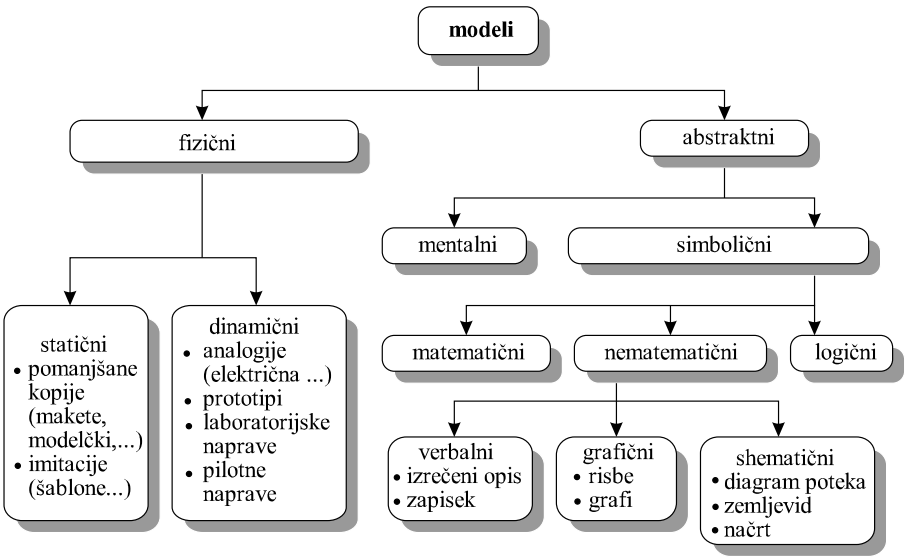
- ***model je poenostavljena predstavitev realnega sistema, ki naj omogoči razumevanje, razlago, spremembe ali ohranitve lastnosti, napovedovanje in morda tudi vodenje obravnavanega sistema;***
- ***model je objekt ali koncept, ki predstavlja nekaj drugega, se pravi, da je realnost prenešena v neko uporabno in razumljivo obliko.***

Iz tega izhajajo nekatere pomembne lastnosti modelov kot so npr.:

- model obravnava le bistvene aspekte realnega sistema;
- model poudarja tiste učinke gradnikov sistema, ki so pomembni s stališča namena modeliranja;
- model mora predstaviti naše znanje o sistemu v primerni obliki tudi na nekaterih drugih medijih (papir, računalniški spomin);
- model mora biti kar najbolj enostaven, saj je razvoj univerzalnega modela nemogoč, razvoj prekompleksnega modela pa je nepraktičen in neekonomičen.

Poznamo seveda veliko različnih vrst modelov.

Eno od možnih delitev modelov na različne vrste prikazuje Sl. 7.2.



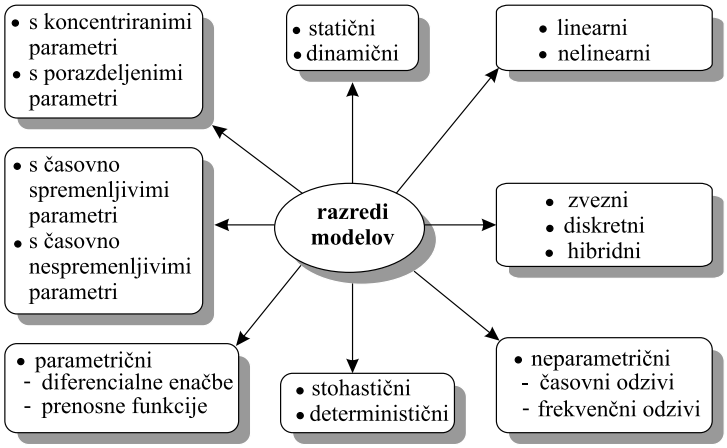
Sl. 7.2. Vrste modelov

Iz slike vidimo, da gre za različne predstavitve nekega realnega procesa, ki smo jih v veliki meri opisali že v poglavju 3, ko smo govorili o načinih predstavitev procesov.

Najpomembenjši in tudi največ uporabljani so *matematični modeli*. Informacija, ki jo dajejo je splošno razumljiva, enoumno razložljiva in precizna, pri čemer je manipulacija in vrednotenje možnih različic relativno poceni. Matematični model tako predstavlja neko preslikavo relacij med fizičnimi spremenljivkami modeliranega sistema v ustrezne

matematične strukture (sisteme algebrajskih in diferencialnih enačb raznih vrst).

Klasifikacija matematičnih modelov omogoča vpogled v možnosti, značaj in obliko modelov. Oglejmo si le enega od možnih načinov razdelitve (Sl. 7.3).



Sl. 7.3. Klasifikacija modelov

Kot vidimo je vidikov, na podlagi katerih lahko klasificiramo procese zelo veliko. Mi se bomo v nadaljevanju osredotočili predvsem na dinamične in deterministične modele v parametrični in neparametrični obliki.

7.2.1.1 Parametrični in neparametrični dinamični modeli

Parametrični in neparametrični modeli igrajo pomembno vlogo v analizi, načrtovanju in vrednotenju sistemov vodenja (Matko, 1991; Zupančič 1995 in 1996).

Parametrični modeli

Glede na klasifikacijo na Sl. 7.3 ***imenujemo vsak opis dinamičnega sistema podan v obliki matematične relacije parametrični model***, saj vsaka relacija vsebuje tudi ustrezne parametre.

Najosnovnejša matematična oblika za zapis dinamičnega sistema je vsekakor *diferencialna enačba*. Za različne modele uporabljamo tudi diferencialne enačbe raznih vrst in tipov (navadne različnih redov, nelinearne, s časovno spremenljivimi koeficienti, parcialne in sisteme omejenih enačb). Če gre za linearne, časovno nespremenljive diferencialne enačbe, jih rešujemo s pomočjo Laplace-ove in inverzne Laplace-ove transformacije, ostale omenjene strukture pa so izredno redko analitično rešljive, in je za eksperimentiranje z njimi nujno potrebna simulacija. Diferencialne enačbe opisujejo sisteme v časovno zveznem prostoru.

V časovno diskretnem prostoru pa je neodvisna spremenljivka “čas” kvantizirana, kar

pomeni, da je interval opazovanja razdeljen na manjše časovne intervale. Odzivi sistema, za katere pravimo, da so vzorčeni, so definirani le v določenih trenutkih. Takšne diskretne sisteme opisujejo diferenčne enačbe, ki so diskretni ekvivalent diferencialnim enačbam. Diferenčno enačbo definiramo kot pravilo, ki izrazi vsak člen nekega niza s pomočjo predhodnih členov enega ali več nizov (n.pr. pri opisu sistemov predstavljata niza vhodne in izhodne signale).

Splošna linearna diferencialna enačba ima obliko:

$$y^{(n)} + a_1 y^{(n-1)} + \cdots + a_n y = b_0 u^{(m)} + b_1 u^{(m-1)} + \cdots + b_m u \tag{7.1}$$

in opisuje linearni časovno nespremenljivi zvezni sistem z enim vhodom *(u)* in enim izhodom *(y)*.

Pri tem oznaka $y^{(n)}$ oziroma $u^{(m)}$ pomeni n-ti odvod funkcij y po času in m-ti odvod funkcije u po času.

Ustrezna diferenčna enačba pa je:

$$y(k) + c_1 y(k-1) + \cdots + c_n y(k-n) = d_0 u(k-1) + \cdots + d_m u(k-m) \tag{7.2}$$

Druga, predvsem v teoriji vodenja zelo pomembna oblika matematičnega zapisa dinamičnega sistema, je tako imenovana *prenosna funkcija*, ki jo lahko jemljemo tudi kot skrajšani zapis diferencialne enačbe. Če na slednjo izvedemo Laplaceovo transformacijo ob ničnih začetnih pogojih, je prenosna funkcija G(s), ki je odvisna od kompleksne spremenljivke s, definirana kot razmerje Laplaceovega transforma izhoda Y(s) in Laplaceovega transforma vhoda U(s) in tako dobimo prenosno funkcijo v obliki:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{b_0 s^m + b_1 s^{m-1} + \cdots b_{m-1} s + b_m}{s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n} \tag{7.3}$$

Če je najvišji odvod v diferencialni enačbi ali največja zakasnitev v diferenčni enačbi ali najvišja potenca v imenovalcu prenosne funkcije enaka n, pravimo, da je sistem reda n. Prenosne funkcije so pomembne ne samo zaradi dejstva, da kompletno opisujejo vhodno-izhodno obnašanje sistema, temveč tudi zato, ker je z njimi možno izvajati različne algebrajske operacije, medtem ko to z diferencialnimi enačbami ni mogoče.

Na podoben način kot zvezno, dobimo tudi diskretno prenosno funkcijo, le da namesto Laplaceove uporabimo z-transformacijo:

$$H(z) = \frac{Y(z)}{U(z)} = \frac{d_0 + d_1 z^{-1} + \cdots + d_m z^{-m}}{1 + c_1 z^{-1} + \cdots + c_n z^{-n}} \tag{7.4}$$

Zvezno prenosno funkcijo je mogoče pretvoriti v diskretno in obratno. Najbolj znana postopka za prehod iz zvezne na diskretno prenosno funkcijo sta metodi stopničaste invariance in bilinearne transformacije.

Kot splošno uveljavljeni standardni zapis nekega sistema moramo omeniti tudi prostor stanj. Pri tem želimo obnašanje sistema prikazati v obliki sistema linearnih diferencialnih enačb prvega reda. V primerjavi s prenosno funkcijo, ki podaja le vhodno-izhodne relacije, je to zapis, ki vsebuje mnogo več informacij o sistemu (tudi njegovi notranjosti) in predstavlja osnovno orodje moderne teorije vodenja, saj je zelo primeren za računalniško obdelavo.

Sistem z enim vhodom in enim izhodom zapišemo z enačbama:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= f(\mathbf{x},u,t) \\ y(t) &= g(\mathbf{x},u,t) \end{aligned} \tag{7.5}$$

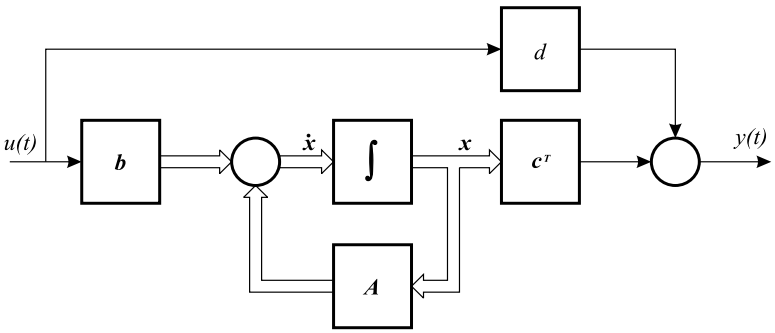
pri čemer prvo imenujemo enačbo stanj, drugo pa izhodno enačbo. Kot vidimo sta gornji enačbi zapisani za časovno spremenljivi sistem, pri čemer sta funkciji f in g v splošnem lahko tudi nelinearni. V primeru časovno nespremenljivega sistema se enačbi poenostavita v obliko:

$$\begin{aligned} \dot{\mathbf{x}} &= f(\mathbf{x},u) \\ y &= g(\mathbf{x},u) \end{aligned} \tag{7.6}$$

V primeru linearnega ali lineariziranega sistema pa dobimo zapis:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{b}u(t) \\ y(t) &= \mathbf{c}^T\mathbf{x}(t) + du(t) \end{aligned} \tag{7.7}$$

pri čemer ima vektor stanj **x** dimenzijo (nx1) in so **A**, **b** in **c** konstantna matrika in vektorja ustreznih dimenzij, d pa je skalar. Njihovo povezavo kaže Sl. 7.4.



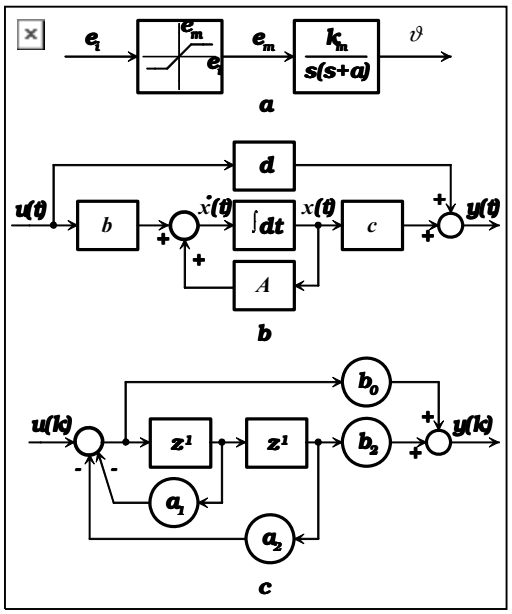
Sl. 7.4. Prikaz elementov zapisa v prostoru stanj

Diskretni zapis linearnega sistema v prostoru stanj pa je:

$$\begin{aligned} \mathbf{x}(k+1) &= \mathbf{F}\mathbf{x}(k) + \mathbf{g}u(k) \\ y(k) &= \mathbf{h}^T\mathbf{x}(k) + d_d u(k) \end{aligned} \tag{7.8}$$

Med parametri **A**, **b**, **c** in d ter **F**, **g**, **h** in d_d prav tako obstajajo transformacije, ki omogočajo prehod iz zveznega v diskretni prostor stanj in obratno.

Grafični prikaz parametričnih modelov predstavljajo bločni diagrami, ki smo jih omenili že v poglavju 3. Le-te sestavljajo bloki, ki pomenijo zvezo med izhodom in vhodom posameznega dela modela. Slednje povezujejo usmerjene (puščica) povezave, ki določajo smer poteka signala. V tovrstnih prikazih nastopajo še sumacijske točke in razcepišča signala. V bloke lahko vpisujemo prenosne funkcije, zapise v prostoru stanj pa tudi simbole različnih nelinearnosti. Prednosti grafičnega prikaza so strukturna jasnost in možnost učinkovitega poenostavljanja in preoblikovanja kompleksnih povezav s pomočjo algebre bločnih diagramov. Tipični bločni diagrami so na Sl. 7.5.



Sl. 7.5. Bločni diagrami:
a. nelinearnost in zvezna prenosna funkcija
b. zapis v prostoru stanj
c. diskretna realizacija prenosne funkcije

Navedimo še, da so podobne pretvorbe, kot smo jih omenili za primer zveznih in diskretnih modelov, definirane med vsemi oblikami parametričnih modelov.

Neparametrični modeli

Neparametrični modeli prikazujejo obnašanje sistemov s pomočjo odzivov (ali kakšnih drugih karakteristik) v časovnem ali frekvenčnem prostoru za zvezne in diskretne sisteme.

Ustrezne časovne odzive na standardne vhodne signale (stopnica, časovno naraščajoča funkcija, impulz, sinusoida, itd.) ali pa na operativne signale industrijskih naprav

dobimo iz meritev ali s pomočjo simulacije iz parametričnih modelov procesov. Obratna pot pa poteka preko metod identifikacije, ki jih bomo opisali kasneje.

Tudi frekvenčni odziv sistema, ki ga imenujemo frekvenčna karakteristika, lahko izmerimo ali pa ga izračunamo analitično iz prenosne funkcije. Frekvenčna karakteristika je lastnost sistema, ki pove kako se sistem v ustaljenem stanju odziva na sinusni vhodni signal. Posnamemo jo tako, da preko določenega področja spreminjamo frekvenco sinusnega signala na vhodu in merimo ustrezni izhodni signal v ustaljenem stanju.

Frekvenčna karakteristika sistema je definirana s kompleksnim izrazom:

$$G(j\omega) = |G(j\omega)| e^{j\Phi(\omega)} \tag{7.9}$$

kjer je $|G(j\omega)|$ absolutna vrednost, $\Phi(\omega)$ pa fazni kot. Podana sta z enačbama:

$$\begin{aligned} |G(j\omega)| &= \sqrt{[\text{realni del } G(j\omega)]^2 + [\text{imaginarni del } G(j\omega)]^2} \\ \Phi(\omega) &= \angle[G(j\omega)] = \arctan\left[\frac{\text{imaginarni del } G(j\omega)}{\text{realni del } G(j\omega)}\right] \end{aligned} \tag{7.10}$$

Frekvenčno karakteristiko dobimo tako, da v prenosno funkcijo G(s) namesto s vstavimo $j\omega$.

∇

Primer 7.1: *Frekvenčna karakteristika sistema 1. reda*

Za sistem 1. reda:

$$G(s) = \frac{K}{Ts+1} \tag{7.11}$$

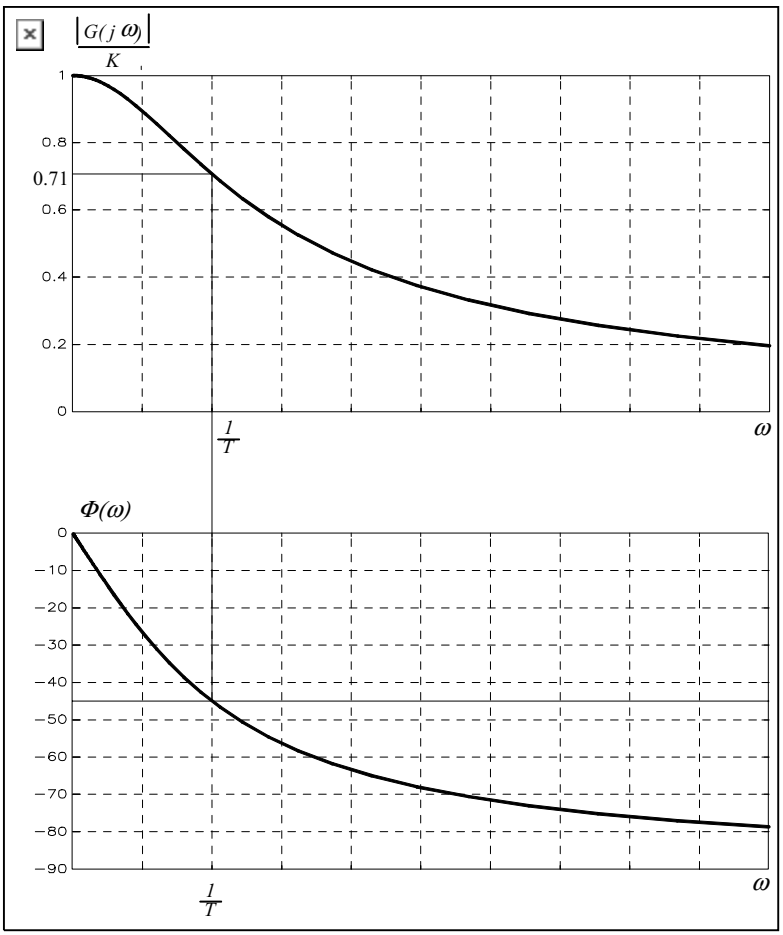
je frekvenčna karakteristika podana z izrazom:

$$G(j\omega) = \frac{K}{Tj\omega+1} \tag{7.12}$$

oziroma:

$$\begin{aligned} |G(j\omega)| &= \frac{K}{\sqrt{1+T^2\omega^2}} \\ \Phi(\omega) &= -\arctan(\omega T) \end{aligned} \tag{7.13}$$

Absolutno vrednost in fazni kot v odvisnosti od ω kaže Sl. 7.6, pri čemer frekvenco v točki $\frac{1}{T}$ imenujemo lomno frekvenco.



Sl. 7.6. Frekvenčna karakteristika sistema 1. reda.

Δ

Prenosne funkcije sistemov lahko podamo tudi v faktorizirani obliki, se pravi s pomočjo produktov ničel (korenov števca) in polov (korenov imenovalca) ter multiplikativne konstante. Ob upoštevanju $s=j\omega$ dobimo:

$$G(j\omega)=\frac{k(j\omega+z_1)\cdots(j\omega+z_m)}{j\omega^r(j\omega+p_1)\cdots(j\omega+p_n)} \tag{7.14}$$

Za primer

$$G(j\omega)=\frac{k(j\omega+z)}{j\omega(j\omega+p)} \tag{7.15}$$

pokažimo način izražave absolutne vrednosti in faznega kota:

$$G(j\omega) = \frac{k|j\omega + z|}{|j\omega||j\omega + p|}$$
$$\angle[G(j\omega)] = \angle[j\omega + z] - \angle[j\omega] - \angle[j\omega + p] \tag{7.16}$$

Frekvenčno karakteristiko dobimo z izbiro različnih točk na $j\omega$ osi ($0 \le \omega \le \infty$). Dve najbolj uveljavljeni predstavitvi frekvenčne karakteristike sta:

- Bodejev diagram in
- Polarni diagram.

Bodejev diagram podaja frekvenčno karakteristiko sistema z dvema diagramoma. V prvem je na ordinatni osi logaritem absolutne vrednosti, v drugem pa fazni kot. V obeh diagramih je na abscisni osi frekvenca v logaritmičnem merilu. Logaritem absolutne vrednosti izražamo v decibelih [dB] in ga določimo z izrazom:

$$L(\omega)[\mathrm{dB}] = 20\log_{10}|G(j\omega)| \tag{7.17}$$

Fazni kot $\Phi(\omega)$ pa podaja enačba (7.10).

Bistvena prednost logaritmičnega merila na ordinati je v tem, da se produkti znotraj faktoriziranega zapisa prenosne funkcije $|G(j\omega)|$ spremenijo v vsoto. Vsak tak člen nato predstavimo s t.i. asimptotskim potekom, ki določa absolutno vrednost za zelo nizke in za zelo visoke frekvence. Tudi fazno karakteristiko dobimo tako, da določimo fazne poteke posameznih komponent, ki so lahko:

- ojačenje k
- integrirni ali diferencirni člen $(j\omega)^{\mp 1}$
- člen 1. reda $(1 + j\omega T)^{\pm 1}$
- člen 2. reda $\left[1 + 2\xi\frac{j\omega}{\omega_n} + \left(\frac{j\omega}{\omega_n}\right)^2\right]^{\pm 1}$
- člen $e^{-j\omega T_m}$, ki opisuje mrtvi čas.

Nato le te seštejemo ter tako dobimo celotni Bodejev diagram obravnavane prenosne funkcije pri čemer natančni potek dobimo s tabeliranjem $L(\omega)$ in $\Phi(\omega)$ za izbrane frekvence.

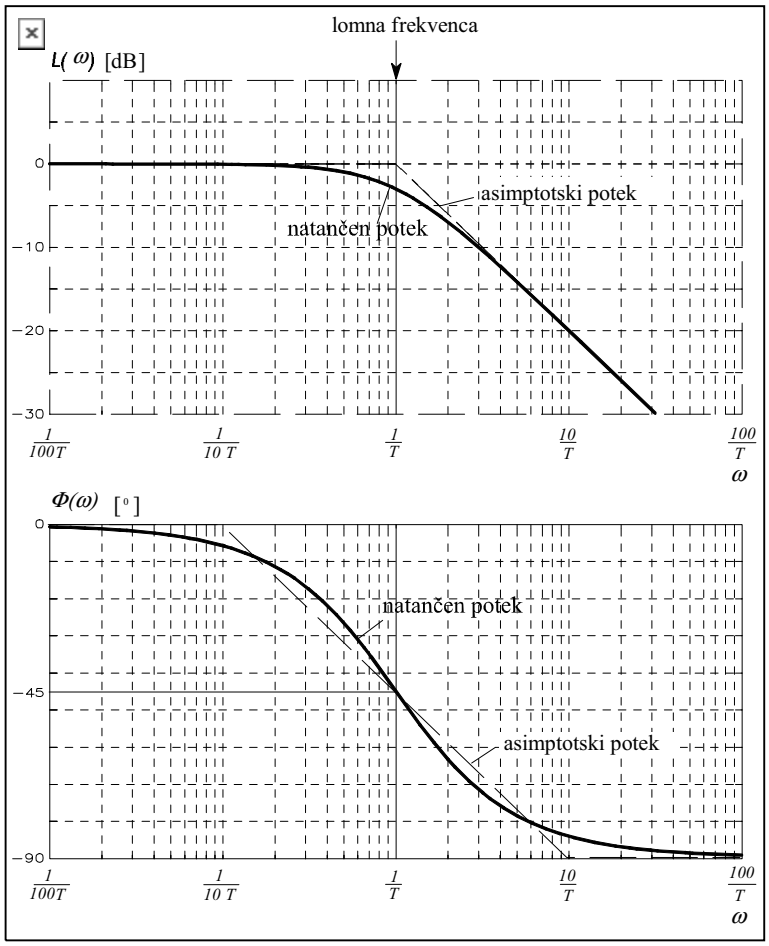
∇

Primer 7.2: *Bodejev diagram sistema 1. reda*

Bodejev diagram sistema 1. reda:

$$G(j\omega) = \frac{1}{1 + j\omega T} \tag{7.18}$$

vidimo na Sl. 7.7. Dobili smo ga tako, da smo na osnovi enačbe (7.18) izračunali $L(\omega)$ in $\Phi(\omega)$. Določimo pa ga lahko tudi eksperimentalno. Sistem vzbujamo s sinusnim signalom različnih frekvenc in merimo amplitudo in fazo izhodnega signala. Glede na to narišemo $L(\omega)$ in $\Phi(\omega)$. Iz tako dobljenega Bodejevega diagrama pa lahko identificiramo prenosno funkcijo sistema s prilagajanjem posameznih asimptot merjenim vrednostim in obratnim postopkom, kot ga uporabimo pri sestavljanju Bodejevega diagrama iz posameznih osnovnih členov.



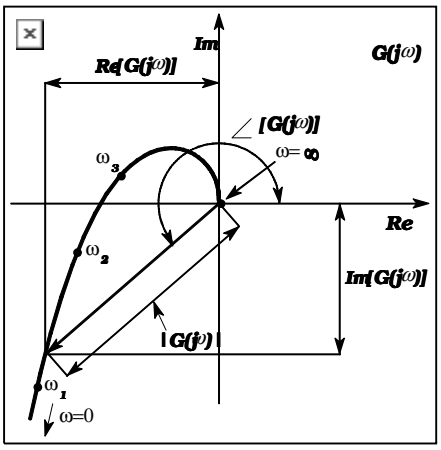
Sl. 7.7. Bodejev diagram člena 1. reda $G(j\omega) = \dfrac{1}{1 + j\omega T}$

Δ

Polarni diagram predstavlja frekvenčno karakteristiko $G(j\omega)$ v kompleksni ravnini. Vsaka točka je podana s polarnim zapisom kompleksorja $G(j\omega) = |G(j\omega)|e^{j\angle[G(j\omega)]}$ pri določeni frekvenci. Če se le ta spreminja od 0 do ∞, zariše kompleksor polarni diagram. Pozitivni fazni kot je definiran od pozitivne realne osi proti določenemu kompleksorju v obratni smeri urinega kazalca. Primer polarnega diagrama prikazuje Sl. 7.8.

Določitev polarnega diagrama je nekoliko zahtevnejši postopek kot npr. računanje Bodejevega diagrama, saj absolutne vrednosti v tem primeru ne moremo dobiti s seštevanjem prispevkov posameznih gradnikov. Zato se včasih posredno uporablja Bodejev diagram tudi za določitev polarnega diagrama. Običajno pa dandanes določamo tako polarni diagram kot tudi druge oblike frekvenčne karakteristike z ustreznimi računalniškimi paketi.

Omenimo še, da obstajajo predstavljene frekvenčne karakteristike tudi za diskretne sisteme.



Sl. 7.8. Polarni diagram

7.2.1.2 Principi modeliranja

K matematičnemu modeliranju lahko v glavnem pristopimo na tri načine.

Teoretično modeliranje

Bistvo tega pristopa je razčlenitev obravnavanega procesa na čim enostavnejše podsisteme. Povezave med slednjimi pa nato določimo na osnovi ravnotežnih enačb in raznih zakonov iz ustreznega področja (npr. Ohmov zakon, Kirchoffov zakon, Hookov zakon, itd.). Omenjene povezave med podsistemi so v naravoslovju in tehniki še nekako določljive, pri tako imenovanih mehkih znanostih (družboslovje, sociologija ter tudi biološke in medicinske vede) pa lahko predstavljajo resen problem. Ker je matematični model kvantitativno formalni zapis sistema, je lahko zelo kompleksen, kar pa mu jemlje

uporabnost. Zato se moramo največkrat odločiti za skrbno pretehtane in dobro argumentirane zanemaritve, poenostavitve in razne predpostavke, ki naj bi omogočale predstavitev sistema z navadnimi, linearnimi diferencialnimi enačbami čim nižjega reda. Poiskati moramo nek kompromis med natančnostjo in kompleksnostjo modela, pri čemer odločilno vlogo igra namen modeliranja. Teoretično modeliranje zahteva zelo poglobljeno in detajlno poznavanje modeliranega procesa, zato je sodelovanje problemskega strokovnjaka v postopku modeliranja nujno (problem ustvarjanja interdisciplinarne skupine). Ena od najvažnejših lastnosti teoretičnih modelov je, da jih lahko uporabimo tudi za podobne probleme ob ustrezni spremembi parametrov. Prav tako je pomembno, da lahko tovrstne modele uporabljamo že v fazi planiranja in načrtovanja nekega objekta in pa povsod tam, kjer meritve na procesu iz kakršnegakoli razloga niso možne.

Eksperimentalno modeliranje

Tovrstno modeliranje, ki ga imenujemo tudi identifikacija, temelji na izbiri vhodov in izhodov obravnavanega sistema in na meritvah teh signalov. Matematično strukturo in njene parametre želimo določiti tako, da bi bili (ob enakih vhodnih signalih) njeni časovni odzivi čim bliže merjenim. Zanimajo nas le vhodno-izhodne informacije, ne pa tudi mehanizmi, ki jih povzročajo. V nasprotju s teoretičnim modeliranjem so številne metode identifikacije dobro algoritmizirane, relativno enostavne in jasne ter dajejo kot rezultat največkrat linearne modele nizkega reda, ki upoštevajo vplive okolice. Uporabljamo jih lahko tudi na področjih, kjer je teoretično modeliranje problematično, zavedati pa se moramo, da iz njih ne dobimo nobene informacije o fizikalnem ozadju problema. Dobljeni modeli so zato uporabni le kot celota. Pojavljajo se težave z ločevanjem koristnega signala od merjenega, ki je moten s šumi, pa tudi z izbiro vzbujevalnih signalov, ki bi zagotovili dovolj informacije o modeliranem procesu. Problematično je predvsem dejstvo, da taki signali niso podobni vhodnim signalom pri normalnem delovanju procesa, in je zato v praksi največkrat težko dobiti dovoljenje za ustrezne eksperimente. Končno je potrebno poudariti, da identifikacije ne moremo uporabiti v primerih, ko ni na razpolago ustreznih meritev.

Kombinirano modeliranje

V praksi se največkrat uporablja kombinirano modeliranje, ki skuša izkoristiti dobre lastnosti obeh omenjenih pristopov. Pri tem strukturo modela v glavnem definiramo s pomočjo teoretičnega pristopa, medtem ko parametre ocenimo s pomočjo eksperimentalnega modeliranja.

7.2.1.3 Ciklični postopek modeliranja

Kot smo že omenili, je postopek modeliranja iterativnega (cikličnega) in interaktivnega značaja. Pristop lahko v splošnem razčlenimo na naslednje faze:

- postavitev problema;
- razvoj modela;

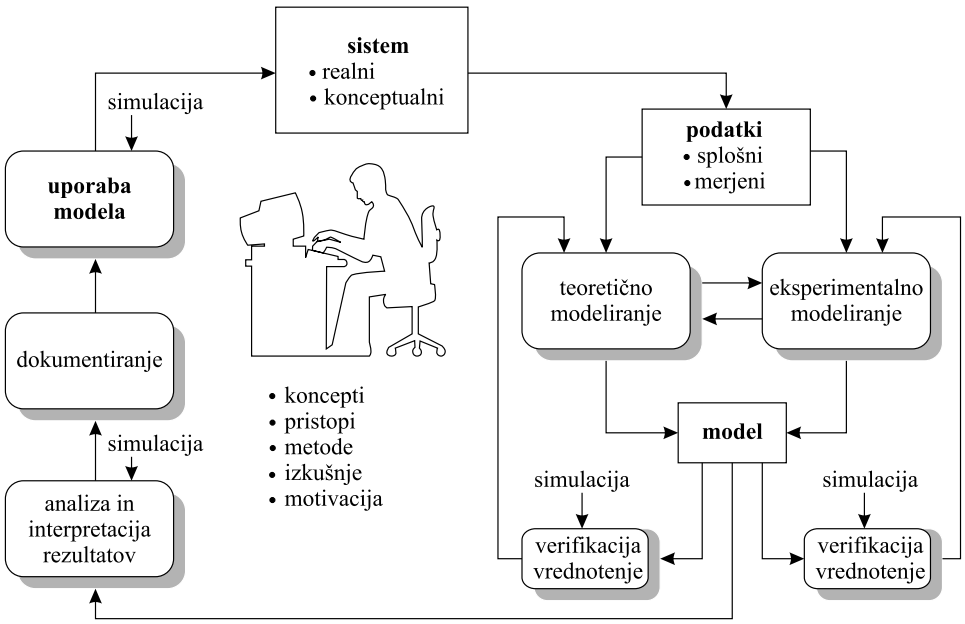
- vrednotenje modela;
- reševanje problema s pomočjo modela;
- testiranje in preverjanje rešitve;
- implementacija rešitve.

Vendar pa je bistvo modeliranja, ki je praktično v vseh fazah podprto z rezultati simulacije, v možnosti in največkrat tudi nujnosti ponavljanja posameznih faz ali skupin faz, dokler ne dosežemo zadovoljivega rezultata, ko je model dejansko uporaben in od njega dobimo, kar smo si zamislili. Grobo shemo cikličnega postopka modeliranja podaja Sl. 7.9.

Iz Sl. 7.9 je razvidno tesno prepletanje modeliranja in simulacije skozi celoten cikel, ki ima razvojni značaj, saj se začne pri različnih hipotezah in domnevah, nadaljuje z razvojem splošnega modela in njegovim poenostavljanjem ter konča z uporabo končnega modela in implementacijo tako dobljenih rezultatov. Pri tem mora biti izvršena transformacija kvalitativnih informacij v kvantitativne podatke. Vidimo, da je računalnik glavni pomočnik strokovnjaka, ki se ukvarja z modeliranjem. Računalnika ne uporabljamo samo za namene simulacije, temveč tudi za zbiranje podatkov, njihovo glajenje, za procesiranje signalov in identifikacijo, za testiranje hipotez in občutljivostno analizo, za vrednotenje in verifikacijo, kakor tudi za optimizacijo, za izvajanje matričnih operacij in za statistično obdelavo podatkov. Interaktivna uporaba računalnika in njegovih grafičnih zmožnosti pripomore k hitrejšemu, natančnejšemu in učinkovitejšemu modeliranju. Iz Sl. 7.9 sta razvidna tudi dva glavna dela cikla modeliranja. Prvi del zajema gradnjo modela, kar pomeni formalizacijo problema. Ta del je bolj znanstveno orientiran v smislu težnje po pridobivanju znanja o problemih, ki jih modeliramo, preko razvoja njihovih abstraktnih predstavitev (matematičnega modela). V nasprotju s prvim delom, ki je na desni strani Sl. 7.9, pa ima drugi del, na levi strani Sl. 7.9, bolj inženirski značaj. Z analizami in ustrezno interpretacijo rezultatov namreč ugotovimo, kako lahko realni sistem modificiramo ali dopolnimo v smislu izpolnjevanja naših ciljev (izpolnitev namena modeliranja). Struktura na Sl. 7.9 vsebuje mnoge povezave in zanke, kar ponovno poudarja iterativnost in prepletenost postopka modeliranja.

V procesu gradnje modela moramo najprej skrbno analizirati sistem, pri čemer pride do odločitev, ki so ključnega značaja za nadaljnje uspešno modeliranje:

- nameni modeliranja morajo biti podani kar najbolj jasno in nedvoumno;
- upoštevati moramo čimveč možnih omejitev;
- splošne predpostavke in zanemaritve morajo biti dobro argumentirane;
- ugotoviti in zagotoviti moramo možnosti za meritve in eksperimentiranje z modeliranim sistemom;
- izbrati moramo strategijo razvoja modela (metode zbiranja in analizo podatkov, ocenjevanje parametrov, tip modela, dosegljive računalniške kapacitete, uporabnost in fleksibilnost simulacijskih orodij, ki so na razpolago, splošnost dobljenih rešitev, možnosti dopolnjevanja modela, itd).



Sl. 7.9. Ciklični postopek modeliranja pri reševanju konkretnih problemov.

Kljub pomembnosti teh odločitev pa se moramo zavedati, da se prav vse lahko še spreminja v cikličnem postopku modeliranja.

Če smo na prej omenjeni način prišli do splošnega matematičnega modela (teoretično modeliranje), je možna odločitev o uporabi simulacije. Če analitična rešitev za matematični model obstaja, potem jo je vredno uporabiti, saj se moramo zavedati, da (za razliko od analitičnega reševanja) simulacija zahteva zelo veliko vhodnih podatkov, ki morajo biti podani na ustrezen način. Po drugi strani pa že najmanjša sprememba strukture modela (npr. dodatek nelinearnosti) pri analitičnem reševanju zahteva največkrat popolnoma drugačen pristop (če je sploh mogoč), pri čemer pa simulacijska strategija ostane nespremenjena. To kaže na fleksibilnost simulacije pri reševanju matematičnih modelov.

Naslednji korak je poenostavljanje modela, saj je ravno enostavnost osnoven kriterij za dobre modele. Pri tem seveda upoštevamo različne aproksimacije, ki jih moramo utemeljiti, pa tudi ustrezno dokumentirati, da tega pri vrednotenju modela ne pozabimo. Preveliko poenostavljanje modela vodi do izgube potrebne točnosti in splošnosti, medtem ko preveč detajlni modeli težijo k univerzalnosti in s tem k preveliki kompleksnosti, kar pa je tudi redkokdaj upravičeno. Zato je bolje razvijati enostavne modele, čeprav se lahko zgodi, da moramo za isti sistem zgraditi več različnih modelov, seveda za razne namene modeliranja.

Pred končno obliko formalnega modela, mora biti slednji ustrezno ovrednoten in verificiran. Nakažimo razlike med vrednotenjem in verifikacijo modela, ki predstavljata pomemben korak v razvoju uporabnih modelov. Verifikacija pomeni postopek

preverjanja ali model deluje tako, kot smo si zamislili (interni značaj), medtem ko vrednotenje ocenjuje, če se model obnaša zadovoljivo podobno modeliranemu procesu (eksterni značaj). Iz povedanega je jasno, da je postopek vrednotenja mnogo bolj problematičen kot so načini verifikacije modela, po drugi strani pa je model, ki ni ustrezno ovrednoten, praktično brez vrednosti, saj ne vemo, za katere situacije ga lahko upravičeno uporabljamo. Glede na omenjene razlike med obema pojmoma lahko še ugotovimo, da se ovrednotenju modela lahko le bolj ali manj približamo, ne moremo ga pa popolnoma doseči, saj tudi realni sistem ne more biti popolnoma znan in model zato ni natančna predstavitev sistema. Vrednotenje je bilo v preteklosti sploh deležno mnogo premalo pozornosti, delno tudi zato, ker zanj ne obstaja noben sistematični postopek ali celo algoritem. Čeprav je bilo razvitih mnogo uspešnih modelov, ne da bi bili ustrezno ovrednoteni, pa raziskovalci vedno bolj težijo ne k absolutnemu, vendar pa razumnemu vrednotenju v smislu opredelitve stopnje zaupanja v model, ki z določeno natančnostjo opisuje sistem. Tako je jasno, da je vrednotenje ena najbolj problematičnih faz v celotnem postopku razvoja modela, saj ob povedanem obstajajo samo še nekateri splošni nasveti kot npr., da modela ne smemo vrednotiti s pomočjo istih podatkov, na katerih je bil razvit, itd.

V ciklu razvoja modela se večkrat lahko pojavi tudi analiza občutljivosti parametrov, saj nas zanima, kateri najbolj vpliva na sistem. Te analize so zelo povezane tudi s trenutno zelo aktualnimi robustnostnimi aspekti.

Pomembne korake pri modeliranju predstavljata tudi interpretacija rezultatov, ki lahko odigra ključno vlogo pri uspešni ali neuspešni uporabi modela, in dokumentiranje, ki je tudi nujen integralni del procesa modeliranja, pa je žal često sploh popolnoma zanemarjeno.

∇

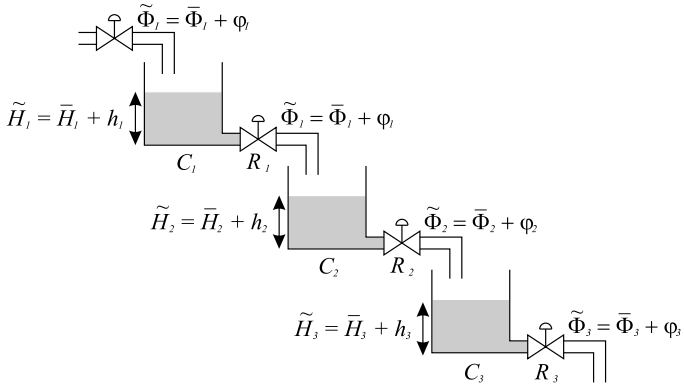
Primer 7.3: *Modeliranje hidravličnega sistema za namen načrtovanja kaskadne regulacije*

S pričujočim preprostim zgledom, želimo prikazati tipični proces teoretičnega modeliranja.

Obravnavani sistem je prikazan na Sl. 7.10.

Iz Sl. 7.10 je razvidno, da je sistem sestavljen iz treh neodvisnih shranjevalnikov tekočine. Vsak ima svojo hidravlično kapacitivnost (C_i), medtem ko izhodni pretoki ($\widetilde{\Phi}_i$) zavisijo od ustreznih hidravličnih upornosti (R_i), ki jih predstavljajo ventili, in od nivojev ($\widetilde{H}_i$) tekočine v shranjevalnikih. Namen modeliranja je razviti model, s pomočjo katerega bo možno načrtati kaskadno regulacijo sistema. (Kaskadna regulacija je podrobneje razložena v podpoglavju 7.3). Regulirna veličina je vhodni pretok ($\widetilde{\Phi}_I$), regulirana veličina je višina tekočine v tretjem shranjevalniku ($\widetilde{H}_3$), pomožna regulirana veličina pa je višina tekočine v prvem shranjevalniku ($\widetilde{H}_1$). Hidravlične

upornosti so nespremenljive.



Sl. 7.10. Hidravlični sistem s tremi shranjevalniki

Ker moramo pri teoretičnem modeliranju sistem razbiti na manjše podsisteme, se v našem primeru odločimo sistem razdeliti na posamezne shranjevalnike. Pri tem gre za različne količine tekočine in zato bo osnovni zakon masno ravnotežje, ki pravi:

vhodni pretoki - izhodni pretoki = sprememba količine shranjene tekočine

Za naš primer volumskih pretokov [m^3/s] tako velja za prvi shranjevalnik:

$$\tilde{\Phi}_I - \tilde{\Phi}_1 = \frac{dV_1}{dt} = S_1 \frac{d\tilde{H}_1}{dt} \tag{7.19}$$

kjer pomenijo $\tilde{V}_i$ volumne tekočine, S_i pa površine prečnih prerezov shranjevalnikov.

Osnovne ravnotežne enačbe pa le redko povezujejo tiste veličine, za katere smo v posameznem primeru zainteresirani (v našem primeru so to povezave med $\tilde{H}_1$ in $\tilde{\Phi}_i$), zato je prvi korak modeliranja posvečen iskanju ustreznih relacij, ki omenjene povezave opisujejo. Za turbulentni pretok preko zožitve v cevi (ventil) velja za ustaljeno stanje:

$$\tilde{\Phi}_1 = K_1 \sqrt{\tilde{H}_1} \tag{7.20}$$

kjer je K_1 koeficient [$m^{5/2}$/s].

Če vstavimo enačbo (7.20) v enačbo (7.19), dobimo nelinearno diferencialno enačbo prvega reda, ki vsebuje ustrezne povezave veličin kot jih zahteva namen modeliranja. Seveda pa bi bil nivo tekočine lahko reguliran tudi z izhodnim pretokom (regulacijski ventil na izhodu shranjevalnika), pri čemer bi v relacijo vključili tudi površino prereza odprtine ventila, ki je v našem primeru upoštevana v koeficientu K_1.

Definirajmo še hidravlično upornost (R) in kapacitivnost (C). Prva je definirana kot

sprememba višine tekočine v shranjevalniku, ki povzroči enotino spremembo pretoka:

$$R = \frac{d\widetilde{H}}{d\widetilde{\Phi}} \tag{7.21}$$

Ker glede na enačbo (7.20) velja:

$$d\widetilde{\Phi}_1 = \frac{K_1}{2\sqrt{\widetilde{H}_1}} d\widetilde{H}_1 \tag{7.22}$$

dobimo:

$$\frac{d\widetilde{H}_1}{d\widetilde{\Phi}_1} = \frac{2\sqrt{\widetilde{H}_1}}{K_1} = \frac{2\sqrt{\widetilde{H}_1}}{\widetilde{\Phi}_1}\sqrt{\widetilde{H}_1} = \frac{2\widetilde{H}_1}{\widetilde{\Phi}_1} \tag{7.23}$$

in končno za naš primer:

$$R_1 = \frac{2\widetilde{H}_1}{\widetilde{\Phi}_1} \tag{7.24}$$

Hidravlična upornost proti turbolentnem pretoku torej zavisi od nivoja tekočine in izhodnega pretoka in jo za majhne spremembe obeh veličin lahko jemljemo kot konstantno.

Hidravlična kapacitivnost je definirana kot sprememba količine shranjene tekočine, ki je potrebna, da povzroči enotino spremembo potenciala (višina tekočine). Zato velja (glej enačbo (7.19)):

$$\frac{d\widetilde{V}}{d\widetilde{H}} = S = \mathrm{C} \qquad \text{in v našem primeru} \quad C_1 = S_1 \tag{7.25}$$

Pri tem moramo ločiti med kapacitivnostjo [m^2] in kapaciteto [m^3] shranjevalnika. Če je površina prereza shranjevalnika konstantna je tudi hidravlična kapacitivnost konstantna za vsak nivo tekočine.

Sedaj se moramo odločiti, ali bomo model uporabili v nelinearni obliki ali pa je upravičeno izvesti postopek linearizacije. V našem primeru bomo model rabili v postopku načrtovanja regulatorja, ki je relativno robusten, pri čemer tudi ne pričakujemo velikih odstopanj od delovne točke sistema. Zato je seveda linarni model zelo uporaben.

Da bi dobili linearni približek nelinearnemu modelu, predvidimo le minimalna odstopnja od neke delovne točke. V našem primeru je le-ta v ustaljenem stanju $(\overline{\Phi}_1, \overline{H}_1, \overline{\Phi}_1)$ z odmiki φ_1, h_1 in φ_1. Nelinearnost razvijemo v Taylorjevo vrsto za določeno delovno točko in zanemarimo člene višjega reda. V našem primeru dobimo z upoštevanjem enačb (7.19), (7.20) in (7.25) nelinearno diferencialno enačbo:

$$C_1 \frac{d\widetilde{H}_1}{dt} = \widetilde{\Phi}_1 - K_1\sqrt{\widetilde{H}_1} \tag{7.26}$$

Definirajmo:

$$\frac{d\widetilde{H}_1}{dt} = f(\widetilde{H}_1, \widetilde{\Phi}_1) = \frac{\widetilde{\Phi}_1}{C_1} - \frac{K_1\sqrt{\widetilde{H}_1}}{C_1} \tag{7.27}$$

Omenjeni način linearizacije narekuje relacijo:

$$\frac{d\widetilde{H}_1}{dt} - f(\overline{H}_1, \overline{\Phi}_1) = \frac{\partial f}{\partial \widetilde{H}_1}(\widetilde{H}_1 - \overline{H}_1) + \frac{\partial f}{\partial \widetilde{\Phi}_1}(\widetilde{\Phi}_1 - \overline{\Phi}_1) \tag{7.28}$$

V delovni točki ($\overline{H}_1, \overline{\Phi}_1$) velja $f(\overline{H}_1, \overline{\Phi}_1) = 0$ in:

$$\left.\frac{\partial f}{\partial \widetilde{H}_1}\right|_{\widetilde{H}_1=\overline{H}_1, \widetilde{\Phi}_1=\overline{\Phi}_1} = -\frac{K_1}{2C_1\sqrt{\overline{H}_1}} = \frac{\overline{\Phi}_1}{\sqrt{\overline{H}_1}}\frac{1}{2C_1\sqrt{\overline{H}_1}} = -\frac{\overline{\Phi}_1}{2C_1\overline{H}_1} = -\frac{1}{R_1C_1} \tag{7.29}$$

$$\left.\frac{\partial f}{\partial \widetilde{\Phi}_1}\right|_{\widetilde{H}_1=\overline{H}_1, \widetilde{\Phi}_1=\overline{\Phi}_1} = -\frac{1}{C_1} \tag{7.30}$$

Tako dobimo enačbo (7.28) v obliki:

$$\frac{d\widetilde{H}_1}{dt} = -\frac{1}{R_1C_1}(\widetilde{H}_1 - \overline{H}_1) + \frac{1}{C_1}(\widetilde{\Phi}_1 - \overline{\Phi}_1) \tag{7.31}$$

in glede na Sl. 7.10:

$$\frac{dh_1}{dt} = -\frac{1}{R_1C_1}h_1 + \frac{1}{C_1}\varphi_1 \tag{7.32}$$

ali:

$$R_1C_1\frac{dh_1}{dt} + h_1 = R_1\varphi_1 \tag{7.33}$$

ki predstavlja linearizirano obliko enačbe (7.26), pri čemer je člen R_1C_1 njena časovna konstanta. Ker je manipulacija z diferencialnimi enačbami težavnejša kot s prenosnimi funkcijami, izrazimo enačbo (7.28) v tej obliki:

$$(R_1C_1s + 1)H_1(s) = R_1\phi_1(s) \tag{7.34}$$

in:

$$\frac{H_1(s)}{\phi_I(s)} = \frac{R_1}{R_1C_1s+1} \tag{7.35}$$

Ker so v našem primeru vsi podsistemi podobni opisanemu, bomo ustrezni matematični model razvili na podlagi prenosne funkcije (7.35). Glede na strukturo kaskadne regulacije, katere načrtovanje predstavlja namen modeliranja, moramo zapisati prenosni funkciji $\frac{H_1}{\phi_I}$ in $\frac{H_3}{H_1}$. Po analogiji lahko zapišemo naslednje enačbe:

$$\frac{H_1}{\phi_I} = \frac{R_1}{R_1C_1s+1} = G_{p2}(s) \tag{7.36}$$

$$\frac{\phi_1}{H_1} = \frac{1}{R_1} \tag{7.37}$$

$$\frac{H_2}{\phi_1} = \frac{R_2}{R_2C_2s+1} \tag{7.38}$$

$$\frac{\phi_2}{H_2} = \frac{1}{R_2} \tag{7.39}$$

$$\frac{H_3}{\phi_2} = \frac{R_3}{R_3C_3s+1} \tag{7.40}$$

Iz enačb (7.37) in (7.39) sedaj izrazimo:

$$\phi_I = \frac{H_I}{R_I} \quad in \quad \phi_2 = \frac{H_2}{R_2} \tag{7.41}$$

ju vnesemo v enačbi (7.38) in (7.40) ter dobimo prenosni funkciji:

$$\frac{H_2}{H_1} = \frac{R_2}{R_1(R_2C_2s+1)} \quad \text{in} \quad \frac{H_3}{H_2} = \frac{R_3}{R_2(R_3C_3s+1)} \tag{7.42}$$

katerih zmnožek daje:

$$\frac{H_3}{H_1} = \frac{R_3}{R_1(R_2C_2s+1)(R_3C_3s+1)} = G_{p1}(s) \tag{7.43}$$

Tako sta za določene vrednosti R_i in C_i prenosni funkciji v enačbah (7.36) in (7.43) pripravljeni za simulacijo in za uporabo glede na namen modeliranja.

Čeprav obravnavani primer vsebuje večino tipičnih korakov procesa modeliranja, se moramo zavedati, da je področje izrazito problemsko orientirano, in so zato pristopi zelo raznoliki. Le-tem se je potrebno ustrezno prilagoditi.

Da bi ilustrirali obravnavane zapise in prikaze modelov, izračunajmo prenosno funkcijo (iz enačb (7.36) in (7.43)):

$$\frac{H_1}{\phi_I}\frac{H_3}{H_1}=\frac{R_3}{(R_1C_1s+1)(R_2C_2s+1)(R_3C_3s+1)} \tag{7.44}$$

Za izbrane vrednosti $C_1=C_2=C_3=1[m^2]$ in $R_1=1$, $R_2=2$, $R_3=3[min/m^3]$ dobimo:

$$\frac{H_3}{\phi_I}=\frac{3}{(s+1)(2s+1)(3s+1)}=\frac{0.5}{s^3+1.83s^2+s+0.167} \tag{7.45}$$

Osnovna diferencialna enačba je tako:

$$\frac{d^3H_3}{dt^3}+1{,}83\frac{d^2H_3}{dt^2}+\frac{dH_3}{dt}+0{,}167H_3=0{,}5\varphi_I \tag{7.46}$$

Prehod na zapis v prostoru stanj pa izvedemo z naslednjo izbiro:

$$\begin{aligned} x_1 &= y \\ \varphi_I &= u \end{aligned} \tag{7.47}$$

ter

$$\begin{aligned} x_1 &= H_3 \\ x_2 &= \dot{H}_3 \\ x_3 &= \ddot{H}_3 \end{aligned} \tag{7.48}$$

pri čemer so s pikami označeni časovni odvodi H_3. Iz tega sledi:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= x_3 \\ \dot{x}_3 &= -1{,}83x_3 - x_2 - 0{,}167x_1 + 0{,}5u \end{aligned} \tag{7.49}$$

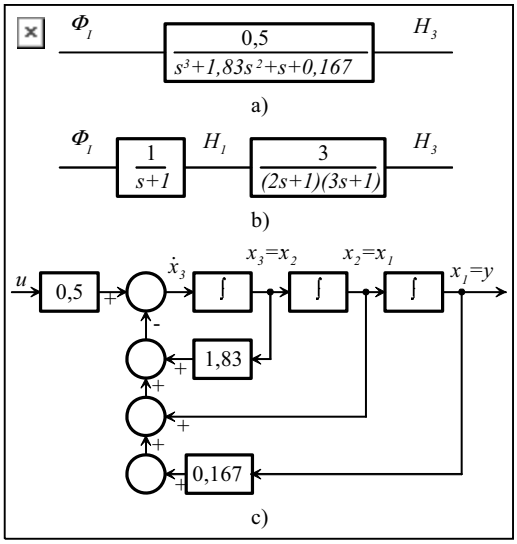
in v vektorsko matrični obliki:

$$\begin{aligned} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1{,}83 & -1 & -0{,}167 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0{,}5 \end{bmatrix}u \\ y &= \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \end{aligned} \tag{7.50}$$

Prehod iz zvezne (enačba (7.45)) na diskretno prenosno funkcijo izvedemo po metodi stopničaste invariance s časom vzorčenja T=1min in dobimo.

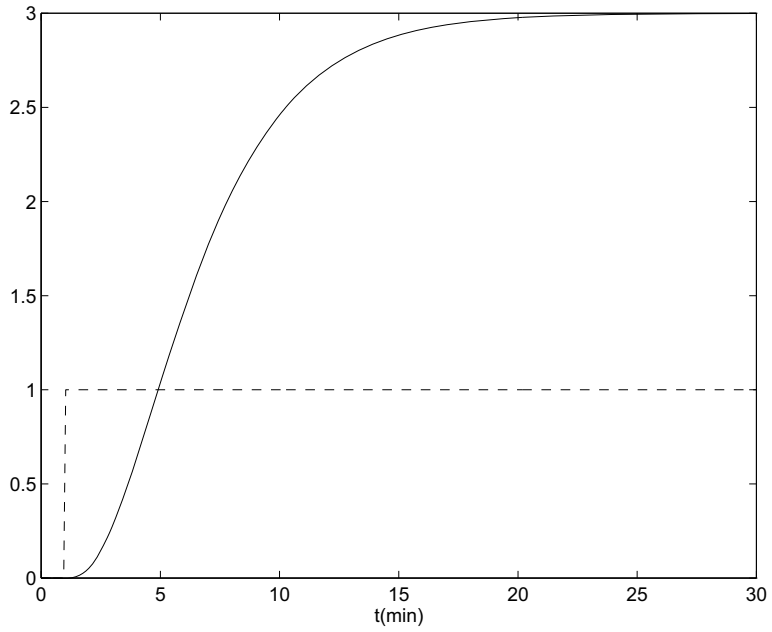
$$\frac{H_3(z)}{\phi_1(z)} = \frac{0{,}0543z^2 + 0{,}1369z + 0{,}021}{z^3 - 1{,}6906z^2 + 0{,}9213z - 0{,}1599} \tag{7.51}$$

Nekaj možnih oblik bločnih diagramov za obravnavani proces prikazuje Sl. 7.11.

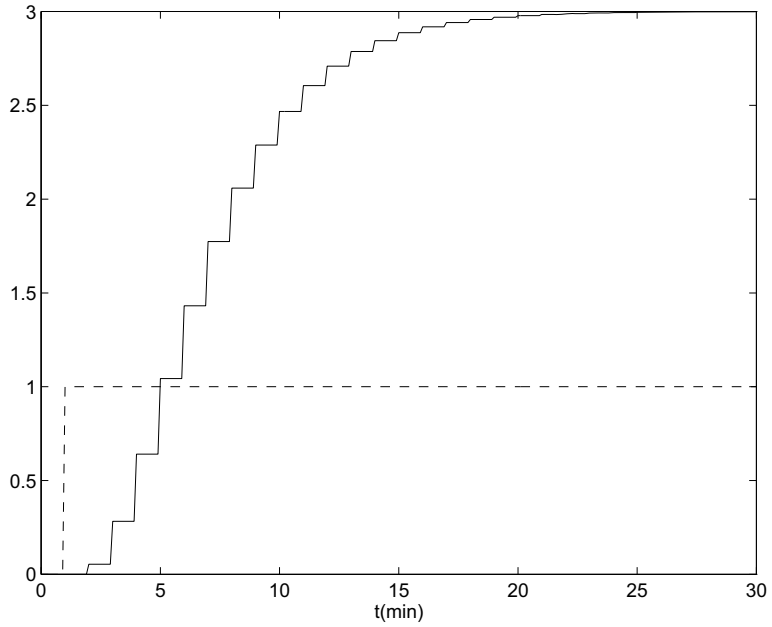


Sl. 7.11. Nekaj možnih oblik bločnih diagramov za obravnavani sistem
a) zvezna prenosna funkcija (en.7.45))
b) razgrajeni sistem pripravljen za kaskadno regulacijo (en. (7.36) in (7.43))
c) bločni diagram zapisa v prostoru stanj (en.(7.49))

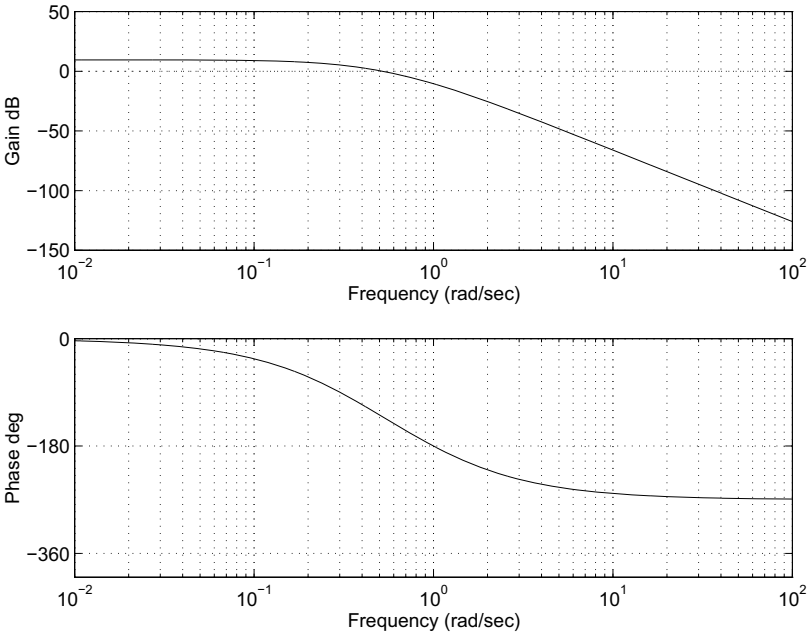
Sl. 7.12 do Sl. 7.15 pa kažejo časovni odziv procesa na enotino stopnico za zvezni in diskretni model ter Bodejev in polarni diagram.



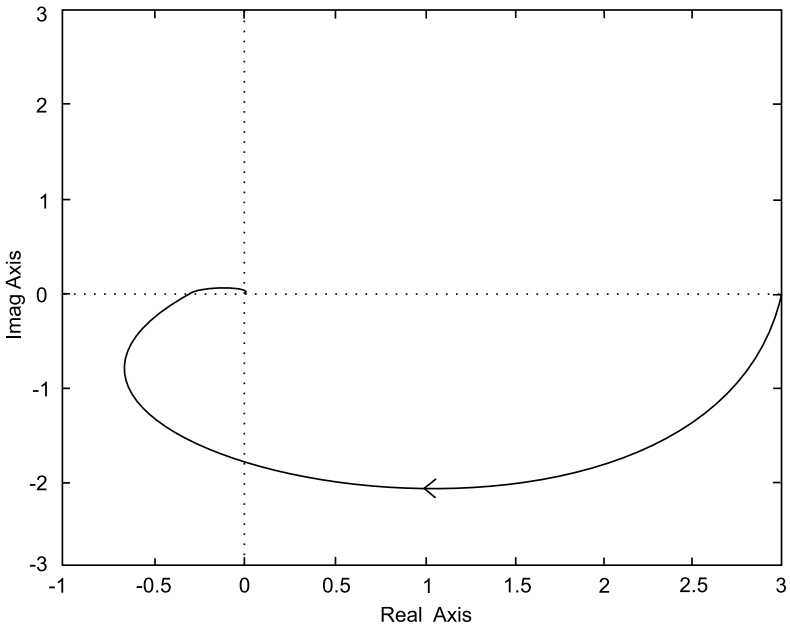
Sl. 7.12. Časovni odziv zveznega modela procesa (en. (7.45)) na enotino stopnico



Sl. 7.13. Časovni odziv diskretnega modela procesa (en. (7.51)) na enotino stopnico



Sl. 7.14. Bodejev diagram procesa iz enačbe (7.45)



Sl. 7.15. Polarni diagram procesa iz enačbe (7.45).

Δ

7.2.1.4 Identifikacija

Področje identifikacije dinamičnih sistemov je nekako od leta 1965 prerastlo v samostojno disciplino na področju vodenja procesov. S povezavo sistemske teorije, teorije signalov, merilne tehnike, modeliranja in računalniške simulacije so nastale številne dobro algoritmizirane in računalniško podprte metode za analizo dinamičnega obnašanja sistemov na osnovi merjenih vhodno-izhodnih podatkov.

Pojem identifikacije in s tem povezane naloge jedrnato opisujejo naslednje ugotovitve:

- **identifikacija procesa je eksperimentalna določitev časovnega obnašanja sistemov;**
- z uporabo merjenih signalov ponazorimo časovno obnašanje procesa znotraj nekega razreda matematičnih modelov;
- razlike med odzivi dejanskega procesa in njegovega matematičnega modela naj bodo čim manjše.

Glede na razširjenost in praktično uporabnost identifikacijskih metod (Matko, 1992), omenjenih že v podpoglavju o eksperimentalnem modeliranju, podajmo najprej korake, ki jih postopek identifikacije zahteva. Ti koraki so:

1. *Definicija namena uporabe*

2. *Zbiranje a-priori znanj*
   Le-to temelji v bistvu na dobrem poznavanju procesa in fizikalnih zakonitosti, ki ga karakterizirajo, na izkušnjah pri preteklem delu s procesom oziroma s sorodnimi procesi in na zbiranju statističnih podatkov o signalih (predvsem motilnih). Pri tem je potrebno odgovoriti predvsem na vprašanje glede strukture modela, njegove linearnosti in časovne (ne)spremenljivosti. Tako zbrane informacije so tudi bistvene pri izbiri identifikacijske metode.

3. *Načrtovanje meritev*
   V tem okviru je potrebno:

   - izbrati vhodne signale, njihovo vrsto (obratovalne, testne), njihovo obliko, amplitudo in frekvenčne spektre;
   - določiti čas vzorčenja;
   - določiti merilni čas;
   - določiti ali bo potekala meritev v odprti ali zaprti regulacijski zanki;
   - odločiti se za nesprotno ali sprotno identifikacijo;
   - izbrati aparaturno opremo;
   - določiti način filtriranja motenj.

4. *Izvedba meritev*

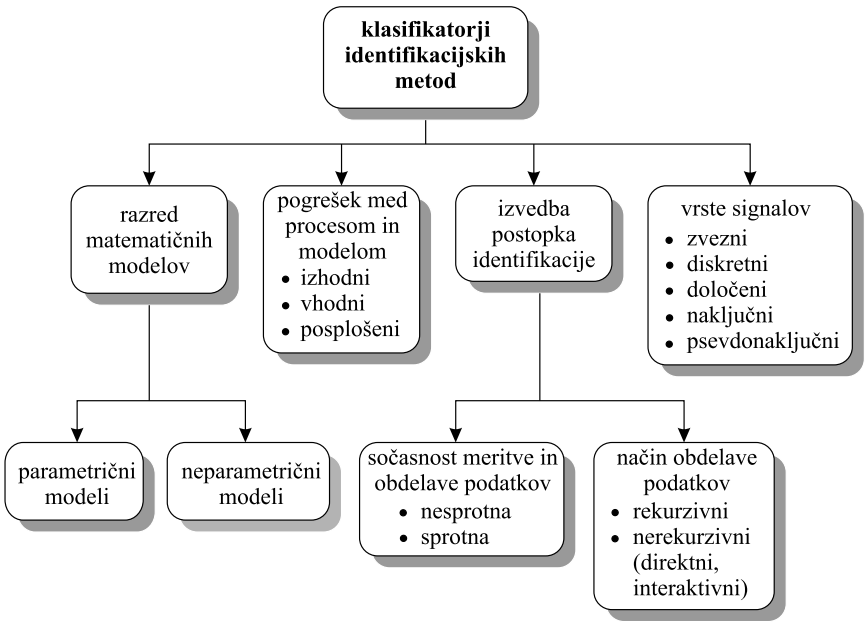
5. *Vrednotenje meritev*

6. *Preizkus identificiranega modela*

Pri izvedbi identifikacijskega postopka pa moramo upoštevati tudi nekatere omejitve kot so npr.:

- merjeni signali vsebujejo motnje;
- čas, ki ga imamo na razpolago za izvedbo identifikacijskega postopka, je omejen;
- velikosti vhodnega in izhodnega signala procesa sta omejeni;
- oblika vhodnega signala je omejena z uporabljenimi aktuatorji.

Identifikacijskih metod je seveda mnogo preveč, da bi lahko podali njihovo celovito klasifikacijo. Da bi ilustrirali široko paleto pristopov, podajamo na Sl. 7.16 različne vrste klasifikatorjev.



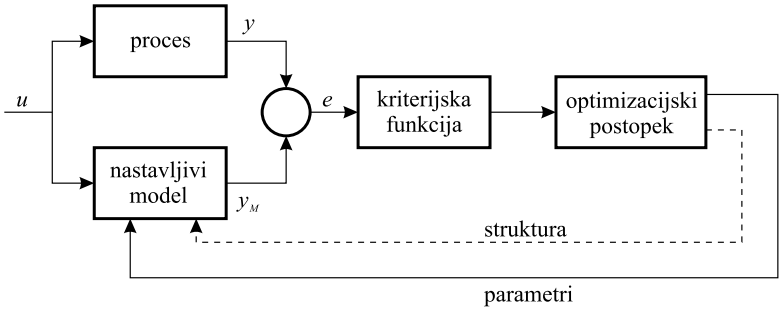
Sl. 7.16. Klasifikatorji, po katerih lahko razvrstimo različne metode identifikacije

Povejmo še, da med najbolj znane metode, ki temeljijo na parametričnih modelih spadajo metoda najmanjših kvadratov, metoda največje verjetnosti in metoda pomožnih spremenljivk. Med neparametričnimi metodami pa so znane Fourierova analiza, analiza frekvenčnega odziva, korelacijska analiza in spektralna analiza.

Končno si na kratko oglejmo še dva pogosto uporabljena parametrična pristopa k identifikaciji procesov (ocenjevanju parametrov), ki jasno kažeta osnovne ideje postopka.

Identifikacija s prilagajanjem modela

Pristop, ki ga včasih imenujemo tudi prilagajanje krivulj (curve fitting), teži odziv modela kar najbolj prilagoditi merjenemu izhodu procesa. Metoda je uveljavljena predvsem pri simulacijsko usmerjenih uporabnikih (tudi netehniške vede), ki jo uporabljajo pri določanju strukture (v manjši meri), predvsem pa parametrov zveznih modelov. Njena poglavitna prednost je, da ni omejena na linearne probleme ocenjevanja parametrov. Bistvo metode je v minimizaciji kriterijske funkcije, ki temelji na pogrešku (razliki) med izhodoma procesa in modela (Sl. 7.17).



Sl. 7.17. Princip identifikacije s prilagajanjem modela

Pri največ uporabljani nesprotni različici te metode imamo podatke o vhodu in izhodu procesa posnete na datoteki, ustrezen simulacijski podprogram pa pri danih začetnih vrednostih parametrov nastavljivega modela izračuna vrednost kriterijske funkcije. S poskušanjem ali pa s pomočjo optimizacijskega podprograma se iterativno bližamo minimumu kriterijske funkcije, pri čemer se za vsak niz parametrov izračuna nova vrednost kriterijske funkcije. Kljub zelo številnim možnim optimizacijskim algoritmom pa se ne moremo v celoti znebiti lastnosti optimizacije (problem definicije kriterijske funkcije, ki bi upoštevala vse naše želje, problem lokalnih minimumov, problem začetnih pogojev, problem konvergence, problem časa izračunavanja itd.).

Metoda najmanjših kvadratov

Metodo najmanjših kvadratov je prvi uporabil Gauss leta 1795 in sicer pri reševanju astronomskega problema. V svojem bistvu predstavlja ta metoda reševanje predoločenega sistema enačb. Tipičen problem, ki ga lahko rešimo je ocenitev parametrov neznanega procesa pri opazovanju merljivih izhodov, ki pa so moteni s šumom. Te parametre ocenimo tako, da uporabimo matematični model procesa in minimiziramo kvadrat razlike med izhodom procesa in modela (to razliko imenujemo tudi pogrešek). Rešitev problema je enostavna, če je matematični model procesa linearen v parametrih, ki jih ocenjujemo.

Običajno ocenjujemo več neznanih parametrov hkrati. Tipični primeri uporabe so t.i. regresijske krivulje. Regresijska krivulja pomeni najboljšo krivuljo, ki jo lahko potegnemo "skozi" merjene točke, pri čemer pojem "najboljši v smislu najmanjših

kvadratov" pojasnimo tako, da je vsota kvadratov odstopanj meritev od regresijske krivulje najmanjša. Matematični model regresijske krivulje reda q zapišemo:

$$y_0 = K_0 + K_1 u + K_2 u^2 + \ldots + K_q u^q \tag{7.52}$$

pri čemer so vsa opazovanja y in u mišljena ob istem času. S stališča sistemske teorije je proces (7.52) statičen, saj izhod y zavisi le od vhoda u v istem trenutku, ne pa od njegovih preteklih vrednosti.

En. (7.52) lahko zapišemo tudi v naslednji obliki:

$$y_0(k) = K_0 + K_1 u(k) + K_2 u^2(k) + \ldots + K_q u^q(k) \tag{7.53}$$

Enačbi (7.52) in (7.53) sta sicer nelinearni (glede na u), vendar pa sta linearni glede na parametre ocenjevanja K_i. Za ocenitev $q+1$ parametov K_i $i=0,1,2,\ldots,q$ potrebujemo vsaj $q+1$ opazovanj v različnih trenutkih (in seveda pri različnih vrednostih u). Naša opazovanja so seveda motena s šumom (n), zato lahko opazujemo le merljivi izhod:

$$y(k) = y_0(k) + n(k) \tag{7.54}$$

ne pa nemotene veličine $y_0(k)$. Zato uporabimo v praksi običajno mnogo več opazovanj kot je število parametrov, ki jih ocenjujemo. Vsako opazovanje predstavlja eno enačbo, sistem enačb pa rešimo v smislu najmanjših kvadratov. Za nastavitev enačb potrebujemo še enačbo modela (veličine so označene s strešico), ki je v našem primeru:

$$\hat{y}(k) = \hat{K}_0 + \hat{K}_1 u(k) + \hat{K}_2 u^2(k) + \ldots + \hat{K}_q u^q(k) \tag{7.55}$$

ter enačbo pogreška:

$$e(k) = y(k) - \hat{y}(k) \tag{7.56}$$

Sistem enačb pogreška za N opazovanj dobimo z upoštevanjem en. (7.55) in (7.56) in ga lahko zapišemo v vektorsko matrični obliki:

$$\mathbf{y} - \mathbf{U}\hat{\mathbf{k}} = \mathbf{e} \tag{7.57}$$

pri čemer so:

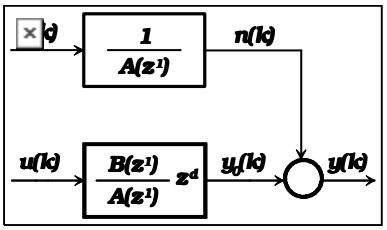
$$\mathbf{U} = \begin{bmatrix} 1 & u(0) & u^2(0) & \ldots & u^q(0) \\ 1 & u(1) & u^2(1) & \ldots & u^q(1) \\ & & \vdots & \ddots & \\ 1 & u(N-1) & u^2(N-1) & \ldots & u^q(N-1) \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} y(0) \\ y(1) \\ \vdots \\ y(N-1) \end{bmatrix} \quad \mathbf{e} = \begin{bmatrix} e(0) \\ e(1) \\ \vdots \\ e(N-1) \end{bmatrix} \quad \hat{\mathbf{k}} = \begin{bmatrix} \hat{K}_0 \\ \hat{K}_1 \\ \vdots \\ \hat{K}_q \end{bmatrix} \tag{7.58}$$

Če odvod kriterijske funkcije $(\mathrm{V} = \mathbf{e}^T\mathbf{e})$ po parametrih izenačimo z nič, dobimo optimalno oceno vektorja $\hat{\mathbf{k}}$ v smislu najmanjših kvadratov v obliki:

$$\hat{\mathbf{k}} = \left[\mathbf{U}^T\mathbf{U}\right]^{-1}\mathbf{U}^T\mathbf{y} \tag{7.59}$$

in ker je drugi odvod V po $\hat{\mathbf{k}}$ pozitiven, je izračunani optimum minimum.

Za dinamične sisteme, ki jih običajno, pri uporabi metode najmanjših kvadratov, podamo v diskretni obliki (Sl. 7.18), pa so, v nasprotju z omenjenim nelinearnim statičnim primerom, koeficienti v sistemu enačb dobljeni v večih zaporednih časovnih trenutkih.



Sl. 7.18. Moteni proces.

Vektorsko matrični zapis pogreška je :

$$\mathbf{y} - \mathbf{\Psi}\hat{\boldsymbol{\vartheta}} = \mathbf{e} \tag{7.60}$$

pri čemer je:

$$\hat{\vartheta}^T = \left[\hat{a}_1, \ldots, \hat{a}_n, \hat{b}_1, \ldots, \hat{b}_n\right] \tag{7.61}$$

vektor ocenjevanih parametrov diskretnega zapisa modela, vektorja **y** in **e** pa sta:

$$\mathbf{y} = \begin{bmatrix} y(k-N+1) \\ y(k-N+2) \\ \vdots \\ y(k-1) \\ y(k) \end{bmatrix} \quad \mathbf{e} = \begin{bmatrix} e(k-N+1) \\ e(k-N+2) \\ \vdots \\ e(k-1) \\ e(k) \end{bmatrix} \tag{7.62}$$

Pri tem je matrika

$$\mathbf{\Psi} = \left[\mathbf{\Psi}_1, \mathbf{\Psi}_2\right] \tag{7.63}$$

sestavljena iz matrik:

$$\mathbf{\Psi_1} = \begin{bmatrix} -y(k-N) & -y(k-N-1) & \dots & -y(k-N-n+1) \\ -y(k-N+1) & -y(k-N) & \dots & -y(k-N-n+2) \\ \vdots & \vdots & \ddots & \vdots \\ -y(k-2) & -y(k-3) & \dots & -y(k-n-1) \\ -y(k-1) & -y(k-2) & \dots & -y(k-n) \end{bmatrix} \tag{7.64}$$

in

$$\mathbf{\Psi_2} = \begin{bmatrix} u(k-N-d) & u(k-N-d-1) & \dots & u(k-N-n-d+1) \\ u(k-N-d+1) & u(k-N-d) & \dots & u(k-N-n-d+2) \\ \vdots & \vdots & \ddots & \vdots \\ u(k-d-2) & u(k-d-3) & \dots & u(k-d-n-1) \\ u(k-d-1) & u(k-d-2) & \dots & u(k-d-n) \end{bmatrix} \tag{7.65}$$

Ocena parmetrov $\hat{\boldsymbol{\vartheta}}$ v smislu najmanjših kvadratov je tako:

$$\hat{\boldsymbol{\vartheta}} = \left[\mathbf{\Psi}^{\mathrm{T}}\mathbf{\Psi}\right]^{-1}\mathbf{\Psi}^{\mathrm{T}}\mathbf{y} \tag{7.66}$$

pri čemer mora biti število opazovanj N v praksi mnogo večje kot dvakratno število parametrov, ki jih ocenjujemo.

Metoda najmanjših kvadratov ima mnogo izpeljank in obstaja tudi v rekurzivni obliki (novo oceno izračunamo kot vsoto stare ocene in s korekturnim faktorjem pomnožene razlike med novo meritvijo in napovedjo meritve na osnovi parametrov zadnjega koraka).

Pri praktični uporabi identifikacije se pojavljajo problemi pristranskosti in konvergence, numerični problemi ter vprašanje identifikabilnosti, zelo pomembna pa je tudi izbira vhodnega signala, časa vzorčenja in strukture modela.

∇

Primer 7.4: *Identifikacija modela hidravličnega procesa*

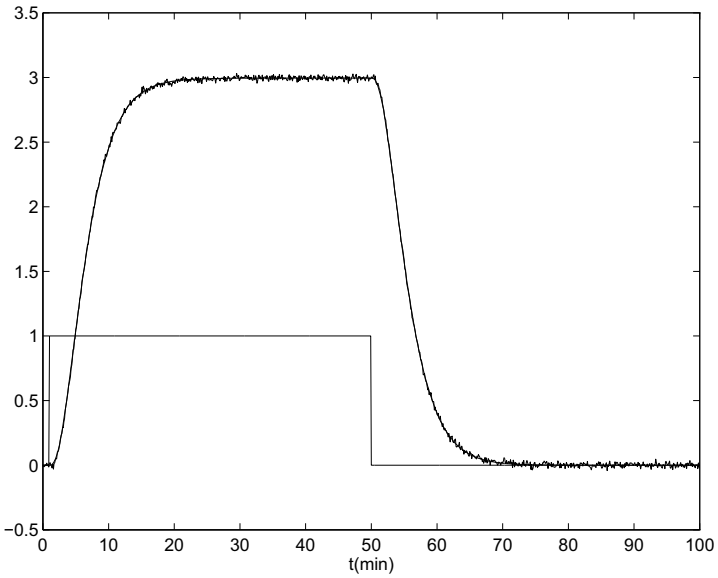
Oglejmo si uporabo obeh omenjenih pristopov k identifikaciji procesov, na primeru modela hidravličnega procesa s tremi shranjevalniki iz primera 7.3.

Metoda ocenjevanja parametrov s pomočjo prilagajanja modela (glej Sl. 7.17) daje ob podani strukturi modela in kriterijski funkciji v obliki integrala kvadrata pogreška naslednjo prenosno funkcijo:

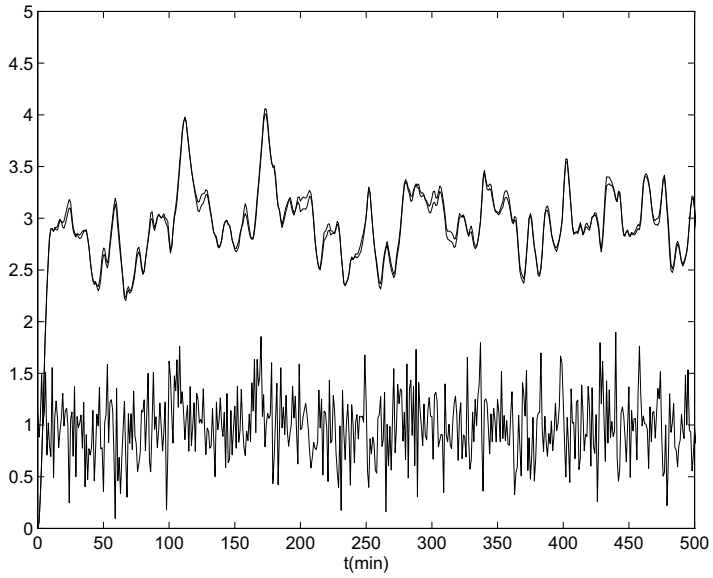
$$\frac{H_3}{\phi_I} = \frac{0{,}503}{s^3 + 1{,}842s^2 + 1{,}005s + 0{,}168} \tag{7.67}$$

kar pomeni parametre, ki so skoraj identični originalnim (en. (7.45)).

Sl. 7.19 kaže dobro ujemanje merjenih podatkov in odziva modela.



Sl. 7.19. Identifikacija za primer treh shranjevalnikov z metodo prilagajanje modela (impulzni vhod, merjeni odziv (ošumljen) in odziv modela)



Sl. 7.20. Identifikacija za primer treh shranjevalnikov z metodo najmanjših kvadratov (vhodni signal (spodaj) teri odziva modela in sistema)

Pri uporabi metode najmanjših kvadratov pa smo izhajali iz diskretnega zapisa prenosne funkcije (en. (7.51)) in dobili naslednji rezultat:

$$\frac{H_3(z)}{\phi_1(z)} = \frac{0{,}0543z^{-1} + 0{,}1347z^{-2} + 0{,}0227z^{-3}}{1 - 1{,}6911z^{-1} + 0{,}9216z^{-2} - 0{,}159z^{-3}} \qquad (7.68)$$

Vidimo, da so tudi v tem primeru identificirani koeficienti praktično identični, kar z dobrim prileganjem odziva modela in procesa ilustrira tudi Sl. 7.20.

V tem delu poglavja smo skušali nakazati značaj in vlogo modeliranja, ki predstavlja eno od pomembnih faz v življenjskem ciklu projektov na področju tehnologije vodenja. Sistemski pristop k modeliranju naj bi zagotovil potrebno sistematičnost pri razvoju modelov in nas spomnil tudi na korake, ki bi jih drugače morda pozabili izvesti. S tem se lahko izognemo dvema ekstremnima situacijama. Prvo imenujemo tveganje razvijalca modela, kjer slednji ne zaupa svojemu izdelku in ga še nepotrebno razširja in modificira, kar je posledica dejstva, da modela ni znal pravilno ovrednotiti. Druga situacija, ki jo imenujemo tveganje uporabnika modela, pa je povezana z neupravičeno uporabo modela v primerih, za katere ni bil razvit. Tako lahko ali pridemo do prekompleksnega in predragega modela ali pa do včasih katastrofalnih posledic, ki jih lahko povzroči nepravilna raba modela.

Zato naj bo inženir, ki se loteva modeliranja, kar se da specifičen v smislu vidikov, ki se tičejo področja, iz katerega izvira problem (nasloni naj se na problemskega strokovnjaka - problem vzpostavitve interdisciplinarne skupine), vendar pa kar se da fleksibilen in splošen v vidikih povezanih s simulacijo in simulacijskimi orodji. Tako ne bo prišlo do situacij, ko se mora strokovnjak za modeliranje preveč poglabljati v področje problema ali pa problemski strokovnjak žrtvovati preveč časa za spoznavanje tehnik modeliranja. Na tak način je možno uspešno reševanje konkretnih problemov z omenjenim pristopom v sprejemljivem času.

Zadnje raziskave na področju modeliranja so po eni strani usmerjene v računalniško podprto načrtovanje modelov, ki temelji na avtomatiziranih postopkih podprtih z ekspertnimi sistemi, ustreznimi bazami znanj, objektno orientiranimi pristopi itd. Po drugi strani pa se ob že uporabljanih in na posameznih področjih uveljavljenih pristopih kot so: prostorno modeliranje, Bond grafi, Petrijeve mreže, kvalitativno modeliranje, itd., vedno bolj kažejo možnosti vključevanja pristopov s področja umetne inteligence v postopke modeliranja (mehki modeli, nevronske mreže, genetski algoritmi, itd.). Največ pa seveda obetajo ustrezne kombinacije v smislu hibridnih modelov, ki naj bi združile kar največ dobrih lastnosti posameznih pristopov.

7.2.2 Metode za simulacijo dinamičnih sistemov

Simulacija dinamičnih sistemov je iterativna metoda, s pomočjo katere proučujemo funkcionalne lastnosti dinamičnih sistemov z eksperimentiranjem na ustreznem

modelu realnega objekta.

Jedro vsakega eksperimenta je simulacijski tek, ki prične v trenutku $t=t_0$ (*začetni čas simulacije oz. opazovanja*) in se konča v trenutku $t=t_{max}$ (*končni čas simulacije*). Med simulacijskim tekom se generirajo časovni poteki spremenljivk modela realnega objekta.

Osnova simulacije je reševanje algebrajsko-diferencialnih enačb z integriranjem. Analitična rešitev diferencialne enačbe in tabeliranje rešitve v določenih točkah neodvisne spremenljivke nima nobene zveze s simulacijo. Včasih sicer na ta način lahko pridemo hitreje do natančnejših rezultatov, vendar je v praksi uporabnost takega analitičnega pristopa zelo omejena (npr. le za linearne sisteme).

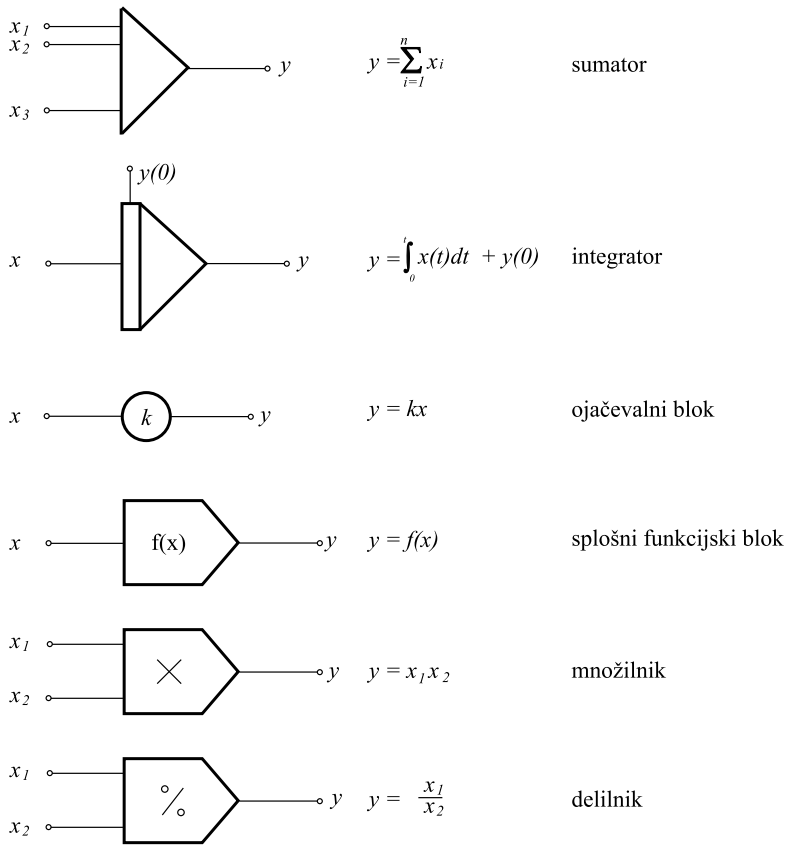
Simulacijo uporabljamo, če realni objekt še ne obstaja, če je eksperimentiranje z realnim objektom drago, nevarno ali dolgotrajno, zelo pomembna pa je simulacija tudi na področju vadbe in izobraževanja. Na področju vodenja sistemov se simulacija zlasti uporablja:

- za razvoj metod in celovitih rešitev vodenja;
- za odkrivanje in spoznavanje napak;
- za varen zagon in ustavitev procesa;
- za vadbo operaterjev in
- kot orodje za pomoč operaterjem pri odločanju.

Simulacijske metode in pristopi se delijo glede na vrsto modela, ki ga simuliramo, na zvezne, diskretne in hibridne (kombinirane). Glede na odnos med realnim in simulacijskim (računalniškim) časom pa lahko poteka simulacija v realnem času, počasneje ali hitreje. Na tem mestu se bomo omejili na zvezno simulacijo.

Osnova za zvezno simulacijo nekega modela je posebna grafična predstavitev, ki jo bomo imenovali simulacijska shema. Le-ta ima veliko skupnega z že prej omenjenimi bločnimi diagrami, ki jih zlasti uporabljamo pri zapisu sistemov vodenja. Zato bomo tudi osnovne gradnike simulacijskih shem imenovali bloke. Blok predstavlja funkcijo (dinamično ali algebraično operacijo med vhodnimi in izhodnimi signali) določenega gradnika simulacijske sheme. Grafično predstavitev bloka pa imenujemo ikona. Danes se v praksi uporabljajo zelo različne oblike simulacijskih shem, tako da praktično vsak izdelovalec simulacijskega orodja uvede kakšne svoje bloke oz. ikone. S čim manjšim naborom blokov oz. ikon bomo skušali uvesti shemo, ki bo uporabna za kakršno koli simulacijsko orodje.

Osnovne bloke, ki se uporabljajo v simulacijski shemi, prikazuje Sl. 7.21.



Sl. 7.21. Pogosto uporabljeni bloki oz. ikone v simulacijski shemi

Idealno orodje za modeliranje in simulacijo bi moralo omogočiti zapis modela (realnega problema) s fizikalnimi zakoni, ki izhajajo iz osnovnih zakonov o ohranitvi mase, energije oz. gibalne količine.

Poglejmo si sedaj osnovne pristope k simulaciji.

7.2.2.1 Izhodiščni zapis: sistem diferencialnih enačb prvega reda

Večina simulacijskih orodij zahteva zapis modela v prostoru stanj, t.j. s sistemom diferencialnih enačb prvega reda tako kot smo to predstavili v enačbi (7.5).

Zapišimo enačbo stanj v nekoliko poenostavljeni obliki

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), t) \tag{7.69}$$

Ker smo želeli poudariti univerzalnost simulacijskega pristopa, smo uporabili splošen zapis. Le-ta velja kot smo že omenili tudi za nelinearne in časovno spremenljive sisteme. Model torej opišemo tako, da opišemo vse njegove odvode. Ustrezne izraze

izvedemo z gradniki (bloki), ki jih nudi simulacijsko orodje. Z integriranjem odvodov generiramo spremenljivke stanj

$$\mathbf{x}(t) = \int \dot{\mathbf{x}}(t)dt \tag{7.70}$$

Izhode sistema dobimo z realizacijo algebrajskih enačb v obliki

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t),t) \tag{7.71}$$

Običajno enačbi (7.69) in (7.71) vnese uporabnik, enačbo (7.70) pa izvede simulacijsko orodje.

∇

Primer 7.5: *Simulacija ogrevanja prostora*

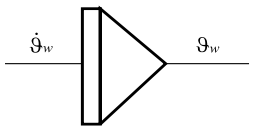
Matematični model ogrevanja prostora podaja enačba (2.4) v podpoglavju 2.2.3.

Če zamenjamo razliko temperatur $T(t)$-T_0 s ϑ_w, $\frac{A \cdot k_{izg}}{m \cdot c_p}$ z $\frac{1}{T}$ ter $\frac{1}{m \cdot c_p}$ s $\frac{k}{T}$ dobimo bolj običajno obliko enačbe

$$\dot{\vartheta}_w = -\frac{1}{T}\vartheta_w + \frac{k}{T}p \tag{7.72}$$

v kateri eksplicitno nastopata časovna konstanta *(T)* in ojačenje sistema *(k)*.

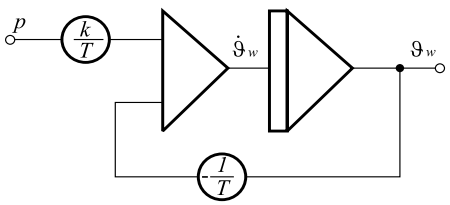
Ker imamo opravka le z eno diferencialno enačbo 1. reda, narišemo en integrator. Ustrezen korak prikazuje Sl. 7.22.



Sl. 7.22. Simulacijska shema, ki iz odvoda temperature generira temperaturo

Nato generiramo desno stran enačbe (7.72). Končno simulacijsko shemo prikazuje Sl. 7.23.

Kot lahko vidimo na Sl. 7.23, smo desno stran enačbe (7.72) realizirali s sumatorjem in dvema ojačevalnima blokoma. Potrebna predznaka sta vključena v ojačevalnih blokih.



Sl. 7.23. Simulacijska shema temperaturnega procesa

Δ

7.2.2.2 Indirektna metoda

Ker so matematični modeli dinamičnih sistemov običajno opisani z diferencialnimi enačbami višjih redov, uporabimo indirektno metodo kot splošnejši pristop. Pravzaprav je to postopek pretvorbe diferencialnih enačb višjega reda v več diferencialnih enačb prvega reda. Po tej metodi je potrebno najvišji odvod integrirati tolikokrat, kolikor je njegov red. S tem indirektno tvorimo vse nižje odvode in samo spremenljivko.

Indirektno metodo opišimo za sistem

$$y^{(n)} + f(y^{(n-1)}, y^{(n-2)}, \ldots, y', y, u; t) = 0 \qquad (7.73)$$

y je izhodni signal, u je vhodni signal, t pa je neodvisna spremenljivka simulacije (čas). Postopek opišemo v treh korakih:

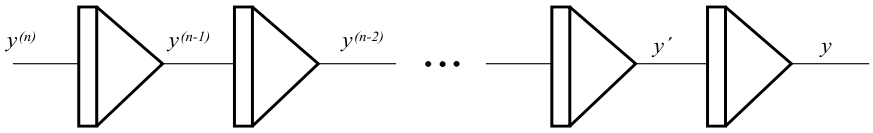
1. korak

Preuredimo diferencialno enačbo tako, da ostane na levi strani najvišji odvod, vse ostalo pa prenesemo na desno stran. Če je sistem zapisan v prostoru stanj (s sistemom diferencialnih enačb 1. reda), je zapis že ustrezen in prvi korak odpade

$$y^{(n)} = -f(y^{(n-1)}, y^{(n-2)}, \ldots, y', y, u; t) \qquad (7.74)$$

2. korak

Narišemo kaskado n integratorjev, če je n red najvišjega odvoda.

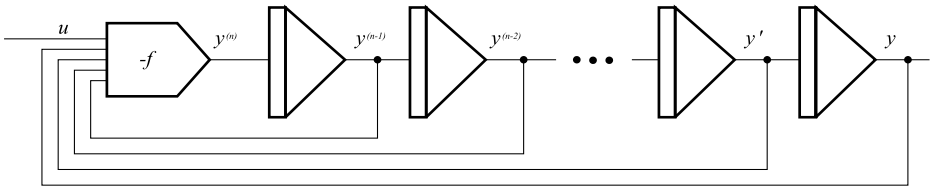


Sl. 7.24. Drugi korak indirektnega postopka

Predpostavimo, da je vhod v prvi integrator (najvišji odvod) znan, preostali integratorji pa tvorijo nižje odvode in samo spremenljivko oz. rešitev diferencialne enačbe.

3. korak

Z upoštevanjem nižjih odvodov in rešitve diferencialne enačbe generiramo negativno funkcijsko odvisnost, ki realizira desno stran enačbe (7.74). Izhod bloka, ki tvori negativno funkcijsko odvisnost, je enak najvišjemu odvodu, zato ga moramo povezati na vhod prvega integratorja. Pri generaciji funkcije -f uporabljamo različne bloke (razen integratorja), kar zavisi od oblike diferencialne enačbe. Dobljeni obliki, ki jo prikazuje Sl. 7.25, pravimo tudi kanonična oblika.



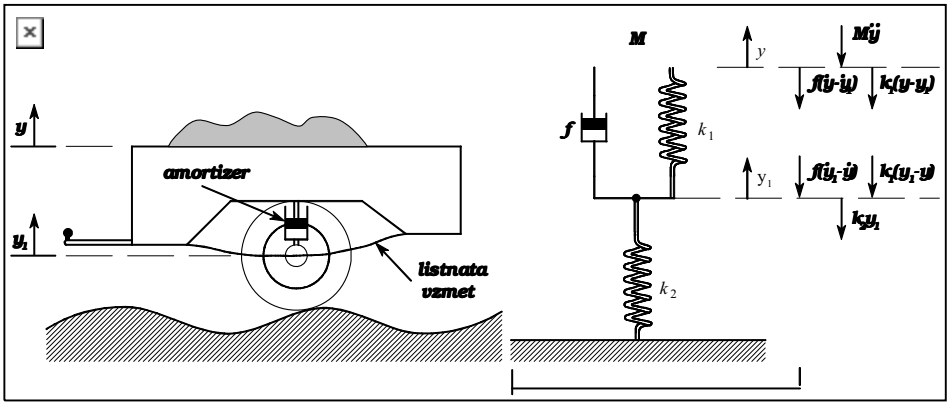
Sl. 7.25. Tretji korak indirektnega postopka in končna simulacijska shema

Postopek, ki smo ga opisali, je neposredno uporaben, če diferencialna enačba ne vsebuje odvodov vhodnega signala. Če pa le-ti nastopajo, je v primeru linearnih sistemov bolj smiselno sistem simulirati po konceptu prenosnih funkcij, kar bomo opisali nekoliko kasneje.

∇

Primer 7.6: *Simulacija vzmetenja avtomobilske prikolice*

Vzmetenje avtomobilske prikolice in njen mehanski model prikazuje Sl. 7.26 (Matko in ostali, 1992). M je polovična masa prikolice in tovora (maso kolesa zanemarimo), k_1 je konstanta elastičnosti listnate vzmeti, k_2 je konstanta elastičnosti pnevmatike, f je konstanta dušenja amortizerja, y je navpični pomik prikolice (karoserije), y_1 pa je pomik kolesa. Proučujemo navpično gibanje prikolice, ko jo razbremenimo.



Sl. 7.26. Vzmetenje avtomobilske prikolice in ustrezni mehanski model

Matematični model podaja enačba

$$\dddot{y} + \frac{k_1 + k_2}{f}\ddot{y} + \frac{k_2}{M}\dot{y} + \frac{k_1 k_2}{Mf} y = 0 \qquad y(0) = -y_0 \tag{7.75}$$

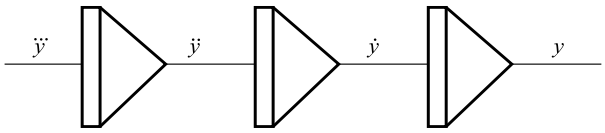
ki jo lahko poenostavimo v obliko

$$\dddot{y} + a\ddot{y} + b\dot{y} + cy = 0 \qquad y(0) = -d \tag{7.76}$$

1. korak: Preuredimo enačbo (7.76)

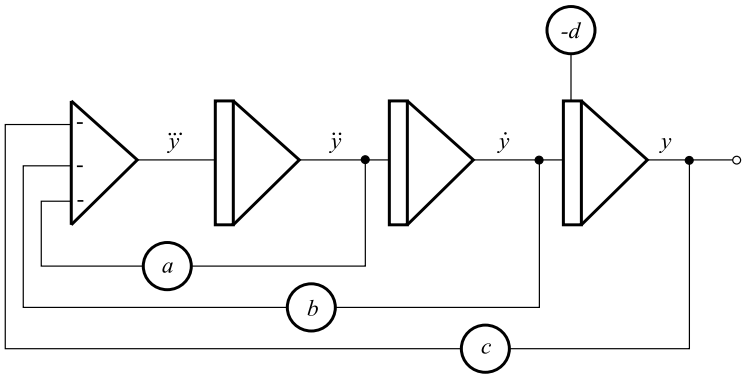
$$\dddot{y} = -a\ddot{y} - b\dot{y} - cy \qquad y(0) = -d \tag{7.77}$$

2. korak: Narišemo kaskado treh integratorjev, kar prikazuje Sl. 7.27.



Sl. 7.27 Simulacijska shema za drugi korak

3. korak: Zaključimo simulacijsko shemo, kot prikazuje Sl. 7.28. V tem primeru smo predznake podali v sumacijskem bloku.



Sl. 7.28. Simulacijska shema sistema avtomobilskega vzmetenja

Δ

7.2.2.3 Simulacija prenosnih funkcij

Prenosne funkcije imajo pomembno vlogo pri analizi in načrtovanju sistemov. Zlasti je učinkovita nazornost bločnih diagramov, v katerih nastopajo med ostalimi funkcionalni bloki tudi prenosne funkcije.

Od številnih metod za simulacijo prenosnih funkcij bomo opisali le *delitveno metodo*.

Postopek bomo prikazali na spodnjem primeru.

∇

Primer 7.7: *Simulacija prenosne funkcije po delitveni metodi*

Prenosno funkcijo

$$G(s) = \frac{Y(s)}{U(s)} = \frac{as^3 + bs^2 + cs + d}{s^3 + es^2 + fs + g} \tag{7.78}$$

simuliramo z delitveno metodo s pomočjo naslednjih korakov:

Če koeficient pri členu z najvišjo potenco v imenovalcu ni enak 1, je potrebno vse člene polinomov v števcu in imenovalcu deliti s tem koeficientom.

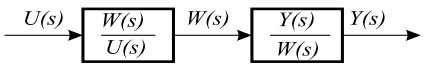
Prenosno funkcijo $\frac{Y(s)}{U(s)}$ razdelimo s pomočjo pomožne spremenljivke *W(s)* v dve prenosni funkciji; prva predstavlja realizacijo imenovalca

$$\frac{W(s)}{U(s)} = \frac{1}{s^3 + es^2 + fs + g} \tag{7.79}$$

druga pa realizacijo števca

$$\frac{Y(s)}{W(s)} = as^3 + bs^2 + cs + d \tag{7.80}$$

Delitveni postopek nazorno prikazuje Sl. 7.29.



Sl. 7.29. Bločna shema, ki predstavlja delitveni postopek

Preuredimo enačbo (7.79) v obliko

$$s^3 W = U - es^2 W - fsW - gW \tag{7.81}$$

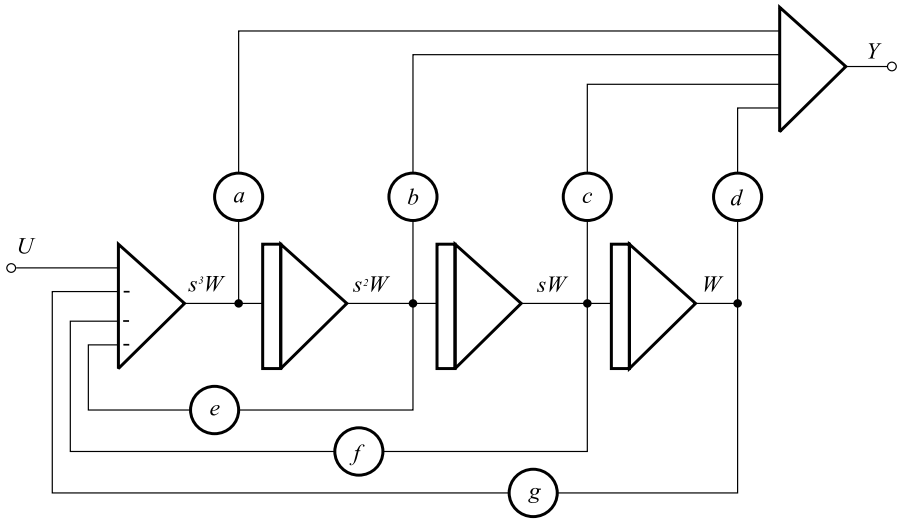
Preuredimo enačbo (7.80) v obliko

$$Y = as^3 W + bs^2 W + csW + dW \tag{7.82}$$

S pomočjo enačb (7.81) in (7.82) realiziramo simulacijsko shemo, ki jo prikazuje Sl. 7.30.

Če koeficient pri členu s^3 ne bi bil 1, bi bile vrednosti ojačevalnih blokov v Sl. 7.30 kvocienti koeficientov a, b, c, d, e, f in g s tem koeficientom. Kot vidimo iz Sl. 7.30, sestoji struktura iz dveh delov. Prvi del (enačba (7.81)) realizira imenovalec z indirektno metodo (le ta namreč omogoča, da so razen izhodne spremenljivke dostopni tudi odvodi), drugi del pa generira števec (enačba (7.82)). Metoda je zelo primerna za simulacijo večih prenosnih funkcij z enakim imenovalcem, saj le enkrat simuliramo prenosno funkcijo $\frac{W}{U}$.

Shema na Sl. 7.30 predstavlja realizacijo prenosne funkcije, ki je v teoriji vodenja znana kot *vodljivostna kanonična oblika*. Metoda potrebuje toliko integratorjev, kolikor je red prenosne funkcije. Uporabljamo jo tudi pri simulaciji sistemov, ki so opisani z diferencialnimi enačbami, ki vsebujejo tudi višje odvode vhodnega signala (npr. $\dddot{y} + e\ddot{y} + f\dot{y} + gy = a\dddot{u} + b\ddot{u} + c\dot{u} + du$ je ekvivalenten zapisu s prenosno funkcijo po enačbi (7.78)).



Sl. 7.30. Simulacijska shema ob uporabi delitvene metode

Δ

7.2.2.4 Modularni pristop

Pri simulaciji kompleksnejših sistemov je treba že v fazi modeliranja upoštevati principe modularnosti in objektne orientiranosti, kar pomeni, da večje sklope realiziramo z manjšimi zaključenimi podsklopi (podmodeli, prenosne funkcije, …). Medtem ko je pri analitičnih metodah analize in načrtovanja potrebno take podsklope združevati oz. poenostavljati (npr. poenostavljanje bločnih diagramov s pomočjo algebre bločnih diagramov), pa je pomembna prednost pri simulaciji v tem, da vsak podsklop skušamo ločeno simulirati po eni izmed obravnavanih metod in nato

posamezne podsklope ustrezno povezati. Na ta način so simulacijske sheme zelo pregledne in uspešno služijo za dokumentacijo.

∇

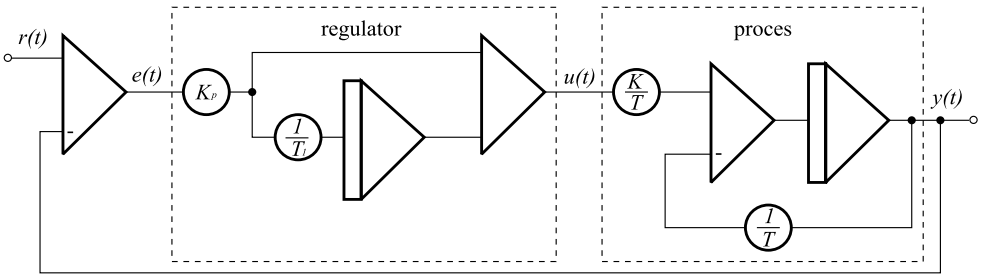
Primer 7.8: *Simulacija regulacijskega sistema s PI regulatorjem (glej podpoglavje 7.3.3.3)*

$$G_R(s) = \frac{U(s)}{E(s)} = K_P(1 + \frac{1}{T_I s}) \tag{7.83}$$

reguliramo temperaturo v prostoru. Model ogrevanja podaja prenosna funkcija

$$G_P(s) = \frac{Y(s)}{U(s)} = \frac{K}{Ts+1} \tag{7.84}$$

Ob upoštevanju nazornosti in modularnosti dobimo simulacijsko shemo, ki jo prikazuje Sl. 7.31. Slika pregledno kaže regulacijsko strukturo. Če bi npr. simulirali regulacijski sistem s pomočjo ene zaprtozančne prenosne funkcije $\frac{Y(s)}{R(s)}$, ne bi bila dostopna regulirna veličina *u(t)*.



Sl. 7.31 Simulacijska shema sistema za regulacijo temperature

Ne glede na to, da danes že obstajajo orodja, ki so specializirana za različna področja ali pa orodja, ki imajo večjo podporo v fazi modeliranja, smo si ogledali le osnovni in predvsem splošnonamenski simulacijski pristop. Le-ta je univerzalno uporaben pri simulaciji linearnih ali nelinearnih modelov iz katerihkoli področij.

Če je dinamični model podan v diskretni obliki (diferenčno-algebrajske enačbe, diskretne prenosne funkcije), ostane simulacijski koncept nespremenjen. Namesto integratorja kot osnovnega dinamičnega operatorja zveznih sistemov nastopa diskretna zakasnitev (z^{-1}), namesto odvodov (npr. $\dot{x}, \ddot{x}, \ldots$) pa predikcije stanj (npr. $x(k+1)$, $x(k+2)$,…).

∆

7.3 Metode in postopki za realizacijo funkcij vodenja

Če je bil prejšnji del poglavja namenjen splošnejšim in širše uporabnim metodam in postopkom, potem je namen tega podpoglavja predstaviti nekoliko bolj specifične pristope in postopke, ki jih uporabljamo pri realizaciji mehanizmov posameznih funkcij vodenja. Te mehanizme smo na splošnejšem nivoju opisali že v četrtem poglavju, tukaj bomo pa nekatere od njih skupaj s principi in postopki njihovega načrtovanja nekoliko bolj poglobljeno obdelali.

Podpoglavje smo strukturirali po različnih funkcijah vodenja, od katerih jih večino najdemo v vsakem nekoliko večjem in zahtevnejšem sistemu vodenja. Predstavljene metode pa segajo od povsem rutinskih in običajnih, do nekoliko bolj eksotičnih, ki se v praksi zaenkrat še redkeje uporabljajo.

Prvi del poglavja je namenjen postopkom za realizacijo različnih funkcij nadzora, pri čemer je poseben poudarek dan problemom ocenjevanja nemerljivih spremenljivk, kot eni od zahtevnejših funkcij v tem sklopu. Sledi podrobnejši opis problematike načrtovanja in realizacije postopkov za odkrivanje napak v procesih, kar prav tako spada med zahtevnejše funkcije nadzora.

V nadaljevanju sta obdelani dve najbolj pogosti funkciji in sicer sekvenčno vodenje in regulacija. Težišče je dano regulaciji, kjer so podrobneje predstavljeni proporcionalno-integrirno-diferencirna (PID) regulacija, adaptivna regulacija, prediktivna regulacija in multivariabilna regulacija.

V zadnjem delu pa so predstavljeni nekoliko novejši koncepti hibridnega vodenja, ki pomenijo kombinacijo sekvenčnega vodenja in regulacije, poglavje pa se zaključuje s predstavitvijo metod za optimiranje procesov.

Večina metod (z izjemo hibridnega vodenja in optimiranja) se nanaša na tako imenovani osnovni nivo vodenja.

7.3.1 Nadzor procesov

Pojem nadzora opredeljuje nalogo sprotnega ugotavljanja stanja tehnološkega sistema. Pri tem gre za tri pomembne vidike:

- stanje v katerem je proces (v smislu faze obratovanja);
- stanje opreme ter
- stanje produktov in proizvodnje.

Stanje v katerem sistem obratuje je ponavadi definirano z nekim območjem procesnih veličin. Regulacijski sistem poskrbi, da se procesne veličine tudi dejansko gibljejo v tem področju. Morebitno odstopanje se sporoči operaterju v obliki alarma. V procesnih in kemijskih industrijah pa je dostikrat tako, da dejanskega stanja ni možno neposredno

meriti. Zato se je treba poslužiti tim. *izračunanih* spremenljivk, ki so stvarni pokazatelj stanja procesa, niso pa neposredno merljive. Le-te pa je treba izpeljati iz dostopnih meritev.

Ugotavljanje *stanja opreme* se v tradicionalni industrijski praksi rešuje s periodičnimi pregledi (enkrat ali večkrat letno) ali tekom rednih remontov. Pri kritičnih tehnologijah je to seveda premalo, saj sleherni izpad, denimo elektrarne, pomeni takoj velikanske stroške. V takih primerih se zahteva sprotno ugotavljanje stanja opreme (aktuatorjev, senzorjev in procesnih komponent). Tudi v drugih panogah procesne industrije, še posebej tam kjer je imperativ kakovost, se zahteva *sprotno* vzdrževanje, t.j. sprotno odkrivanje napak v dovolj zgodnji fazi, še preden se poruši funkcionalnost sistema.

Namen ugotavljanje *stanja produkta* je sprotno določanje njegove kakovosti ter, po možnosti, odkrivanje vzrokov za morebitna odstopanja od predpisanih normativov. Problematika je precej podobna problematiki ugotavljanja stanja opreme.

V podporo omenjenim funkcijam služijo funkcije zbiranja podatkov iz senzorjev, njihovo shranjevanje v obliki zgodovine procesa ter predstavitev v numerični ali grafični obliki na zaslonu. Na podlagi prispelih podatkov iz procesa operater najprej izloči pomembne informacije in nato sprejema odločitve ter ukrepa. Opazovanje, izločanje informacije, odločanje in ukrepanje operaterja tvorijo *nadzor procesa* (Sl. 7.32).

Bistvena razlika med nadzorom in regulacijo je v pogostosti poseganja v proces. Pri regulacijah so presledki med dvema zaporednima posegoma praviloma kratki, največ reda velikosti sekunde ali minute. Pri nadzoru pa so le-ti neenakomerni in se običajno izražajo v urah ali dnevih. V nekaterih primerih (npr. pri čistilnih napravah) operaterji na osnovi opazovanih podatkov sproti posegajo v sistem preko njegovih regulacijskih vhodov. Gre za specifično mešanico nadzora in vodenja, ki jo imenujemo *nadzorno vodenje*. Običajno ga zasledimo pri procesih z zelo počasno dinamiko.

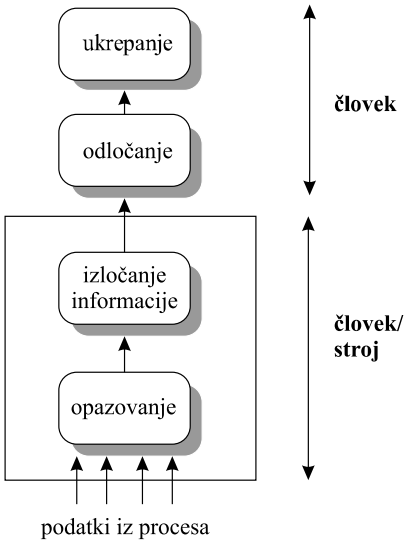
Naloge procesnega nadzora

V sklopu procesnega nadzora mora operater:

- opazovati prikaze s podatki o izmerjenih spremenljivkah v procesu;
- ugotoviti sleherno odstopanje v delovanju procesa od normalnega stanja;
- sklepati o možnih vzrokih za nepravilno delovanje;
- ukrepati v izjemnih situacijah in spraviti sistem v področje varnega in učinkovitega delovanja;
- pravilno oceniti stanje v primeru pomanjkljivih ali premalo natančnih podatkov.

Vseh nalog je lahko toliko, da jih operater težko kakovostno in zanesljivo opravlja. Glavni problem je v tem, da ni vsaka številka ali izmerjeni podatek tudi *informacija.* Dejstvo je, da operater potrebuje *relevantno* informacijo in ne zgolj kopico podatkov. Torej je poglavitno vprašanje kako odpraviti *redundanco* v podatkih, ki se posredujejo operaterju. Trenutno mora večino tega opraviti kar operater sam, pri čemer je potrebno

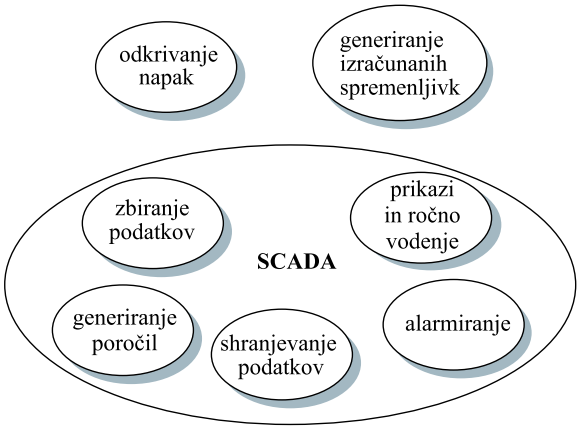
precej izkušenj, zbranosti in neprekinjene pozornosti. V okviru delovnika lahko to zahteva velik napor. Zato se od sodobnih sistemov nadzora pričakuje, da bodo nudili *podporo operaterju* predvsem pri *izločanju kvalitetne informacije* iz podatkov, ki prihajajo iz procesa ter njene predstavitve v jasnem in preprosto dosegljivem formatu. S tem naj bi se operaterja razbremenilo odvečnih naporov (Sl. 7.32).



Sl. 7.32. Osnovne faze pri procesnem nadzoru

Osnovne funkcije sistemov nadzora

Osnovni elementi nadzornih sistemov so podani na Sl. 7.33 in jih lahko opredelimo kot:



Sl. 7.33. Osnovni elementi sistemov nadzora (v večji elipsi so segmenti, ki jih podpirajo trenutni sistemi SCADA)

- zbiranje podatkov;

- generiranje poročil;
- shranjevanje podatkov;
- alarmiranje;
- prikazi in ročno vodenje;
- generiranje izračunanih spremenljivk in
- odkrivanje napak.

V današnjih SCADA (Supervisory Control And Data Acquisition) sistemih praviloma nastopa prvih pet in deloma šesti element. Avtomatsko odkrivanje napak ter generiranje zahtevnejših izračunanih spremenljivk nastopata bolj redko.

Izraz *izračunane spremenljivke* se nanaša na širši spekter spremenljivk, ki jih dobimo z matematičnimi operacijami na eni ali več merjenih spremenljivkah. Na ta način se računajo:

- trendi v statistiki procesnih spremenljivk;
- ocene nemerljivih a pomembnih stanj procesa;
- ocene redko vzorčenih reguliranih spremenljivk, za katere ne obstajajo ustrezni senzorji, temveč le laboratorijske analize.

Avtomatsko *odkrivanje napak* je postopek za odkrivanje potencialnih virov tistih napak, ki lahko pripeljejo proces izven zahtevanega območja delovanja ali celo do izpada.

V nadaljevanju tega podpoglavja bomo podrobneje spregovorili o generiranju izračunanih spremenljivk ter odkrivanju napak, več o ostalih funkcijah pa podaja podpoglavje 8.4.2, ki je posvečeno programskim orodjem za nadzorne sisteme.

7.3.1.1 Generiranje izračunanih spremenljivk

Poglejmo si nekatere najpogostejše pristope.

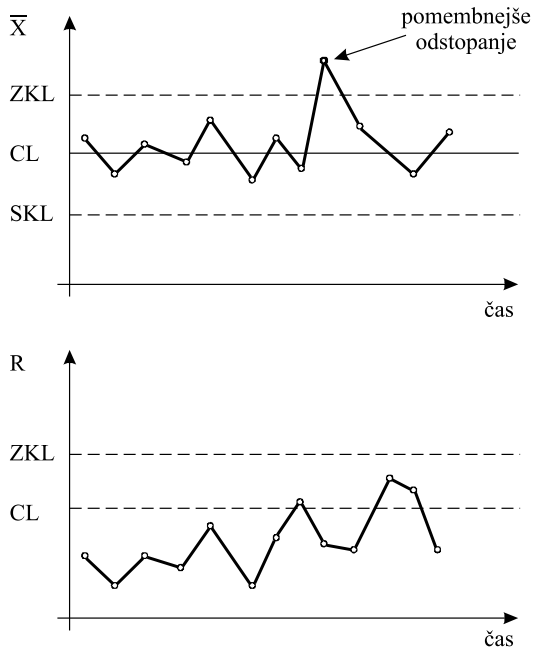
Statistično vodenje procesa: diagrami vodenja

Statistično vodenje procesa (angl. statistical process control) je množica postopkov, ki se lahko uporabljajo med rednim obratovanjem procesa s ciljem izboljšanja učinkovitosti proizvodnje (Mendel, 1987). Le-ti so primerni za nadzor kvalitete produkta, hitrosti proizvodnje, proizvodnih stroškov, skratka za ugotavljanje *stanja produktov in proizvodnje*. Najbolj pogosto se uporabljajo po zaključku proizvodnega procesa pri nadzoru izhodnega produkta. Uporaba pa je smiselna v vseh fazah proizvodnega procesa, tudi pri nadzoru vhodnih materialov in posameznih procesnih enot.

Statistično vodenje procesa temelji na statističnih izračunih, ki so izvedeni na vzorčenih vrednostih procesnih veličin. Pri tem vnaprej predpostavimo, da se naključne vzorce da opisati z neko funkcijo porazdelitve verjetnosti. Običajno je to Gaussova normalna porazdelitev, ta pa je definirana s srednjo vrednostjo μ in standardno deviacijo σ.

Statistične analize so dober pokazatelj razmer pri velikem številu vzorcev ali meritev. Statistične parametre, ki jih pri tem dobimo, npr. μ in σ, običajno primerjamo z referenčnimi vrednostmi, ki ustrezajo normalnim razmeram. Ko vrednotimo statistične parametre določenega števila vzorcev, le-ti več ali manj odstopajo od referenčnih vrednosti.

Za odločanje o tem, ali je prišlo do kakšnega opaznejšega odstopanja, uporabljamo *diagrame vodenja* (angl. control charts). Z diagrami vodenja nazorno ugotavljamo odklone srednje vrednosti ($\overline{x}$) in standardne deviacije (R) nekega vzorca od referenčnih vrednosti. Srednjo vrednost dobljenih vzorcev iz procesa ter njihovo stresanje preprosto rišemo v okviru predpisane meje statističnega zaupanja (Sl. 7.34).



Sl. 7.34. Diagrami vodenja

Potem se moramo odločiti med dvema hipotezama in sicer, da do odklona ni prišlo, oz. da je do odklona prišlo. Na primer: vzemimo hipotezo, da se srednja vrednost obravnavanih vzorcev ujema z referenčno srednjo vrednostjo CL, tj. $\overline{x} = CL$. Njena alternativna hipoteza je potem $\overline{x} \neq CL$. Če srednja vrednost podatkov izmerjenega vzorca pade zunaj predpisanih meja ZKL in SKL (Sl. 7.34), potem lahko sklepamo, da je veljavna druga hipoteza ($\overline{x} \neq CL$). To pomeni, da je v proizvodnem procesu nekaj narobe. Podobno lahko sklepamo v primeru prekoračitve stresanja na R diagramu.

Poleg testiranja prekoračitve meja je za potrebe nadzora potrebno določati še trende v statistiki opazovanih procesnih spremenljivk.

Glavne težave pri diagramih vodenja so v tem, da obravnavajo "nihanja" v vsaki procesni spremenljivki posebej, ne da bi upoštevali medsebojno odvisnost med procesnimi spremenljivkami. Zato včasih lahko pride do hkratnega odklona v večih spremenljivkah zaradi skupnega vzroka. Operater seveda dobi več alarmov naenkrat, o vzroku pa mora sklepati sam.

Analiza glavnih komponent

Diagrami vodenja so uporabni za izločanje informacije iz vzorcev ene procesne spremenljivke. V industrijskih procesih se ponavadi zajema veliko število podatkov o procesu. Tipično je, da so zajeti podatki medseboj zelo korelirani, ter da je v njih prisoten šum. Pravimo, da sta časovna poteka dveh spremenljivk korelirana, če je dinamika prve podobna dinamiki druge. Na primer: če vrednost prve spremenljivke naraste, naraste tudi vrednost druge in obratno. V tem primeru vsebuje kvalitetno informacijo le ena spremenljivka, drugi pa pravimo, da je redundantna. Redundantnih spremenljivk pa je lahko zelo veliko. Zaradi tega postane nadzor procesa zelo naporen, saj je zelo težko opazovati, npr. nekaj deset (podobnih) časovnih potekov in ugotavljati ali je kaj narobe.

Analiza glavnih komponent (PCA – Principal Component Analysis) je zelo praktično orodje za izločanje informacij iz velikega števila vzorčenih signalov. Z ustrezno kombinacijo originalnih meritev PCA pravzaprav poišče nekaj značilnih spremenljivk, v katerih je povzeta vsa bistvena informacija o procesu. Najpogosteje se uporabljajo dve do tri značilne spremenljivke, ki jim pravimo *zadetki*.

Idejo PCA najkrajše razložimo takole. Denimo, da v procesu merimo J spremenljivk $x_1,x_2,...,x_J$ ter, da za vsako spremenljivko razpolagamo s K vzorci. Zaradi preglednosti zložimo K vzorcev za vsako spremenljivko v vektor, npr. $\mathbf{x}_i=[x_i(1),x_i(2),...,x_i(K)]'$. Vpeljimo še matriko $\mathbf{X}=[\mathbf{x}_1\ \mathbf{x}_2\ ...\ \mathbf{x}_J]$. Bistvo PCA je v naslednjem:

Prvič, PCA razstavi matriko **X** na sledeči način:

$$\mathbf{X} = \mathbf{t}_1\mathbf{p}_1^T + \mathbf{t}_2\mathbf{p}_2^T + \cdots + \mathbf{t}_r\mathbf{p}_r^T \qquad (7.85)$$

kjer vektorji $\mathbf{t}_i$ (dim($\mathbf{t}_i$)=K) predstavljajo *zadetke*.

Drugič, število zadetkov je (bistveno) manjše od števila spremenljivk, t.j. $r<J$. To je glavni namen kompresije podatkov oziroma izločanja dejanske informacije iz obstoječih meritev.

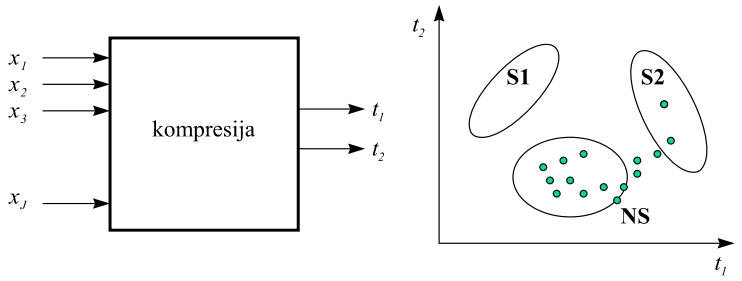
Tretjič, vektorji $\mathbf{p}_1$, $\mathbf{p}_2$,..., $\mathbf{p}_r$ so medseboj ortogonalni, t.j. $\mathbf{p}_i^T\ \mathbf{p}_j=0$ za $i\neq j$, dim($\mathbf{p}_i$)=J.

Vektorje $\mathbf{p}_1$, $\mathbf{p}_2$,..., $\mathbf{p}_r$ lahko določimo iz nabora referenčnih potekov, ki odražajo pričakovano obnašanje procesa. Pri tem običajno uporabljamo zbrano zgodovino procesa. Postopek opravimo ločeno od delovanja procesa, torej na nesproten način (off-line), rezultat obdelave pa so vektorji $\mathbf{p}_1$, $\mathbf{p}_2$,..., $\mathbf{p}_r$, ki predstavljajo "prstni odtis" procesa in jih shranimo za nadaljnje sprotno procesiranje.

Sprotno procesiranje poteka takole. Ko dobimo neki novi nabor meritev iz procesa v časovnem trenutku nT_0 (T_0 je čas vzorčenja), t.j. $\mathbf{x}(n)= [x_1(n),x_2(n),...,x_J(n)]$ potem namesto, da bi opazovali originalne vrednosti $\mathbf{x}$, opazujemo *zadetke*, t.j. vrednosti $t_1(n)$, $t_2(n)$,..., $t_r(n)$. Vrednost zadetka $t_i(n)$ zlahka izračunamo takole:

$$t_i(n) = \mathbf{x}(\mathbf{n})\mathbf{p}_i \qquad i = 1,2,...,r \tag{7.86}$$

Situacijo ponazarja Sl. 7.35 za primer, ko za opis procesa potrebujemo le dva zadetka. Če je v procesu vse v redu, se mora točka (t_1, t_2) nahajati v predpisanem področju (NS). Ko je nekaj narobe, se točka (t_1, t_2) "sprehodi" drugam (v kakšen drug "oblaček" – npr. S2 ali S1). Na ta način operater zlahka ugotovi da je nekaj narobe.



Sl. 7.35. Osnovna ideja analize glavnih komponent. V ravnini zadetkov je označeno področje pričakovanega delovanja (NS) in področja, ki ustrezajo različnim anomalijam v procesu

Ocenjevanje nemerljivih stanj procesa

V procesnem vodenju se dostikrat srečujemo s situacijo, ko bi za učinkovito odločitev pri nadzoru potrebovali več informacije, predvsem o nekaterih fizikalnih veličinah, ki se jih ne da neposredno meriti. Razlog za to je, da ustrezni senzorji bodisi ne obstajajo, ali pa so predragi. V kemijskih industrijah je dosti takih primerov, zlasti kadar gre za podatke o koncentracijah kemijskih snovi v surovinah ali produktih in podobno (npr. Kappa število pri ugotavljanju kvalitete celuloze). Ker hitrost spreminjanja koncentracije snovi zavisi od pogojev v kemijskem procesu, pravimo, da koncentracija predstavlja *dinamično stanje*[8] procesa.

Pojem dinamičnega stanja (v nadaljevanju stanje) se nanaša na minimalno količino informacije, ki je potrebna za enolični opis obnašanja procesa v prihodnosti na osnovi poznanih vhodnih signalov, začetnih vrednosti stanj in znanih vrednosti parametrov sistema. Na primer: za opis gibanja masne točke, ki je obešena na vzmeti, sta potrebni dve stanji in sicer položaj ter hitrost masne točke. Stanje procesa, ali

[8] Pojem stanja se uporablja v dveh različnih kontekstih. Po eni strani s tem označujemo diskretno stanje procesa (npr. zagon, ustavitev, okvara itd.), po drugi strani pa pod tem razumemo stanje dinamičnega sistema (hitrost, položaj, temperatura itd.).

ustrezna funkcija stanja, lahko odgovarja neki fizikalni veličini, čeprav to ni nujno. Razlog je v tem, da je s transformacijami spremenljivk stanja vedno možno preiti iz enega prostora stanj v drug prostor stanj.

V kemijskih procesih pogosto srečujemo situacijo, ko nekatera stanja procesa odražajo pomemben aspekt delovanja, a so nemerljiva. Sprotno spremljanje njihovih ocen je važno za uspešen nadzor in vodenje procesa. Naloga *ocenjevanja stanj* je izračunati ocene neznanih stanj iz vhodnih in izhodnih izmerjenih podatkov in *modela* procesa.

Ocenjevanje stanj v linearnih sistemih

Da bi lažje predstavili osnovno idejo problematike ocenjevanja stanj, vzemimo linerni časovno nespremenljivi sistem, ki je definiran z naslednjim modelom v prostoru stanj:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) \end{aligned} \tag{7.87}$$

kjer so **x**, **u** in **y** *n*-dimenzionlni vektor stanja, *m*-dimenzionalni vektor vhodov in *p*-dimenzionalni vektor merljivih izhodov. Če je možno *enolično* ugotoviti stanje sistema $\mathbf{x}(t_0)$ v trenutku t_0 na osnovi izmerjenih $\mathbf{y}(t)$ in $\mathbf{u}(t)$ v časovnem intervalu $t_0 \leq t \leq T_1$, potem pravimo, da je sistem *spoznaven.* Lepa lastnost linearnih časovno nespremenljivih sistemov je, da je spoznavnost *inherentna lastnost* sistema, t.j. definirana je z njegovo konstrukcijo in ne zavisi od vzbujanja procesa. Prevedeno v matematični jezik to pomeni, da je spoznavnost definirana s polnim rangom (reda n) *matrike spoznavnosti* **O** sistema, to je:

$$\mathbf{O} = \begin{bmatrix} \mathbf{C}^T \\ \mathbf{C}^T\mathbf{A} \\ \mathbf{C}^T\mathbf{A}^2 \\ \cdots\cdots \\ \mathbf{C}^T\mathbf{A}^{n-1} \end{bmatrix} \tag{7.88}$$

Če je sistem zares spoznaven, potem je možno oceniti stanja v (7.87) iz merjenih izhodov ***y***, npr. s pomočjo *Luenbergerjevega opazovalnika,* ki je naslednje oblike (Sl. 7.36):

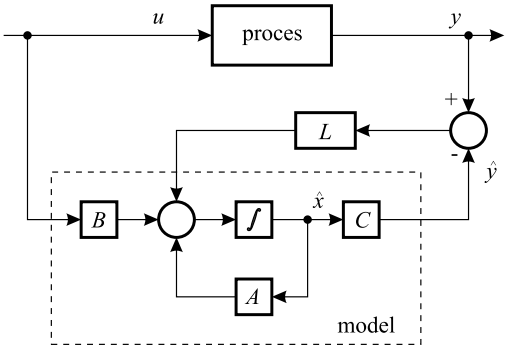
$$\dot{\hat{\mathbf{x}}}(t) = \mathbf{A}\hat{\mathbf{x}}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{L}(\mathbf{y}(t) - \mathbf{C}\hat{\mathbf{x}}(t)) \tag{7.89}$$

Simbol $\hat{\mathbf{x}}$ označuje vektor ocenjenih stanj procesa, **L** je matrika ojačenj opazovalnika. Dinamiko napake ocen $\mathbf{e}(t) = \mathbf{x}(t) - \hat{\mathbf{x}}(t)$ zlahka ugotovimo iz izrazov (7.87) in (7.89), iz česar dobimo:

$$\dot{\mathbf{e}}(t) = (\mathbf{A} - \mathbf{L}\mathbf{C})\mathbf{e}(t) \tag{7.90}$$

Očitno napaka ocen zavisi od izbrane matrike ojačenj **L**. Če je opazovalnik stabilen, potem napaka z naraščanjem časa upada eksponencialno proti nič. Da bi to dosegli, je potrebno skrbno izbrati ojačenje opazovalnika **L**, in sicer tako, da imajo vse lastne vrednosti matrike (**A**-**LC**) negativne realne dele. Omenjene lastne vrednosti imenujemo

”poli opazovalnika”. Za namen načrtovanja stabilnega opazovalnika lahko uporabimo metodo premikanja polov. Bolj ko so realni deli polov negativni, hitrejša je rekonstrukcija stanja oziroma napaka ocene gre hitreje proti nič. Pri tem pa občutljivost na šum in napake modeliranja narašča. Zato je pri načrtovanju vedno potreben kompromis med hitrostjo rekonstrukcije in občutljivostjo.



Sl. 7.36. Opazovalnik stanj

Ocenjevanje stanj v nelinearnih sistemih

Spoznavnost pri linearnih sistemih je *globalna* lastnost, kar pomeni da je sistem spoznaven ne glede na področje delovanja. Pri nelinearnih sistemih pa to ni tako. Tu govorimo o *lokalni* spoznavnosti, t.j. spoznavnosti v okolici neke ravnotežne točke (Muske in Edgar, 1997). Predpostavimo, da je v času t_0 sistem v stanju $\mathbf{x}_0$. V časovnem intervalu $t_0 \leq t \leq T$ vektor stanj opiše tirnico $\mathbf{y}(t,t_0,\mathbf{x}_0,\mathbf{u}(\mathrm{t}))$. Predpostavimo še, da se tirnica vektorja stanj giblje v bližini ravnotežne točke. Če je na osnovi tirnice $\mathbf{y}(t)$ in $\mathbf{u}(t)$ možno enolično ugotoviti, da je v trenutku t_0 sistem bil v stanju $\mathbf{x}_0$ za katerikoli dopustni vhod $\mathbf{u}(t)$ (se pravi tak, ki “vozi” sistem okoli ravnotežne točke), potem pravimo, da je sistem lokalno spoznaven.

Če je sistem lokalno spoznaven v vsaki ravnotežni točki, potem je tudi globalno spoznaven.

Pojem lokalne spoznavnosti se da uporabiti pri načrtovanju opazovalnika za delovanje v režimih blizu ravnotežne točke. Dinamiko nelinearnega sistema

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x},\mathbf{u},\mathrm{t}) \\ \mathbf{y} &= \mathbf{g}(\mathbf{x},\mathrm{t}) \end{aligned} \tag{7.91}$$

lineariziramo okoli ravnotežne točke $\mathbf{x}^*$ z uporabo Taylorjeve vrste:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x},\mathbf{u},t) = \mathbf{f}(\mathbf{x}^*,\mathbf{u},t) + \frac{\partial \mathbf{f}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x}-\mathbf{x}^*)+\ldots$$
$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x},\mathbf{u},t) = \mathbf{g}(\mathbf{x}^*,\mathbf{u},t) + \frac{\partial \mathbf{g}}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\mathbf{x}^*}(\mathbf{x}-\mathbf{x}^*)+\ldots \tag{7.92}$$

Potem lahko spoznavnost določimo z enakim testom kot v primeru linearnih sistemov, t.j. z analizo ranga matrike opazljivosti **O** pri čemer vzamemo $\mathbf{A}(t) = \partial\mathbf{f}/\partial\mathbf{x}$ in $\mathbf{C}(t) = \partial\mathbf{g}/\partial\mathbf{x}$. En možen način ocenjevanja stanj v okolici točke $\boldsymbol{x}^*$ je tim. razširjeni opazovalnik, ki ima naslednjo obliko:

$$\dot{\hat{\mathbf{x}}}(t) = \mathbf{f}(\hat{\mathbf{x}},\mathbf{u},t) + \mathbf{L}(\mathbf{y}(t) - \mathbf{g}(\hat{\mathbf{x}},\mathbf{u},t)) \tag{7.93}$$

pri čemer matriko ojačenj **L** računamo na podoben način kot pri linearnih sistemih, le da namesto matrik **A** in **C** uporabljamo matrike prvih odvodov v izrazu (7.92).

∇

Primer 7.9: *Ocenjevanje gostote snovi v šaržnem kemijskem reaktorju*

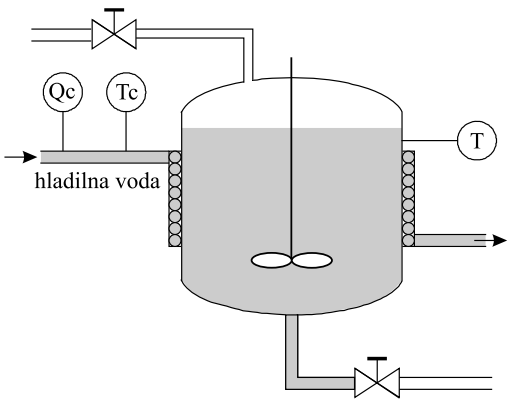
Vzemimo šaržni reaktor, v katerem poteka eksotermna ireverzibilna reakcija prvega reda. Naj bo rezultat reakcije produkt B, ki nastane iz reaktanta A ($A \xrightarrow{k} B$), pri čemer pomeni *k* hitrost reakcije. Toplota nastala med reakcijo, se odvaja s hladilno tekočino, ki se pretaka skozi cevi, ki so nameščene v plašču reaktorja (Sl. 7.37). Naj bo temperatura hladilne tekočine T_c, njen pretok pa Q_c. Zaradi enostavnosti privzamemo, da je pretok ves čas konstanten. Zaradi poenostavitve modela predpostavimo še, da je toplotni upor skozi hladilne cevi zanemarljiv, da je toplotna kapaciteta hladilne tekočine velika, ter da je temperatura T_j enaka povsod na hladilnem plašču. Na osnovi teh predpostavk imamo $T_j=T_c$. Dinamika sistema je opisana z nelinearno diferencialno enačbo drugega reda:

$$\frac{dC_A}{dt} = -\alpha C_A e^{-E/RT} \tag{7.94}$$

$$\frac{dT}{dt} = -\frac{\lambda}{\rho C_p}\alpha C_A e^{-E/RT} - UA(T - T_j) \tag{7.95}$$

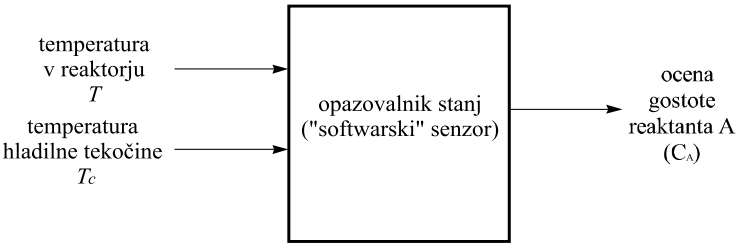
kjer je (V) volumen tekočine v reaktorju, (U) celokupen koeficient prenosa toplote, (A) površina plašča, (λ) specifična toplota reakcije, (α) Arrheniusova konstanta, (E) aktivacijska energija, (R) konstanta idealnega plina, (ρ) gostota tekočine, (C_p) toplotna kapaciteta tekočine v reaktorju. Pri tem so koeficienti modela λ, ρ, C_p in U neodvisni od temperature v reaktorju T.

Proces je definiran z dvema stanjema, t.j. temperaturo v reaktorju T in gostoto C_A reaktanta A. Temperatura T_c hladilne tekočine predstavlja vhod v sistem. Med reakcijo se meri le temperatura v reaktorju. Zanima pa nas tudi gostota C_A, ker le-ta odraža učinkovitost reakcije in kvaliteto produkta (čim več komponente B ob čim manjši količini preostale komponente A v čim krajšem času).



Sl. 7.37. Hlajen šaržni kemijski reaktor

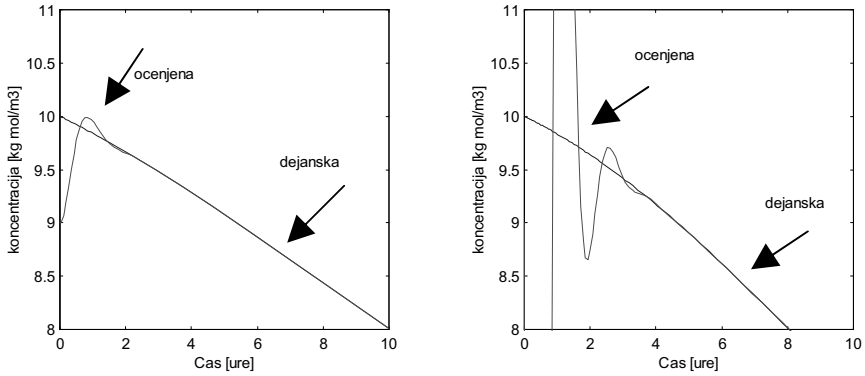
Namen zgleda je ilustrirati praktični pomen ocenjevanja stanj v industrijskih procesih. Ker nimamo merilnika koncentracije, je potrebno informacijo o gostoti C_A rekonstruirati iz poteka temperature s pomočjo modela procesa. Na ta način model prevzame vlogo "softwarskega" senzorja (Sl. 7.38).



Sl. 7.38. Razširjeni opazovalnik stanj v vlogi "softwarskega" senzorja.

Za rekonstrukcijo koncetracije smo uporabili razširjeni opazovalnik (7.93). Matriko ojačenj **L** smo dobili po linearizaciji modela (7.94) in (7.95) okoli vrednosti $C_A^*=5$ in $T^*=T_c=298^oK$. Na Sl. 7.39a sta podana poteka dejanske koncentracije in njene ocenjene vrednosti. Vidimo, da opazovalnik hitro "ujame" dejansko vrednost koncentracije, kar predstavlja kvalitetno informacijo za operaterja. Žal, zaradi nelinearnosti procesa, opazovalnik zgubi sposobnost tako hitre rekonstrukcije v primeru, če se temperatura hladilne tekočine občutneje spremeni. Na Sl. 7.39b sta narisana odziva v primeru, ko se le-ta dvigne za 4 ^oK.

Pričakujemo torej lahko, da bo razširjeni opazovalnik dajal dobre odzive le v okolici nominalne temperaure hladilne tekočine (izkaže se, da začetne vrednosti koncentracije ne vplivajo bistveno na kvaliteto rekonstrukcije). Če bi želeli dobiti dobre odzive v širšem območju temperature hladilne tekočine, potem je treba vzeti nelinearni opazovalnik stanj (Muske in Edgar, 1997).



Sl. 7.39. Časovna odziva dejanske koncentracije reaktanta A in njegove ocenjene vrednosti
a) v nominalnem režimu (Tc=T=298oK)*
b) izven nominalnega režima (Tc=302oK).

Δ

7.3.1.2 Odkrivanje napak v procesih

Ostra konkurenca na trgih vsiljuje zahteve po večji kakovosti produktov ter večji učinkovitosti proizvodnje. Zgolj dobra regulacija tega še ne zagotavlja. Za to je potrebno precej več, predvsem pa izpopolnjen nadzor procesa. Med drugim naj bi ta omogočal pravočasno odkrivanje morebitnih napak v njegovem delovanju. Kljub vse večji zanesljivosti sodobnih industrijskih tehnologij, ki se kaže tudi v manjšem številu okvar, možnost za nastop le-teh še vedno obstaja. V mnogih panogah so učinki napak lahko povezani z velikanskimi stroški (npr. prekinitev proizvodnje v valjarni zaradi odpovedi električnega motorja, ki poganja valje, ali pa zmanjšanje kvalitete produkta v kemijskem reaktorju zaradi napake v senzorju), da o posledicah okvar v kakšni jedrski elektrarni niti ne govorimo.

Zgodnje odkrivanje napak ni vedno preprosto, zlasti ne, kadar gre za napake nizkih intenzitet *znotraj regulacijske zanke.* Delovanje regulacijske zanke preprosto zakrije njihove učinke. Takšne napake je težko odkriti zgolj z opazovanjem sprememb v merjenih signalih. Za bolj učinkovito odkrivanje napak potrebujemo podrobnejšo obdelavo vzorčenih signalov iz procesa na osnovi *modela* procesa. Žal obstoječi komercialni sistemi SCADA tega ne omogočajo. Trenutno zmorejo le navadno proženje alarmov v trenutkih, ko signal prekorači neke predpisane meje, kar je koristno le v primeru večjih izpadov. Tovrstni sistemi tudi ne znajo avtomatsko poiskati možnih vzrokov za alarme; le-to je zaenkrat v celoti prepuščeno operaterju.

Sodoben razvoj sistemov za zgodnje odkrivanje napak poteka v grobem v dveh smereh. Po eni strani se že pojavlja nova generacija "inteligentnih" merilnih in izvršnih sistemov, ki imajo vgrajeno sposobnost samodejnega odkrivanja napak (Henry, 1994). To pomeni, da operater nima na razpolago le vrednosti izmerjene veličine, temveč tudi

informacijo o tem, koliko je le-ta verodostojna. Po drugi strani pa so seveda za odkrivanje napak, ki nastajajo na procesni opremi, še vedno potrebni celovitejši pristopi. Široko dostopni komercialni produkti tega ne podpirajo, najbrž zaradi dejstva, da vsak tehnološki proces ponavadi terja zelo specifične rešitve. Če že obstajajo, potem take rešitve zasledimo kot namenske programske module, integrirane v kakšnega od obstoječih SCADA sistemov (glej Sl. 7.40).

Postopek odkrivanja napak v procesih poteka običajno v dveh korakih. V prvem koraku je potrebno najprej *zaznati* napako (angl. *detection*). Pri tem gre za zaznavanje odstopanja procesnih signalov od nominalnih vrednosti. V drugem koraku je potrebno ugotoviti mesto napake, zato temu pravimo *lokalizacija napake* (angl. fault isolation). V nekaterih primerih se zahteva še tretji korak, t.j. *identifikacija napake,* ki poda oceno velikosti napake ter obliko njenega časovnega poteka. Za postopek lokalizacije in identifikacije napak se pogosto uporablja izraz *diagnostika napak*.

Življenjski cikel sistemov za odkrivanje napak

Načrtovanje diagnostičnega sistema se začne z vprašanji: "Kaj so cilji?", "Kaj je znano?" ter "Kakšne so omejitve?"

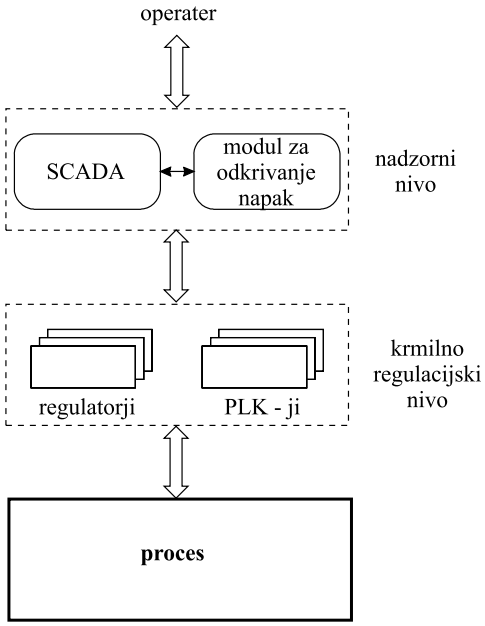
Do odgovorov lahko pridemo s celostno obravnavo vprašanj v okviru petih glavnih faz, ki so v principu enake tistim, ki smo jih opisali v okviru splošnega življenjskega cikla sistemov (poglavje 2). Te faze so:

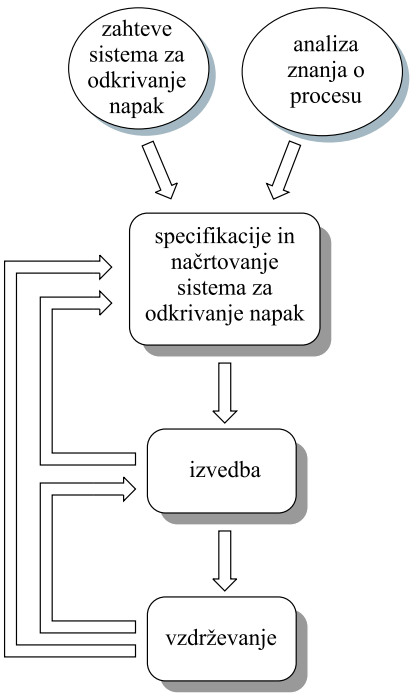
- definicija zahtev, ki jih mora zadovoljiti diagnostični sistem;
- analiza znanja o procesu;
- specifikacija in načrtovanje diagnostičnega sistema;
- implementacija in
- vzdrževanje.

Omenjeni sklop imenujemo tudi življenjski cikel sistemov za odkrivanje napak (Sl. 7.41).

Osnovni namen *definicije zahtev* je odgovoriti na vprašanji *kaj* je treba narediti in *zakaj*. Pri tem razlikujemo tri vrste zahtev:

- zahteve diagnostičnega algoritma;
- zahteve glede napak ter
- zahteve glede modela.





Sl. 7.40. Modul za odkrivanje napak kot del sistema.

Sl. 7.41. Življenjski cikel sistemov za odkrivanje napak.

V *zahtevah diagnostičnega sistema* so vsebovani cilji in funkcije, ki jih je potrebno realizirati (Leitch 1993). Naravo napak, ki naj bi jih sistem bil zmožen odkrivati, je potrebno opredeliti z *zahtevami glede napak*, saj je reševanje problemov diagnostike zelo odvisno prav od narave napak.

Pri diagnostiki igra pomembno vlogo poznavanje sistema, ki se izraža v obliki *modela*. Kakšen model bo potreben, zavisi od obsega in kvalitete znanja o sistemu. Ali bodo potrebni kvalitativni ali kvantitativni modeli, je potrebno opredeliti v *zahtevah glede modela*.

Nadaljnjo globljo obdelavo zahtev glede modela dosežemo z *analizo znanja o procesu.* Le-ta vsebuje analizo dostopnih podatkov in izkustvenega znanja o procesu, ki ga je potrebno urediti v obliki modela procesa.

Namen faze *specifikacije* je predlagati rešitev diagnostičnega problema z upoštevanjem vseh prej omenjenih zahtev. Namen *načrtovanja* pa je izdelati rešitev, ki bo primerna za *implementacijo*.

Vzdrževanje diagnostičnih sistemov je faza, ki se ji ni mogoče izogniti v življenjskem ciklu, saj vsaka tehnološka sprememba sistema zahteva tudi ustrezne modifikacije v programski kodi, ki opravlja diagnostiko

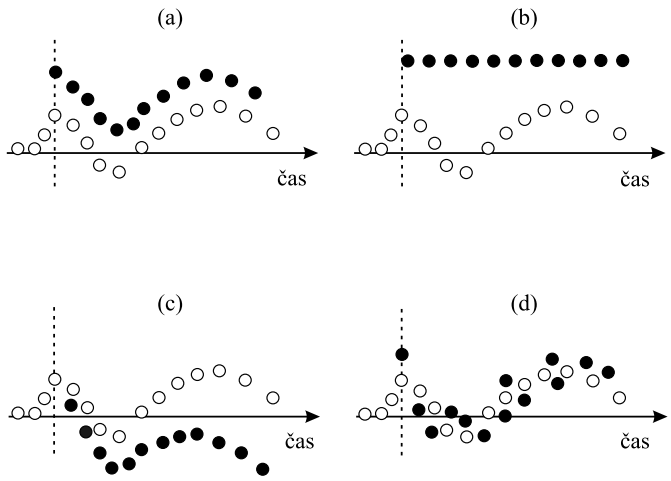
V nadaljevanju bomo razčlenili nekatere važnejše lastnosti življenjskega cikla diagnostičnega sistema.

Zahteve diagnostičnega sistema

Zahteve glede napak

Kakšne napake naj bo sistem sposoben zaznati? To je pomembno vprašanje, saj ni metode, ki bi bila primerna za odkrivanje vseh možnih napak. Zato je potrebno že na začetku definirati določene predpostavke o naravi napak, kot so:

Izvor napake. Napake se lahko pojavijo na senzorjih, aktuatorjih in procesni opremi, lahko pa se pripetijo tudi kot posledica napačnih posegov operaterja (npr. pomotoma odprt ročni ventil). Najpogostejše napake na senzorjih so pristranskost (bias), popolna odpoved, lezenje in zmanjšanje precizije (gl. Sl. 7.42)).



Sl. 7.42. Vrste napak v izmerjenih podatkih, (O) dejanske vrednosti (•) podatki iz senzorjev: (a) pristranskost, (b) odpoved, (c) lezenje in (d) zmanjšanje precizije

Na mehanskih aktuatorjih imamo pogosto opravka s trenjem, mrtvo cono (npr. pri prenosih). Na procesni opremi so pogoste zamašitve v pretočnih ceveh, puščanje, nabiranje oblog in podobno.

Dinamično obnašanje napake. Pojem dinamike napake nam pove, kako se intenziteta napake spreminja v odvisnosti od časa. V tem smislu ločimo:

- nenadne napake (pojavijo se nenadoma in trajajo ves čas do popravila);
- napake, ki se počasi razvijajo (npr. postopno mašenje cevi);
- občasne napake (pojavijo se v kratkih presledkih in potem izginejo, npr. slab kontakt).

Enojne/večkratne napake. Največkrat se v realnih sistemih zgodi le ena napaka naenkrat. Včasih pa se kajpak več napak pojavi hkrati zaradi nekega skupnega vzroka (npr. požar).

Zahteve diagnostičnega algoritma

Delovanje diagnostičnega algoritma ocenjujemo na podlagi uspešnega zaznavanja napak in uspešne lokalizacije. Pri zaznavanju so pomembne naslednje lastnosti:

Čim manjše število lažnih alarmov: lažni alarm pomeni, da je proces sicer v normalnem stanju, vendar algoritem "misli", da je v njem prisotna napaka.

Čim manjše število spregledanih alarmov: le-ti se zgodijo takrat kadar algoritem "molči", v resnici pa je v procesu prisotna napaka.

Čim krajši čas detekcije: gre za čas, ki preteče od pojava napake do trenutka, ko jo algoritem dejansko zazna.

Robustnost: sposobnost algoritma, da reagira le na spremembe v signalih, ki so posledica dejanskih napak in ne pogreškov modela, ali drugih motenj.

Idealno bi bilo, če bi zaznavanje potekalo brez lažnih in spregledanih alarmov na eni strani ter brez zakasnitve, na drugi strani. Žal pa hitra detekcija zahteva povečano občutljivost algoritma, kar pomeni manjšo robustnost na motnje. To posledično vodi k lažnim alarmom (npr. zaradi šuma v signalih).

Naloga lokalizacije je generirati listo "osumljenih" napak. Pri tem pa želimo doseči predvsem naslednje:

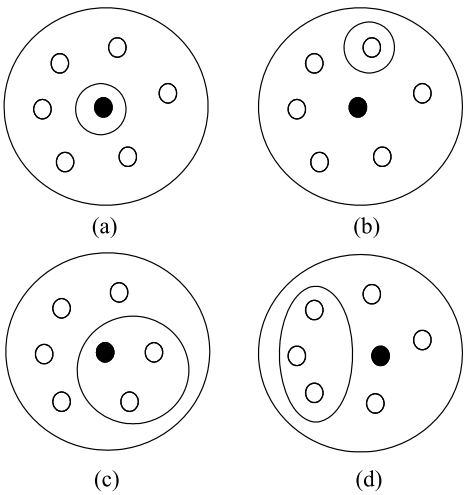
Natančnost: lokalizacija je natančna, kadar je dejanska napaka vsebovana na listi osumljenih napak.

Ločljivost: gre za zmožnost diagnostičnega sistema, da na osnovi razpoložljivih informacij iz procesa poda *minimalno* število osumljenih napak.

Diagnostična stabilnost: sposobnost diagnostičnega sistema, da poda množico osumljenih napak, ki ostane nespremenjena, ne glede na potek signalov v procesu.

Zanesljivost: sposobnost diagnostičnega sistema, da poda točno diagnozo za vse napake, vključno s tistimi, o katerih nima "izkušenj" iz preteklosti.

Lokalizacija z visoko ločljivostjo rezultira v minimalnem številu možnih napak pri danih podatkih iz procesa. Zaradi tega pa lahko trpi natančnost. Zato je v splošnem vedno potreben kompromis. Medsebojno povezavo ločljivosti in natančnosti ilustrira Sl. 7.43.



Sl. 7.43. Kvaliteta lokalizacije: odnos med ločljivostjo in natančnostjo ((O) - osumljena napaka, (•) - dejanska napaka) (a) maksimalna ločljivost in maksimalna natančnost (b) največja ločljivost ob napačni diagnozi (c) slaba ločljivost ampak ob natančni diagnozi (d) slaba ločljivost in nenatančna diagnoza.

Znanje o procesu

Kako natančno je potrebno poznati proces, da dosežemo zahtevano kvaliteto diagnostike?

Odgovor zavisi od postavljenih zahtev. V vsakem primeru izrazimo znanje o procesu v obliki modelov, ki so lahko na različni stopnji natančnosti t.j. deterministični modeli (statični, dinamični, linearni, nelinearni), naključni, semi-kvantitativni, modeli tipa reda velikosti, kvalitativni in čisto hevristični (Boolean) modeli. Da bi dosegli čim višjo diagnostično natančnost, je potrebno včasih združevati vso razpoložljivo znanje oziroma modele na različnih nivojih abstrakcije.

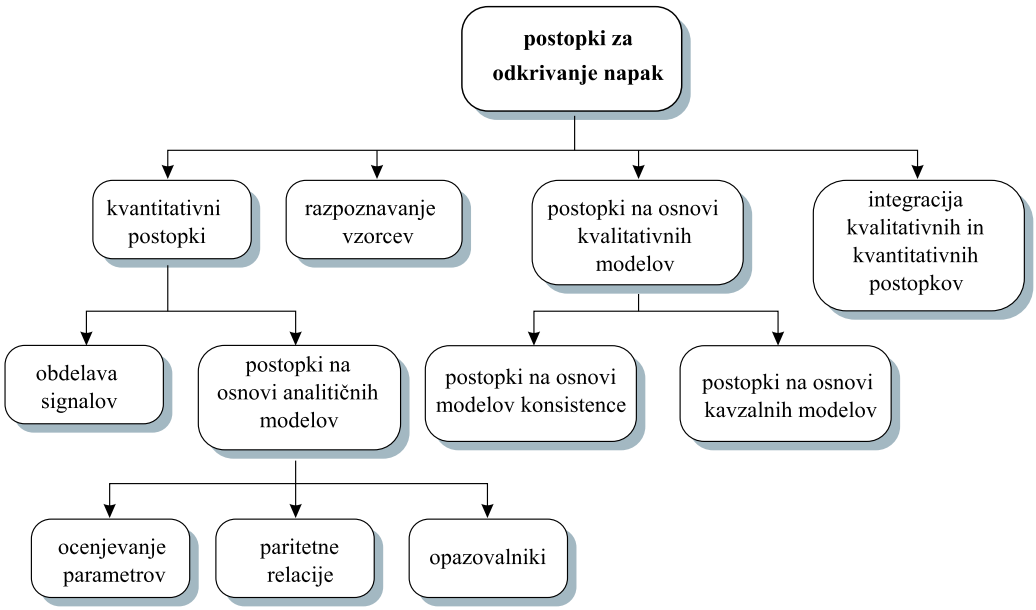
<u>Postopki za odkrivanje napak</u>

Zahteve diagnostičnega algoritma, zahteve glede napak ter zahteve glede modela določajo tip uporabljenega diagnostičnega postopka. Ker so zahteve močno odvisne od tehnologije procesa in razpoložljivega znanja, je dostikrat potrebno uporabiti različne postopke za dosego cilja. Na Sl. 7.44 je podana preprosta porazdelitev diagnostičnih algoritmov na podlagi stopnje natančnosti uporabljenih modelov procesa. Več podrobnosti zasledimo v delu Pattona in sodel. (1989).

Kvantitativne postopke lahko razdelimo na več skupin, odvisno od tega na kakšen način računamo simptome. (Simptom pomeni odstopanje neke veličine (izmerjene ali izračunane) od zaloge vrednosti, ki definirajo normalno stanje procesa.)

Pri metodah, ki temeljijo na *obdelavi signalov*, pridemo do simptomov tako, da na signalih neposredno izvajamo relativno preproste transformacije kot so povprečenje, računanje trendov, statistika amplitudne porazdelitve, spektralna močnostna gostota, itd. Eksplicitni model procesa pri tem ni potreben.

Pri *postopkih na osnovi analitičnih modelov* pa je seveda eksplicitni model potreben. Te postopke bomo kasneje nekoliko podrobneje obravnavali.



Sl. 7.44. Pregled osnovnih postopkov za odkrivanje napak

Pristopi na podlagi *razpoznavanja vzorcev* uporabljajo neposredno zvezo med napakami in simptomi. Njihova prednost je, da vzorci ne odražajo interne strukture procesa, zato modeliranje ni potrebno. Vzorci so lahko podani eksplicitno, v obliki logičnih pravil ali implicitno, npr. v obliki nevronske mreže. Taki pristopi so primerni le, ko je na razpolago dovolj podatkov iz preteklega delovanja procesa. Vendar pa nimajo sposobnosti razpoznati napake, ki niso zabeležene v zgodovini procesa.

Postopki, ki temeljijo na *kvalitativnih modelih,* omogočajo opis obnašanja procesa na visokem nivoju abstrakcije. Razlikujemo dve glavni skupini modelov:

- modeli na osnovi konsistence ter
- kavzalni modeli.

V prvo skupino štejemo postopke, ki uporabljajo formalizem de Kleer in Browna, kvalitativno simulacijo (QSIM), njeno mehko različico Fusim, semi-kvantitativne modele in modele dogodkov. Širši pregled postopkov kvalitativnega modeliranja lahko

zasledimo v knjigi Welda in de Kleera (1989).

Glavna lastnost kavzalnih modelov so kavzalne povezave med napakami in procesnimi veličinami. Splošno znane so tehnike, ki temeljijo na predznačenih usmerjenih grafih (Signed Directed Graphs). Tovrstni modeli skušajo podati širjenje napak z uporabo logičnih relacij, ki kažejo, kako nastane odstopanje neke procesne veličine iz odstopanj drugih veličin ali napak v procesu.

Odkrivanje napak s pomočjo analitičnih modelov

Ta razred metod uporablja analitične modele procesa. Celoten diagnostični postopek sestoji iz treh korakov:

- generiranje residualov;
- vrednotenje residualov in
- sklepanje o vrsti napake.

V prvem koraku generiramo signale, imenovane *residuali*. Njihova glavna lastnost je, da so približno enaki nič, ko napak ni, in različni od nič, ko pride do napake. Biti morajo tudi neodvisni od načina delovanja procesa.

Namen *vrednotenja* je ugotoviti kvalitativno vrednost residuala, t.j. ali je nič oz. različen od nič. S *sklepanjem* na osnovi kvalitativnih vrednosti residualov skušamo določiti mesto napake.

Ločimo tri glavne pristope generiranja residualov:

- ocenjevanje parametrov;
- pristopi na osnovi opazovalnikov stanj in
- pristopi na osnovi paritetnih relacij.

Pristop na osnovi ocenjevanja parametrov temelji na dejstvu, da napaka povzroči spremembe fizikalnih parametrov modela. Residuale generiramo z ocenjevanjem parametrov in primerjanjem ocen z nominalnimi vrednostmi.

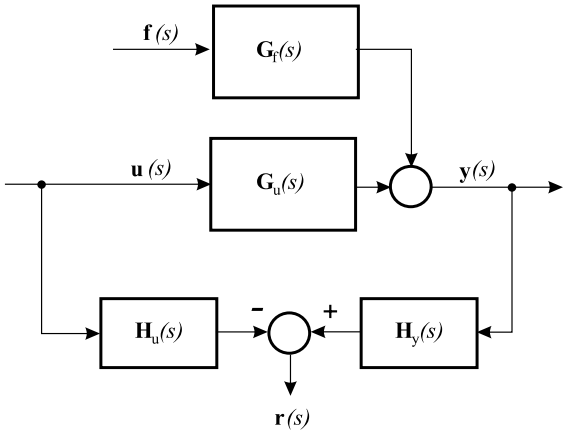
Ko gre za linearne sisteme, lahko pristope na osnovi opazovalnikov stanj in tiste na osnovi paritetnih relacij predstavimo s splošno strukturo, ki je prikazana na Sl. 7.45.

Pri tem predpostavimo, da ima model sistema naslednjo obliko:

$$\mathbf{y}(s) = \mathbf{G}_{\mathrm{u}}(s)\mathbf{u}(s) + \mathbf{G}_{\mathrm{f}}(s)\mathbf{f}(s) \tag{7.96}$$

kjer **u, y** in **f** predstavljajo vektorje vhodov, merljivih izhodov iz sistema ter napak. Z $\mathbf{G}_{\mathrm{u}}$ označimo prenosno funkcijo med vhodi in izhodi procesa, z $\mathbf{G}_{\mathrm{f}}$ pa prenosno funkcijo med napakami in izhodi procesa. Generator residualov lahko potem zapišemo v obliki:

$$\mathbf{r}(s) = -\mathbf{H}_{\mathrm{u}}(s)\mathbf{u}(s) + \mathbf{H}_{\mathrm{y}}(s)\mathbf{y}(s) = \left(-\mathbf{H}_{\mathrm{u}}(s) + \mathbf{H}_{\mathrm{y}}(s)\mathbf{G}_{\mathrm{u}}(s)\right)\mathbf{u}(s) + \mathbf{H}_{\mathrm{y}}(s)\mathbf{G}_{\mathrm{f}}(s)\mathbf{f}(s) \tag{7.97}$$



Sl. 7.45. Splošna struktura generatorja residualov

Če dosežemo, da velja $\mathbf{H}_u(s)+\mathbf{H}_y(s)\mathbf{G}_u(s)\equiv 0$, postanejo residuali odvisni le od napak **f**. Iz (7.96) sledi

$$\mathbf{r}(s) = \mathbf{H}_{\mathrm{y}}(s)\mathbf{G}_{\mathrm{f}}(s)\mathbf{f}(s) \tag{7.98}$$

Relacija (7.98) je zelo splošna in jo zasledimo tako pri metodah, ki temeljijo na paritetnih relacijah kot tudi pri metodah na osnovi opazovalnika.

Pri metodah, ki temeljijo na *paritetnih relacij,* za generacijo residuala uporabimo dinamično odvisnost med vhodom in izhodom. Tako dobimo relacije, ki so neodvisne od vhodov v proces.

Osnovna ideja pristopa z opazovalniki je, da ocenimo izhod sistema iz razpoložljivih meritev. Posplošeni Luenbergerjev opazovalnik ima za ta problem naslednjo obliko:

$$\begin{aligned} \dot{\boldsymbol{\omega}} &= \mathbf{F}\boldsymbol{\omega} + \mathbf{K}\mathbf{y} + \mathbf{J}\mathbf{u} \\ \mathbf{r} &= \mathbf{L}_1\boldsymbol{\omega} + \mathbf{L}_2\mathbf{y} + \mathbf{L}_3\mathbf{u} \end{aligned} \tag{7.99}$$

pri čemer dinamiko opazovalnika nastavljamo z matriko **F**.

Residuali za lokalizacijo napake

Z analizo enačbe (7.98) lahko ugotovimo, kateri residuali postanejo različni od nič ob prisotnosti neke napake. Temu seznamu pravimo *koda* napake. Vse kode skupaj tvorijo *množico kod*, ki jo lahko predstavimo v obliki matrike (glej Tabela 7.1). Ločljivost napak zelo zavisi od njihovih kod. Pri tem razlikujemo tri tipe množic kod:

- neločljive množice;
- slabo ločljive in

- strogo ločljive.

Vsi trije primeri so ilustrirani v spodnji tabeli. V prvem primeru imata napaki F_1 in F_2 enaki kodi; zato ju ne moremo enolično lokalizirati, saj so v obeh primerih vsi trije residuali različni od nič. V drugem primeru so tri kode slabo ločljive, ker že sprememba enega bita v kodi (npr. zaradi šuma), povzroči nastanek kode, ki pripada drugi napaki. Na primer, če je v sistemu prisotna napaka F_2, in če se npr. zaradi šuma, prvi bit v drugem stolpcu spremeni od 0 na 1 (primer b)), dobimo kodo, ki je identična kodi za napako F_1.

S primerno transformacijo originalnega modela je možno dobiti t.i. *strukturirane residuale*, ki vodijo k strogo ločljivim množicam kod (glej primer c)) in tako omogočajo enolično lokalizacijo celo večkratnih napak. Torej, strukturirani residual se odziva le na *določene* napake in je *neobčutljiv* na vse ostale napake.

Tabela 7.1. Trije tipi množic kod: a) neločljive b) slabo ločljive in c) strogo ločljive množice

	F_1	F_2	F_3	F_1	F_2	F_3	F_1	F_2	F_3
r_1	1	1	0	1	0	0	1	0	0
r_2	1	1	0	1	1	0	0	1	0
r_3	1	1	1	1	1	1	0	0	1
	a)			b)			c)		

Vrednotenje residualov

V resnici residuali niso nikoli popolnoma enaki nič, četudi v procesu ni prisotna napaka. Zato je potrebno ovrednotiti residual, t.j. pretehtati ali je le-ta "enak nič" ali je "različen od nič". Da bi se pri tem izognili napačnim in spregledanim alarmom (zaradi šuma, napak modeliranja in drugih motenj), največkrat uporabljamo metodo povprečenja residualov in testiranja mejnih vrednosti. Ko pričakujemo šibke spremembe residualov, so najbolj uporabni statistični testi (CUSUM, Generalized Likelihood Ratio test).

Na primer, če uporabimo test mejnih vrednosti, nam da lahko vrednotenje residualov sledeč rezultat:

$$\mu_i = \begin{cases} 1 & |r_i| > prag \\ 0 & |r_i| \le prag \end{cases}$$

kjer je μ_i kvalitativna vrednost residuala r_i.

Sklepanje o vrsti napake

Cilj tega koraka je najti stopnjo zaupanja za vsako možno napako na osnovi

ovrednotenih residualov.

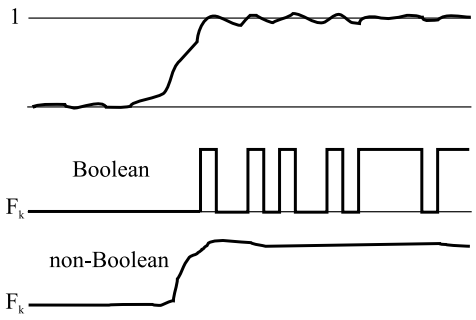
Če po vrednotenju residualov dobimo množico binarnih kod, sklepamo na izvor napake s primerjavo dobljenih kod s tistimi iz celotne množice kod. Na žalost, naletimo na težave, če simptomom pripisujemo le binarne vrednosti, še posebno, če je v sistemu prisoten šum. Tedaj namreč že majhne spremembe residualov, v bližini mejnih vrednosti, lahko povzročijo pogoste spremembe binarnih vrednosti simptomov. To nadalje vodi k pogostemu spreminjanju diagnostičnih rezultatov, kar je znano kot diagnostična nestabilnost (Sl. 7.46).

Namesto binarnih vrednosti 0 in 1, s katerimi opišemo prekoračitev mejne vrednosti, je bolje uporabljati *stopnjo prekoračitve,* ki zavzema vrednosti iz intervala [0,1]. Stopnjo prekoračitve mejne vrednosti lahko dobimo z "mehčanjem" (gl. Sl. 7.47), npr. s sledečo funkcijo residuala

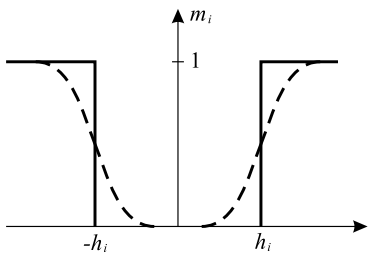
$$m_i(r_i) = \frac{b}{1 + \frac{1-a}{a}\left(\frac{h_i}{r_i}\right)^{2\gamma}} \tag{7.100}$$

kjer je γ parameter mehčanja, a je verjetnost pripisana vrednosti praga h_i in b je maksimalna mera verjetnosti danega residuala. Funkcija (7.100) je osnova za *aproksimativno sklepanje.*

Rezultat aproksimativnega sklepanja je spisek kandidatov s pripisano verjetnostjo $0 \le m(F_i) \le 1$. To je "mehki" rezultat, v nasprotju s "trdim" rezultatom, ki ga dobimo z uporabo binarnih podpisov. Slednji konča tako, da za vsako napako F_i pove, ali je prisotna ($m(F_i)$=1) ali ne ($m(F_i)$=0).



Sl. 7.46. Ilustracija diagnostične nestabilnosti

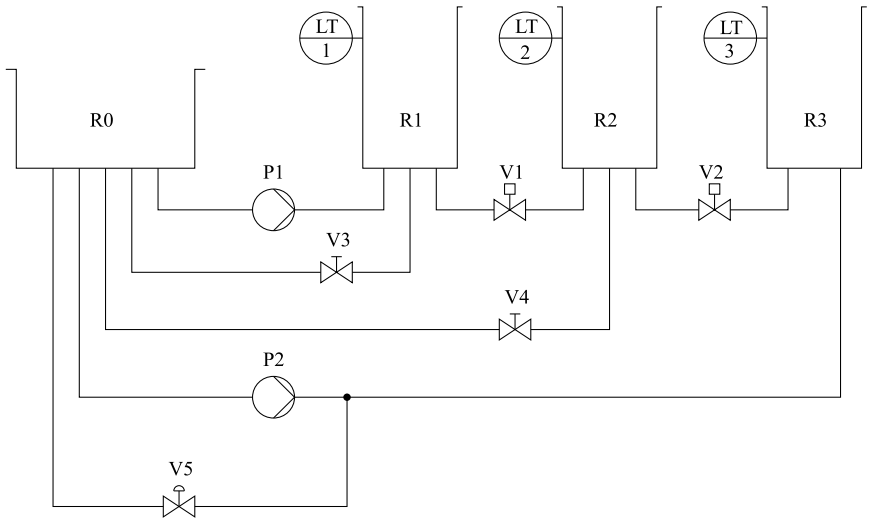


Sl. 7.47. Ilustracija "mehkega" (črtkano) in "trdega" vrednotenja prekoračitve praga (polna linija)

∇

Primer 7.10: *Odkrivanje napak na sistemu treh posod*

Predstavili bomo primer odkrivanja napak na osnovi analitičnega modela, pri čemer bo kot objekt služil laboratorijski proces treh posod, ki smo ga predstavili že v poglavju o procesih. Shema procesa je prikazana na Sl. 7.48.



Sl. 7.48. Shema procesa.

Kot smo že omenili, proces sestavljajo tri posode R1, R2 in R3, ki so povezane s cevmi, po katerih se pretaka voda. V našem primeru imata posodi R1 in R3 vlogo zbirne posode za posodo R2, R0 pa je zbirni rezervar za cel sistem. Vsebina iz R1 in R3 se zmeša v R2 in potem pretaka nazaj v R0. Nivo v posodi R1 se regulira s hitrostjo vrtenja črpalke ω_1, medtem ko se nivo v R3 regulira s signalom u_5 na regulacijskem ventilu V5 (črpalka P2 ves čas deluje pri konstantni hitrosti ω_2). Iz procesa so dostopni naslednji signali: nivoji h_{1m}, h_{2m}, h_{3m} v posodah R1, R2 in R3, ter ω_1 in u_5.

Zahteve diagnostičnega algoritma

Spisek možnih napak v procesu podaja Tabela 7.2.

Tabela 7.2. Seznam napak

#	Vrsta napake
F1	Puščanje v R1
F2	Zamašitev V1
F3	Napaka v senzorju LT 1
F4	Zamašitev V4
F5	Napaka v senzorju LT 2
F6	Napaka v senzorju LT 3
F7	Zamašitev V2

Algoritem naj bo sposoben odkriti vsaj enojne nenadne napake. Velikost napake, na katero naj diagnostični algoritem še reagira, naj bo 20% celotnega območja napake. Na primer, 5cm zamika pri senzorju nivoja v posodi R3 je približno 20% nivoja v delovni točki.

Znanje o procesu (model procesa in napak)

Obnašanje procesa je opisano s podrobnim semi-fizikalnim modelom. Celoten model skupaj z napakami je predstavljen v prostoru stanj na naslednji način:

$$
\begin{aligned}
\dot{h}_1 &= \frac{1}{A_1}\left(k_{1F}(h_{1stat}(\omega_1)-h_1)+k_{2F}\sqrt{h_{1stat}(\omega_1)-h_1}-(1-f_2)k_{V1}\sqrt{\rho g(h_1-h_2)}\right)-f_1\\
\dot{h}_2 &= \frac{1}{A_2}\left((1-f_2)k_{V1}\sqrt{\rho g(h_1-h_2)}+(1-f_7)k_{V2}\sqrt{\rho g(h_3-h_2)}-(1-f_4)k_{V4}\sqrt{\rho g h_2}\right)\\
\dot{h}_3 &= \frac{1}{A_3}\left(f_{1F}(s_5)(h_{3stat}(s_5)-h_3)+f_{2F}(s_5)\sqrt{h_{3stat}(s_5)-h_3}-(1-f_7)k_{V2}\sqrt{\rho g(h_3-h_2)}\right)\\
h_{1m} &= h_1+f_3\\
h_{2m} &= h_2+f_5\\
h_{3m} &= h_3+f_6
\end{aligned}
\tag{7.101}
$$

kjer je $\mathbf{x}=[h_1,h_2,h_3]^T$ vektor spremenljivk stanja, $\mathbf{u}=[\omega_1,s_5]^T$ vektor regulirnih vhodnih signalov in $\mathbf{f}=[f_1,f_2,f_3,f_4,f_5,f_6,f_7]^T$ vektor napak v procesu.

Načrtovanje algoritma za zaznavanje in odkrivanje napak

Generiranje residualov

Najprej bomo načrtali množico treh residualov na osnovi sledečega opazovalnika stanj:

$$
\begin{aligned}
\dot{\hat{h}}_1 &= \alpha_1(h_{1m}-\hat{h}_1)+\frac{1}{A_1}\left(k_{1F}(h_{1stat}(\omega_1)-h_{1m})+k_{2F}\sqrt{h_{1stat}(\omega_1)-h_{1m}}-k_{V1}\sqrt{\rho g(h_{1m}-h_{2m})}\right)\\
\dot{\hat{h}}_2 &= \alpha_2(h_{2m}-\hat{h}_2)+\frac{1}{A_2}\left(k_{V1}\sqrt{\rho g(h_{1m}-h_{2m})}+k_{V2}\sqrt{\rho g(h_{3m}-h_{2m})}-k_{V4}\sqrt{\rho g h_{2m}}\right)\\
\dot{\hat{h}}_3 &= \alpha_3(h_{3m}-\hat{h}_3)+\frac{1}{A_3}\left(f_{1F}(s_5)(h_{3stat}(s_5)-h_{3m})+f_{2F}(s_5)\sqrt{h_{3stat}(s_5)-h_{3m}}-k_{V2}\sqrt{\rho g(h_{3m}-h_{2m})}\right)
\end{aligned}
\tag{7.102}
$$

ki ga dobimo z razširitvijo Luenbergerjevega opazovalnika za linearne sisteme. Le-ta je potreben zato, ker odvodov h_1, h_2 in h_3 ne moremo neposredno meriti. Torej jih je treba oceniti. Potem lahko zapišemo:

$$
\begin{aligned}
r_1 &= h_{1m} - \hat{h}_1 \\
r_2 &= h_{2m} - \hat{h}_2 \\
r_3 &= h_{3m} - \hat{h}_3
\end{aligned}
\tag{7.103}
$$

Opazimo lahko, da drugi členi na desni strani vsebujejo samo merjene veličine. Parametri α_1, α_2 in α_3 določajo dinamiko opazovalnika. Z višanjem vrednosti α_1 na primer, povečujemo hitrost rekonstrukcije stanja h_1 in tako skrajšamo zakasnitev pri zaznavanju napak. Hkrati pa se zmanjša občutljivost residuala r_1 na napake z desne strani enačbe (7.101). Zato je potrebno najti kompromis med hitrostjo in občutljivostjo. V našem primeru smo izbrali α_1=0.5, α_2=0.25 in α_3=0.05.

Oblika odziva residualov na napake je odvisna od časovnega poteka napake. Omejili se bomo samo na stacionarne lastnosti residualov, ki jih povzročajo nenadne napake. Množica kod (incidenčna matrika) je za naš primer prikazana v Tabeli 7.3.

Očitno napak F_2 in F_3 ni možno lokalizirati zaradi njunih enakih kod. Enako velja za F_6 in F_7.

Izbira vrednosti pragov

Zaradi prisotnosti šuma v sistemu in napak v modelu, so residuali običajno različni od nič tudi, ko v procesu ni napake. Zato mora biti mejna vrednost definirana tako, da se loči odklon residualov zaradi dejanske napake od odklona zaradi drugih vzrokov. Znano pa je tudi, da je občutljivost algoritma za zaznavanje napak zelo odvisna od vrednosti pragov pri vrednotenju residualov. Preveč nizka vrednost praga lahko poveča število nepravilnih alarmov. S pomočjo matematičnega modela (7.102) smo izbrali vrednosti pragov τ_i za residuale r_i (i=1,2,3) na osnovi analize občutljivosti:

$$
\begin{aligned}
\tau_i &= min\{\tau_{ij}, j = 1,2,...,7\} \\
\tau_{ij} &= \left(\frac{\partial \bar{r}_i}{\partial \bar{f}_j}\right) \bar{f}_j p_j
\end{aligned}
\tag{7.104}
$$

kjer so z $\bar{r}_i$ in $\bar{f}_i$ označene velikosti residuala in napake v stacionarnem stanju, $0 \le p_j \le 1$ pomeni procent od celotnega območja napake $\bar{f}$, na katero naj algoritem reagira. V danem primeru so mejne vrednosti r_1, r_2 in r_3 enake ± 0.5, ± 0.35, ± 2.

Tabela 7.3. Matrika podpisov napak za sistem 3 posod.

	F_1	F_2	F_3	F_4	F_5	F_6	F_7
r_1	1	1	1	0	1	0	0

r_2	0	1	1	1	1	1	1
r_3	0	0	0	0	1	1	1

Vrednotenje residualov in lokalizacija napak

Preden testiramo prekoračitev vrednosti pragov, filtriramo residuale z nizkoprepustnim filtrom.

Pri lokalizaciji napak uporabljamo aproksimativno sklepanje s pomočjo *metode prenosa zaupanja* (angl. Transferable Belief Model, TBM). Posebnost TBM je, da poleg seznama napak in pripadajočih verjetnosti poda še mero zaupanja v diagnostiko (tim. *moč konflikta*). To je število med 0 in 1 pri čemer 0 pomeni, da gre popolnoma zaupati diagnozi, vrednosti blizu 1 pa opozarjajo na previdnost pri interpretaciji rezultatov.

Izvedba

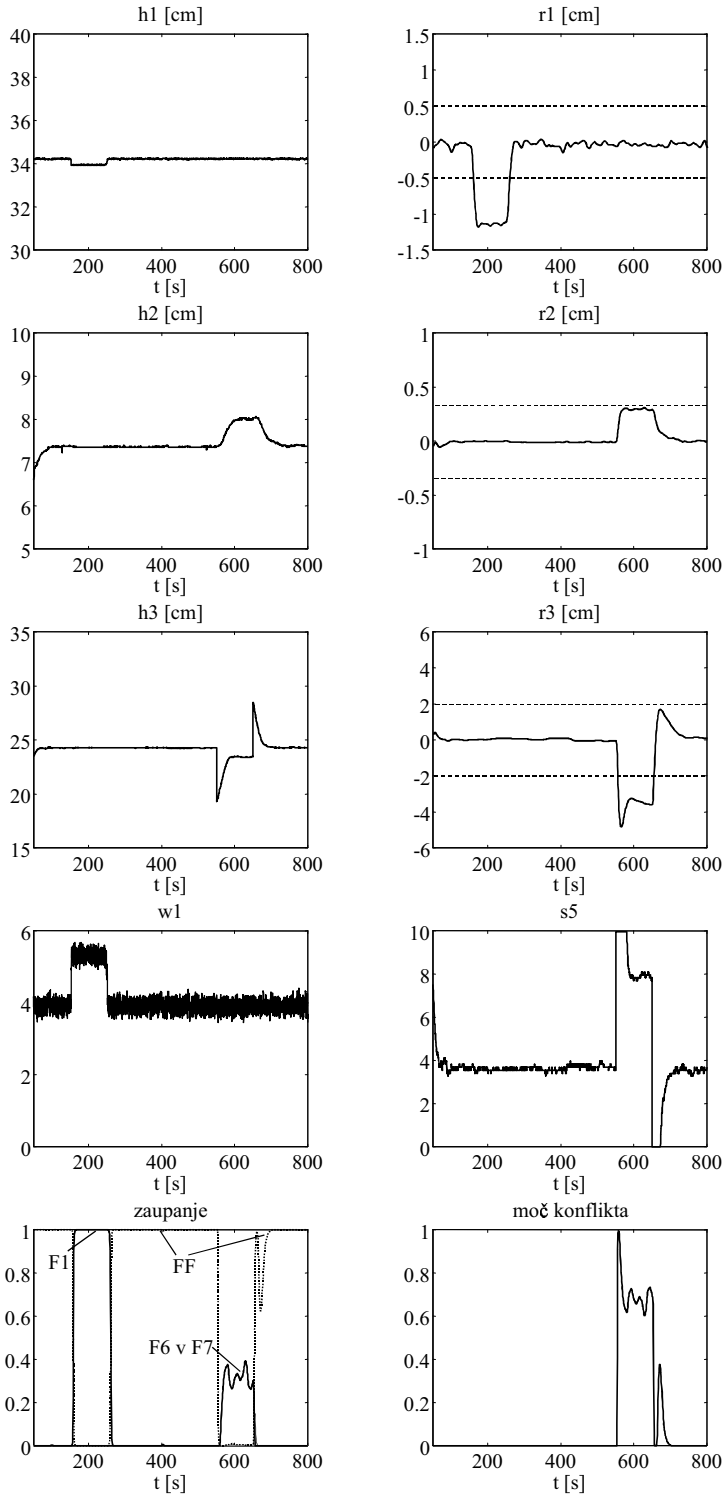
Celoten algoritem za odkrivanje napak smo izvedli v okolju Matlab/Simulink v realnem času. Odzivi so prikazani na Sl. 7.49, kjer sta bili zaporedno vneseni dve napaki, in sicer:

- puščanje posode R1 (napaka F_1) v času med 150s in 250s ter
- 5cm pristranskosti v signalu senzorja h_3 (napaka F_6) v času od 550s do 650s.

V primeru napake F_6 je ločljivost pri lokalizaciji napak slabša kot v prvem primeru. Algoritem namreč ugotovi, da gre za eno izmed napak F_6 ali F_7 ($F_6 v F_7$) in tej domnevi pripiše največjo verjetnost, saj imata F_6 in F_7 enaki kodi. Lahko opazimo, da je moč konflikta sorazmerno visoka. To pomeni, da algoritem ne more postaviti zanesljive diagnoze. To ni nenavadno, če upoštevamo, da je residual r_2 nekoliko nižji od vrednosti praga, kar je posledica sorazmerno majhne napake.

Če bi uporabili navadno "Boolovo" sklepanje, bi ugotovili, da je prišlo le do odklona v residualu r_3. Kode, v kateri je le r_3 različen od nič, ni na seznamu v Tabeli 7.3. Zato bi algoritem ugotovil, da gre za napako neznanega izvora, kar pomeni zgrešeno diagnozo. Zgled potrjuje praktično uporabnost aproksimativnega sklepanja.

Ta in podobni poskusi kažejo, da je *dober model* procesa *bistvenega* pomena za uspešno diagnostiko. Pri tem ima važno vlogo tudi število senzorjev, predvsem kar zadeva lokalizacijo napak. Premalo senzorjev lahko pomeni, da določenih napak ne moremo enolično lokalizirati. Če bi hoteli ločiti med napakami F_6 in F_7, bi morali našemu konkretnemu primeru dodati še en senzor v proces, npr. merilnik pretoka v vejo z ventilom V2.


h1 [cm]
r1 [cm]
h2 [cm]
r2 [cm]
h3 [cm]
r3 [cm]
w1
s5
zaupanje
moč konflikta
F1
FF
F6 v F7
t [s]

Sl. 7.49. Rezultati diagnostike na procesu treh posod. V času med 150 in 250s pušča posoda R1 (napaka F1), med 550 in 650s je prisotna napaka na senzorju h_3 (F6). Algoritem enoumno odkrije F1, v primeru druge napake pa ne more ločiti med F6 in F7. FF pomeni, da v sistemu ni napake.

Δ

7.3.2 Sekvenčno vodenje

Mnogo industrijskih procesov zahteva reguliranje procesne veličine, ki ima za osnovni cilj vzdrževanje procesa v delovni točki, čeprav vplivajo nanj zunanje motnje. Velikokrat pa je problem vodenja drugačen. Namesto predpisanih vrednosti posameznih veličin je predpisano zaporedje delovnih operacij, ki naj se izvršijo.

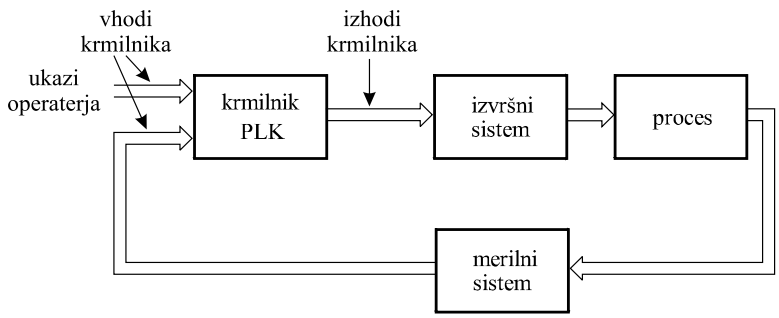
Ker je cilj sistema avtomatskega vodenja v tem primeru zagotavljanje pravilnega zaporedja operacij, govorimo o sekvenčnem vodenju. ***Sekvenčno vodenje je definirano kot vodenje procesa skozi zaporedje diskretnih stanj, s čimer dosežemo določen cilj vodenja.***

Končni cilj takšnega vodenja je pogosto proizvodnja nekega izdelka. Proizvodnja hladilnikov zahteva npr. obdelavo različnih sestavin (kovin, plastičnih mas) ter sestavljanje tako dobljenih delov v končno obliko. Pri tem si številne operacije sledijo v določenem zaporedju - naslednja operacija se izvrši šele, ko je predhodna že končana.

Diskretno stanje procesa predstavlja množica stanj vseh operativnih enot procesa. To so npr.: končna stikala, tipke, ventili, motorji. Stanja posameznih enot lahko zavzamejo le dve vrednosti in jih opisujejo izrazi kot: stikalo S1 je sklenjeno, tipka T1 je pritisnjena, ventil V5 je zaprt, motor M2 je vključen itd. Izvajanje določene operacije pomeni ustrezno kombinacijo stanj posameznih enot in ustreza enemu ali zaporedju več diskretnih stanj.

Stanje procesa se spreminja ob nastopu diskretnih dogodkov. Takšni dogodki so npr.: operater pritisne tipko, nek pomični del stroja doseže končno lego in ob tem sklene mejno stikalo, temperatura doseže določeno vrednost itd. Sistem vodenja mora zagotavljati, da si dogodki in stanja sledijo v predpisanem zaporedju, s čimer dobimo predpisano zaporedje delovnih operacij.

Sistem sekvenčnega vodenja prikazuje Sl. 7.50.



Sl. 7.50. Shema sekvenčnega vodenja

Proces vodimo s krmilnikom, ki zagotavlja pravilno zaporedje stanj in dogodkov. Podatke o stanju procesa oziroma o nastopu posameznih dogodkov dobimo s tipalom podobnimi dvovrednostnimi stikali, ki so priključena na vhode krmilnika. Poleg teh so na vhode krmilnika priključene tudi tipke in stikala, preko katerih lahko operater nastavlja različne načine delovanja. Izhodi krmilnika - krmilni signali pa prožijo aktuatorje v procesu in omogočajo nastop naslednjega dogodka.

Krmilnik je lahko realiziran na različne načine. V preteklosti so bila to najpogosteje kar relejska logična vezja, kasneje so uporabljali polprevodniška logična vezja, v sodobnih sistemih pa se najpogosteje uporabljajo programabilni logični krmilniki (PLK). (Več o PLK bomo podali v poglavju 9.)

S stališča krmilnika je diskretno stanje procesa množica vrednosti vhodnih in izhodnih spremenljivk. Število stanj procesa, ki jih krmilnik lahko zazna, je odvisno od števila vhodno/izhodnih spremenljivk. V primeru, da ima krmilnik 3 vhodne in 3 izhodne spremenljivke, dobimo z ozirom na dve različni vrednosti spremenljivk 2^6=64 možnih diskretnih stanj. Vendar pa običajno v procesu nastopa le nekaj stanj od vseh možnih. Poleg stanj procesa ima lahko krmilnik še dodatna notranja stanja. Število teh je odvisno od kompleksnosti vodenja.

Kot že omenjeno, je cilj sekvenčnega vodenja v večini procesov proizvodnja nekega izdelka. Postopek je sestavljen iz več operacij; nekatere operacije si sledijo zaporedno, nekatere tečejo vzporedno. Nekateri dogodki v procesu zahtevajo vzpostavitev določenega diskretnega stanja, drugi lahko zahtevajo zvezno regulacijo določene spremenljivke v času trajanja določene delovne operacije. V širšem smislu tako sekvenčno vodenje vključuje zaporedne ter vzporedne aktivnosti in diskretne akcije kot tudi zvezno regulacijo; je torej nekakšen nadzornik dogajanja.

7.3.2.1 Osnovni pristopi k sekvenčnemu vodenju

Pri sekvenčnem vodenju se srečujemo z dvema osnovnima vrstama logičnih krmilij:

- *kombinacijska logična krmilja* so krmilja, pri katerih se stanje vhodnih signalov neposredno preslika v določeno stanje izhodnih signalov v smislu Boolove preklopne logike. Takšno krmilje ne vsebuje notranjih pomnilnih elementov;
- *sekvenčna logična krmilja* so krmilja, kjer stanje izhodov ni odvisno le od stanja vhodnih signalov temveč tudi od stanja določenih notranjih signalov (stanja notranjih pomnilnih elementov). Zaradi notranjih pomnilnih elementov se lahko ista kombinacija vhodnih stanj preslika v različne izhodne kombinacije, odvisno od predhodnega dogajanja v krmilju oziroma od zaporedja predhodnih kombinacij na vhodu krmilja.

Vodenje prvih sistemov izvedemo s kombinacijsko logiko, druge pa vodimo s pomočjo sekvenčnih vezij. Prav pogosto pa en in drugi način uporabljamo kombinirano.

Kombinacijska krmilja načrtujemo z metodami, poznanimi iz digitalne tehnike. Zahteve

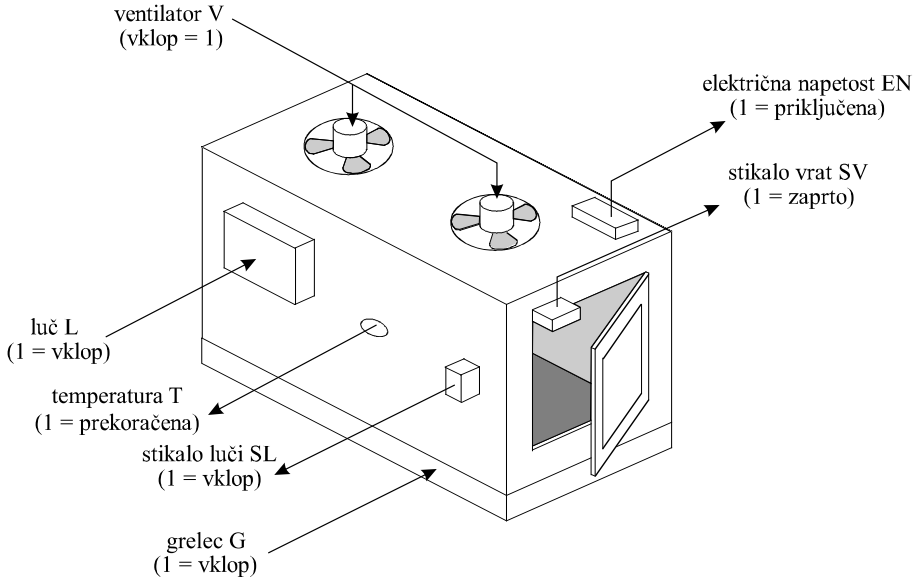
za krmiljenje pretvorimo v pravilnostno tabelo, iz katere potem lahko zapišemo Boolove enačbe za posamezne izhodne spremenljivke. V primeru realizacije krmilja z logičnim vezjem enačbe minimiziramo, v primeru realizacije krmiljenja s programabilnim logičnim krmilnikom pa to ni nujno potrebno. Če je krmilnik ustrezno izbran, velikost njegovega pomnilnika običajno ni problematična. Enačbe na koncu pretvorimo v obliko, primerno za realizacijo vezja ali vnos v programabilni logični krmilnik. To je lahko shema digitalnega vezja, relejska logična shema ali eden od grafičnih opisnih jezikov za programiranje krmilnikov, od katerih bosta dva predstavljena v nadaljevanju. Podrobneje pa bodo programski jeziki logičnih krmilnikov predstavljeni v podpoglavju o programskih orodjih za krmilnike.

Rešitve s sekvenčnim vodenjem zahtevajo manjše in enostavnejše programe kot enakovredne kombinacijske rešitve, imajo pa tudi večjo robustnost z ozirom na šumne signale in motnje. Razlog za to je, da je sekvenčno krmilje sistem, ki ga sestavljajo stabilna stanja. Ko enkrat sistem zavzame neko stanje, izstopi iz njega le, če nastopi izhodni pogoj. Med tem časom ne odgovarja na spremembe vhodnih pogojev. Ta lastnost stabilnih stanj omogoča tudi enostavnejše odkrivanje napak. Če nastopi napačni signal, ostane sistem v trenutnem stanju, zato lahko vzrok napake analiziramo.

Kombinacijska logika

Reševanje s kombinacijsko logiko si oglejmo na primeru laboratorijske peči. Sl. 7.51 shematično prikazuje laboratorijsko peč z vhodi (puščica v peč) in izhodi (puščica iz peči).

Vse spremenljivke so binarne, nakazana je tudi zveza med stanji in spremenljivkami.



Sl. 7.51. Laboratorijska peč

Predvidevamo naslednje možne dogodke, ki povezujejo vse tri izhode z vhodi.

- Grelec (G) se vklopi, ko aktiviramo močnostno stikalo (EN) in so vrata zaprta (SV) in je temperatura (T) pod mejo.
- Ventilator (V) se vklopi, ko je vključen grelec (G), ali ko je temperatura (T) nad mejo in so vrata (SV) zaprta.
- Luč (L) se prižge, ko vklopimo stikalo luči (SL), ali če so vrata odprta.

Rešitev problema je enostavni prenos navedenih trditev o dogodkih v Boolove enačbe:

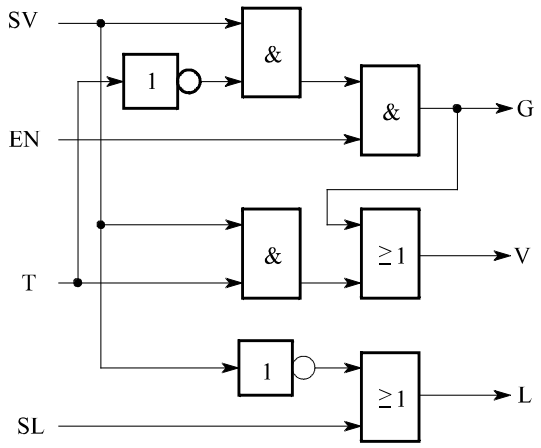
$$\text{Grelec:} \quad G = SV \cdot \overline{T} \cdot EN$$

$$\text{Ventilator:} \quad V = G + T \cdot SV$$

$$\text{Luč:} \quad L = SL + \overline{S}\,\overline{V}$$

Krmilje, s katerim lahko realiziramo gornje enačbe, je podano na Sl. 7.52. Sestavljeno je iz enostavnih logičnih elementov *IN*, *ALI* vrat in negatorjev.

Očitno je, da v tem primeru ne gre za zaporedje dogodkov, ampak samo za določanje izhodov na osnovi logične kombinacije določenih vhodov.



Sl. 7.52. Krmilje

Sekvenčna logika

Za enostavne procese zahteva razvoj programov sekvenčnega vodenja relativno malo načrtovanja, ker so programi kratki in lahko razumljivi v smislu predstavitve in delovanja. Seveda v praksi vezja, s katerimi rešimo probleme sekvenčnega vodenja, niso kot pri kombinacijski logiki omejena na *IN* ter *ALI* vrata, ampak vsebujejo razne logične in pomnilne funkcije skupaj z mnogimi programabilnimi funkcijami, ki jih dovoljujejo moderni PLK.

Ko pa načrtujemo širše in bolj kompleksno krmilje, postane hitro jasno, da neformalen in nesistematičen pristop k načrtovanju vodi do programov, ki so težki (tudi nemogoči)

za razumevanje, iskanje napak in dokumentiranje. Sam načrtovalec še razume delovanje, vendar že po krajšem času to znanje izgubi, če se ne ukvarja vseskozi s programom.

Zato se metodologija načrtovanja obširnejših programov sekvenčnega vodenja ne razlikuje od običajnega računalniškega programiranja, katerega primarne naloge so:

- definicija (specifikacija) naloge;
- načrtovanje programa;
- dokumentiranje;
- preizkušanje programa.

Pri specifikaciji izhajamo iz zadanih ciljev vodenja. Potem, ko smo določili cilje vodenja procesa oz. posameznih funkcijskih blokov in potrebno opremo, moramo s primernim opisom definirati potrebna zaporedja dogodkov.

Tak opis lahko podamo s stavki (besedami) ali z diagramom poteka. Posebno za tiste, ki ne programirajo pogosto, daje besedni opis možnost odkrivanja slabo definiranih sekundarnih ciljev ali pomanjkljive opreme. Vsak opis zaporedja dogodkov ima dva ločena dela:

- opis startne - inicializacijske faze;
- opis operativne - delovne faze;

o katerih bomo podrobneje govorili kasneje.

Besedni opis nato pogosto pretvorimo v nazornejši diagram poteka, katerega osnovni sestavni deli so vhodno/izhodni bloki, bloki akcij in bloki preizkušanja pogojev.

Značilno zgradbo diagramov poteka prikazuje Sl. 7.53.

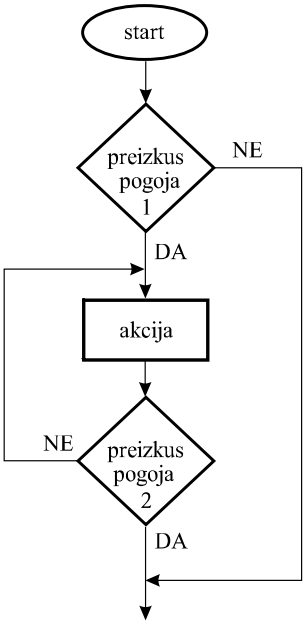
V naslednjem koraku pretvorimo diagram poteka v logične enačbe stanj ali sekvenčni program.

Običajno se dajo PLK programirati v enem ali dveh programskih jezikih, in ker programski jezik domuje v pomnilniku PLK, je potrebna izbira pred nakupom PLK.

Odvisno od potreb je programski jezik lahko enostavna Booleova mnemonika, lestvični diagram z dodatnimi funkcijskimi bloki, eden od takoimenovanih grafičnih funkcijskih jezikov ali visokonivojski jezik opisnih stavkov (podobnih n.pr. BASIC-u). Vsaka od teh možnosti ima svoje prednosti in pomanjkljivosti in večinoma je izbira programskega jezika kompromis med:

- jezikom, ki ga uporabnik najbolje pozna in
- jezikom, ki je najprimernejši za reševanje nekega problema vodenja.

Pri zmogljivejših PLK, ki istočasno rešujejo več različnih nalog (integralno vodenje), mora biti zato dana možnost programiranja v različnih jezikih, ki so najbolje prilagojeni specifičnemu značaju problemov.



Sl. 7.53. Značilni del diagrama poteka

Kot smo že videli, relejske sisteme, ki izvajajo enostavno kombinacijsko logično procesiranje, programiramo z logičnimi enačbami, pri sekvenčnih programih pa si bomo ogledali programiranje z lestvičnimi diagrami in s sekvenčnimi funkcijskimi diagrami.

7.3.2.2 Lestvični diagrami

Lestvični diagrami (Ladder Diagram - LD) so nastali iz potrebe po preglednejši predstavitvi vezja relejskih krmilnikov. Simbolično so predstavljali konkretne elemente, npr. releje z njihovimi kontakti, mejna stikala, motorje, ventile itd. razvrščene v prečkah med dve navpični črti (od tod ime lestvični diagram), ki sta predstavljali pozitivni in negativni pol napetosti. V vsaki prečki je bilo lahko več vhodnih naprav in le ena izhodna. Vhodne naprave so bile predstavljene s kontakti v normalno odprtem (*NO*) ali normalno zaprtem stanju (*NZ*), katerih stanje je ponazarjalo mirovno stanje procesa. Izhodna naprava je bila vključena, kadar je bila preko kontaktov vhodnih naprav sklenjena pot od pozitivnega k negativnemu polu napetosti. Z vklopom so se naenkrat pregledale vse prečke, vendar smo s časovnimi releji lahko zadržali izpolnitev neke prečke. Simboli elementov so bili prilagojeni dejanskim napravam.

Ko so razvili mikroprocesorske PLK, se je spremenil tudi lestvični diagram, ki sedaj predstavlja program PLK. Njegova zgradba je ostala sicer enaka, le zajemanje in izvrševanje posameznih prečk je zaradi zaporednosti delovanja digitalnih računalnikov sedaj sekvenčno. Vendar se program izvaja ciklično z dovolj kratkim časom enega cikla, tako da je delovanje krmilnika navzven podobno relejskemu vezju. Simboli lestvičnega

diagrama ne predstavljajo več fizikalnih relejev, motorjev, itd., ampak samo dele programske in aparaturne opreme krmilnika. Nekaj značilnih simbolov lestvičnega diagrama je zbranih v Tabeli 7.4.

Tabela 7.4. Simboli lestvičnih diagramov

Vrsta elementa	Ime	Značilnost	Simbol
Pogojni element	Vhodni kont. relejev	NO normalno odprt	
	ali	NZ normalno zaprt	
	Vhodni kont. mejnih stikal (tipk)	prehodno odprt	
		prehodno zaprt	
Povezave	Zveze	Vodoravna	
		Navpična	
Izvršni el.	Izhodni kont.	direktni	( )
		inverzni	(/)
	Zapah		(S)
			(R)
	Skok programa		(J)
	Klic podprogama		(C)
Funkcijski bloki: časovniki, števci, registri, primerjalniki Operativni bloki			

S pogojnimi elementi testiramo vrednosti vhodov krmilnika ali notranjih logičnih spremenljivk. Pogojni element ”prevaja”, če je prirejeni signal v ustreznem stanju. Tako npr. normalno odprt kontakt (prvi simbol v tabeli) prevaja, če je prirejeni signal v stanju logične 1 (npr. +24V na vhodni sponki krmilnika), medtem ko normalno zaprt kontakt prevaja, če je prirejeni signal v stanju logične 0. Prehodno odprt kontakt prevaja le v prvem programskem ciklu po spremembi stanja signala iz 0 na 1, prehodno zaprt kontakt pa v prvem programskem ciklu po spremembi stanja signala iz 1 na 0. S povezavami sestavljamo pogojne elemente v logične pogoje in jih povezujemo na izvršne elemente. Zaporedna vezava pogojnih elementov predstavlja logični IN, vzporedna vezava logični ALI.

Izvršni element se aktivira, ko prevaja vezje pogojnih elementov med njim in levo navpično črto, ki predstavlja pozitivni pol napetosti. Aktiviranje izvršnega elementa

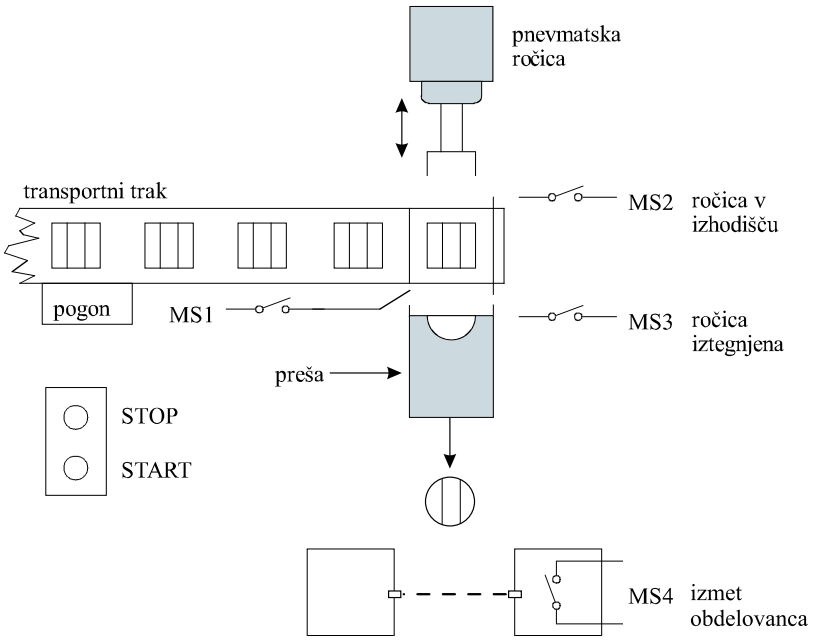
pomeni postavitev prirejenega signala v logično 1 (npr. +24V na izhodni sponki), če gre za direktni izhodni kontakt ali postavitev signala v stanje logične 0, če gre za inverzni kontakt. Izvršni element v obliki zapaha S (Set) se aktivira, ko vezje pogojev prevaja, vendar ostane aktiviran, tudi ko vezje pogojev neha prevajati. Za izključitev takšnega elementa moramo uporabiti zapah R (Reset), ki deaktivira izvršni element, če pripadajoče vezje pogojev prevaja. Poleg naštetih osnovnih elementov vsebujejo lestvični diagrami še števce, časovnike, skoke in klice podprogramov ter druge funkcije. Vendar se tako simboli teh elementov kot njihov nabor razlikujejo med različnimi vrstami PLK.

Vsi simboli lestvičnih diagramov imajo smisel le, če so prirejeni določeni vhodno/izhodni sponki krmilnika ali notranji pomnilniški lokaciji, s čimer dobijo nek fizikalni pomen glede na proces, ki ga s krmilnikom vodimo. Za prireditev programskih elementov k fizikalnim uporabimo prireditveno tabelo.

Najlaže ponazorimo lestvični diagram na konkretnem primeru.

∇

Primer 7.11: *Stroj za preoblikovanje*



Sl. 7.54. Shema stroja za preoblikovanje

Sl. 7.54 prikazuje stroj za preoblikovanje, ki ga sestavlja tekoči trak, ki služi za dostavo surovcev, pnevmatska ročica za podajanje in stiskalnica.

Tekoči trak dostavlja surovce enega za drugim k stiskalnici. Ko je surovec nasproti stiskalnice, mejno stikalo *MS1* ustavi trak. Nato pnevmatska ročica potisne surovec v stiskalnico. Po aktiviranju mejnega stikala *MS3* se vrne ročica v izhodiščni položaj, ko se aktivira mejno stikalo *MS2*, se izvede stiskanje v želeno obliko. Sledi izmet obdelovanca, ki vključi svetlobno stikalo *MS4* in omogoči obdelavo naslednjega surovca.

Za prireditev elementov lestvičnega programa k fizikalnim uporabimo prireditveno tabelo (Tabela 7.5).

Poleg fizičnih vhodov X in izhodov Y smo vpeljali tudi notranje logične spremenljivke ali markerje M, s katerimi bomo realizirali prehajanje stanj sekvenčnega krmilja. Lestvični diagram pripadajočega krmilja prikazuje Sl. 7.55.

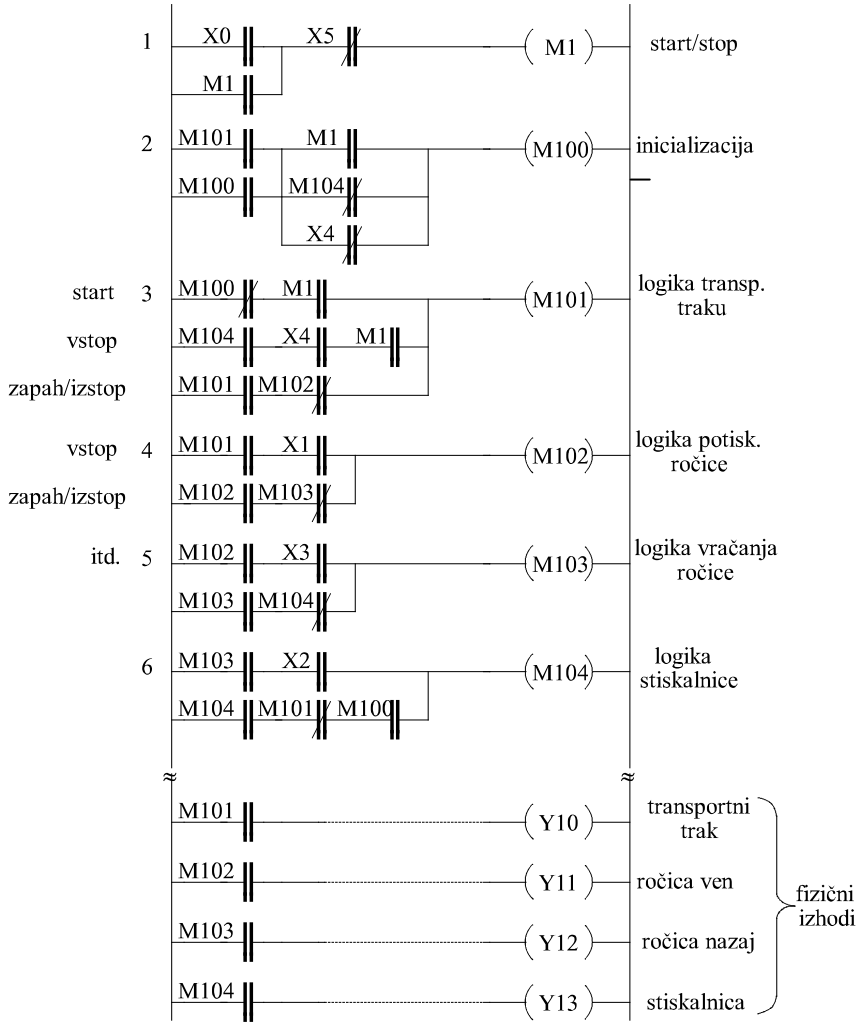
Tabela 7.5. Prireditvena tabela

Signal	PLK naslov
START	X0
MS1	X1
MS2	X2
MS3	X3
MS4	X4
STOP	X5
Osnovno stanje	$\overline{\text{M100}}$
Stanje 1	M101
Stanje 2	M102
Stanje 3	M103
Stanje 4	M104
Transportni trak	Y10
Ročica potiska	Y11
Ročica se vrača	Y12
Stiskalnica	Y13
Start	M1

Večina lestvičnih programov vsebuje nekaj značilnih elementov, ki jih bomo opisali kar na obravnavanem primeru.

a) Postavitev v začetno stanje - inicializacija (prečka 2) s pogoji vklopa in vračanja v začetno stanje reset (prečka 1)

Po vsej verjetnosti ima krmiljena oprema osnovni ali normalni položaj npr. vsi aktuatorji izključeni in vsa mejna stikala odprta, rezervar poln itd., za kar poskrbi faza inicializacije.



Sl. 7.55. Lestvični diagram stroja za preoblikovanje

Večina industrijskih procesov ima ročno start in stop krmiljenje preko tipk, ki omogočajo izvedbo operacij in vračanja. V tem primeru je to zajeto v 1. prečki, ki vsebuje tudi osnovno filozofijo sekvenčnega krmiljenja:

Vstop v stanje, ki ga zadrži ali izstopi.

V obratovalno stanje, ki ga ponazarja M1, vstopimo s pritiskom na tipko *START* (X0), če ob tem ni pritisnjena tudi tipka *STOP* (X5) - v pogoju smo uporabili mirovno zaprt kontakt. Kontakt M1 nato vzdržuje to stanje, tudi ko spustimo tipko *START*, in sicer vse dotlej, dokler ne pritisnemo tipke *STOP* in s tem izstopimo iz stanja. Logična enačba 1. prečke je

$$M1 = X0 \cdot \overline{X5} + M1 \cdot \overline{X5}$$

STANJE VSTOP ZAPAH / IZSTOP

Vsako novo stanje opišemo s podobno zgradbo.

Prečki 1 in 2 omogočata start sekvence po pritisku tipke *START* in zaustavitev po pritisku tipke *STOP*. Pomen prečke 2 je v tem, da definira, kdaj je krmilje v začetnem stanju. To je takrat, ko je M100 = 0 in transportni trak še ni vključen (M101 = 0) (torej preden stiskalnica sploh prične obratovati), ali zatem, ko smo pritisnili tipko *STOP* (M1 = 0) in se je cikel izvršil do konca (M104 = 1 in X4 = 1). Marker M100 ima pri tem invertiran pomen, saj je krmilje v začetnem stanju, če je M100 = 0. Zato dobimo logično enačbo prečke 2 tako, da zgornjo trditev zapišemo v obliki Boolove enačbe in jo negiramo. Lahko bi uporabili tudi izvršni element z inverznim izhodnim kontaktom, v tem primeru logične enačbe ne bi bilo potrebno negirati.

S prečko 2 zagotovimo, da lahko tipka *START* preko markerja M1 sproži sekvenco le, če je krmilje v začetnem stanju, torej ko je M100 = 0 (vstopni pogoj prečke 3), medtem ko po pričetku cikla krmilje na pritisk tipke *START* ne reagira. Tipka *STOP* sekvence ne zaustavi takoj, ampak se tekoči cikel izvrši do konca (izstopni pogoj prečke 6 vključuje tudi M100, ki postane enak 0 ob koncu stiskanja, če M1 ni več prižgan).

b) procesna operativna/sekvenčna logika predstavlja glavni del programa, ki vsebuje programe kombinacijskih zaporednih vezij. Rezultirajoči izhodi običajno ne krmilijo izvršnih členov direktno, ampak posredno preko takoimenovanih marker relejev. V našem primeru predstavljajo operativno fazo prečke 3 do 6, ki se ciklično ponavljajo.

Prečka 3 se aktivira, če je krmilje v začetnem stanju (M100 = 0) in vključimo obratovalno stanje (M1 = 1), ali če je krmilje v obratovalnem stanju in se cikel izvrši do konca (M104 = 1 in X4 = 1). Ko je prečka 3 aktivirana, je M101 enak 1, kar požene motor transpornega traku, dokler ni položaj surovca pravilen (*MS1* oz. X1). Tedaj se aktivira prečka 4 (M101 = 1 in X1 = 1), M102 postane enak 1, takoj v naslednjem programskem ciklu pa se deaktivira prečka 3 zaradi pogoja M102 = 0 v spodnji veji, ki vzdržuje stanje prečke po vstopu.

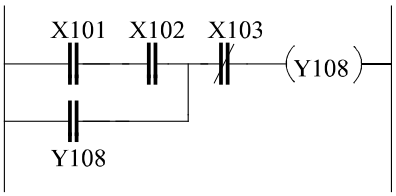
Podobno je realizirano tudi aktiviranje in deaktiviranje ostalih prečk. Prečka 4 pomakne pnevmatsko ročico in s tem potisne surovec v stiskalnico. Mejno stikalo

MS3 (X3) detektira, kdaj je ročica iztegnjena in vključi vračanje ročice (prečka 5). Ko se sklene *MS2* (X2), je ročica v izhodiščnem položaju in prične se stiskanje (prečka 6). Po stiskanju sledi samodejen izmet obdelovanca, ki ga zazna stikalo *MS4* (X4) in preko prečke 3 omogoči ponovni zagon transportnega traku. Če je bila med tekom cikla pritisnjena tipka *STOP*, se po izmetu obdelovanca ne aktivira prečka 3, temveč se krmilje postavi v izhodiščni položaj (prečka 2).

c) Program običajno dopolnimo s prireditveno tabelo za izhodne signale, ki povezuje stanje markerjev s pogoji omogočanja iz a) (zadnje štiri prečke na (Sl. 7.55)).

d) Vsak realni program ima še dodatno logiko za prikazovanje stanj npr. z indikatorskimi lučmi in logiko zaščite ter alarmiranja, ki skrbi, da se oprema in osebje ne poškodujejo.

Oglejmo si še, kako program v lestvičnem diagramu naložimo v pomnilnik (RAM) PLK na primeru zapaha (Sl. 7.56).



0	LD	X101	naloži (NO)
1	AND	X102	veži zaporedno (NO)
2	LD	Y108	naloži (NO)
3	ORB		veži vzporedno
4	ANI	X103	naloži (NZ)
5	OUT	Y108	izhod

Sl. 7.56. Pretvorba lestvičnega diagrama

Δ

7.3.2.3 Sekvenčni funkcijski diagrami

Če postanejo lestvični diagrami preobširni, in če različne naloge sekvenčnega vodenja programiramo z različnimi jeziki, postanejo programi nepregledni in težko razumljivi. Za take primere uvedemo sekvenčne funkcijske diagrame (Sequential Function Chart - SFC). Tabela 7.6 prikazuje simbole sekvenčnih funkcijskih diagramov. Sekvenčne funkcijske diagrame sestavljajo koraki oziroma makrokoraki in prehodi med njimi (Sl. 7.57). Makrokorak je kondenzirana predstavitev skupine korakov in prehodov, ki jo uvaja vhodni korak in zaključuje izhodni korak. Makrokoraki so lahko združeni v sekcije ali so neodvisni.

Prednosti take zgradbe programov:

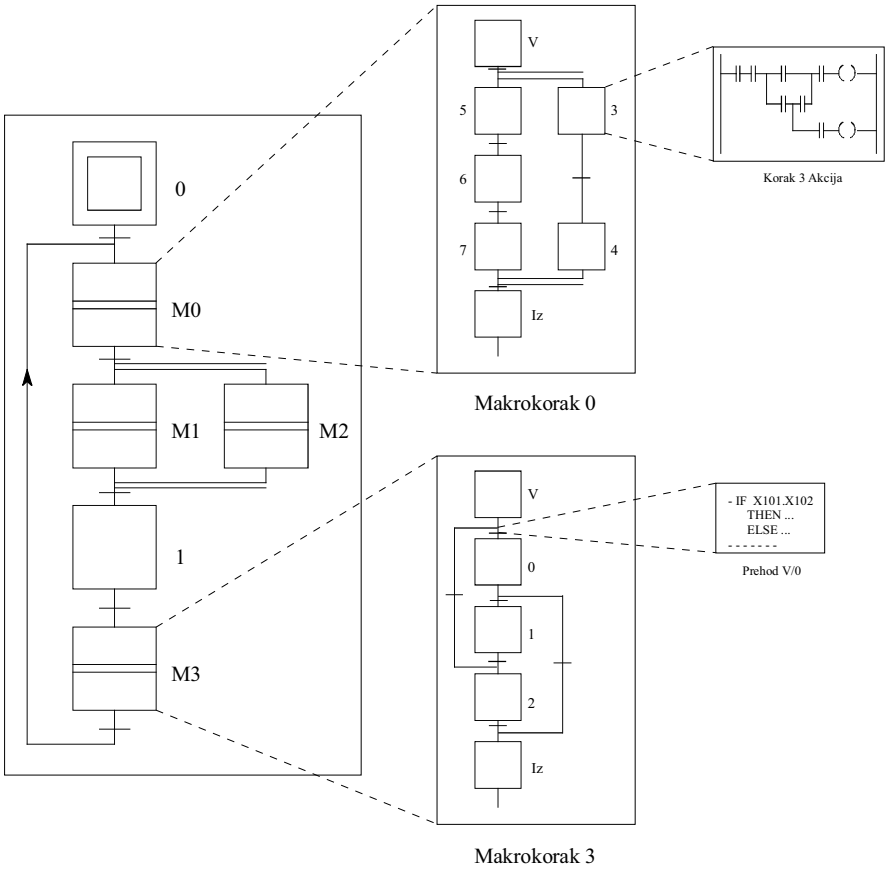
- enostavnejši razvoj programa, katerega dele lahko neodvisno programiramo v različnih jezikih;
- hitrejši čas pregledovanja, zaradi vzporednega izvajanja (časovna delitev) različnih nalog;
- lažje iskanje napak;
- lažje spremembe starih in dodajanje novih programov.

Tabela 7.6. Simboli sekvenčnih funkcijskih diagramov

Koraki	n - ti	n
	vstopni korak makrokoraka	V
	izstopni korak makrokoraka	Iz
Makrokoraki	začetni	
	Monoaktivni makrokorak za programiranje izbranih sekvenc	M_n
	Multiaktivni makrokorak za programiranje sočasnih sekvenc	M_j
Prehodi	od enega koraka k drugemu	
	od enega koraka do večih	
	od večih korakov do enega	
	od večih korakov do večih	
Usmerjene zveze	navzgor	↑
	navzdol	↓
	desno	→
	levo	←

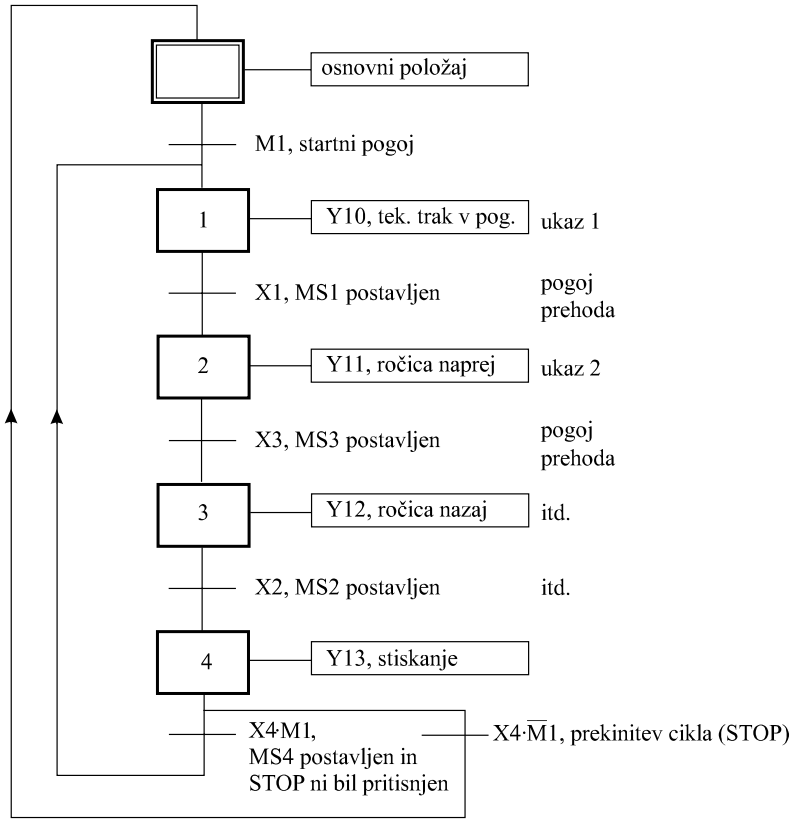
Zaradi naštetih prednosti so sekvenčni funkcijski diagrami najprimernejša osnova

načrtovanja sekvenčnega vodenja. V tej vlogi so vključeni tudi v mednarodni standard IEC 1131.3, ki definira standardne programske jezike logičnih krmilnikov in je podrobneje predstavljen v prispevku o programskih orodjih za krmilnike.



Sl. 7.57. Zgradba sekvenčnih funkcijskih diagramov

Sl. 7.58 prikazuje sekvenčni funkcijski diagram za primer stiskalnice. Prikazana je le glavna sekvenca, medtem ko je potrebno branje tipke *STOP*, ki je neodvisno od glavne sekvence, realizirati posebej.



Sl. 7.58. Sekvenčni funkcijski diagram stroja za preoblikovanje

7.3.3 Regulacije

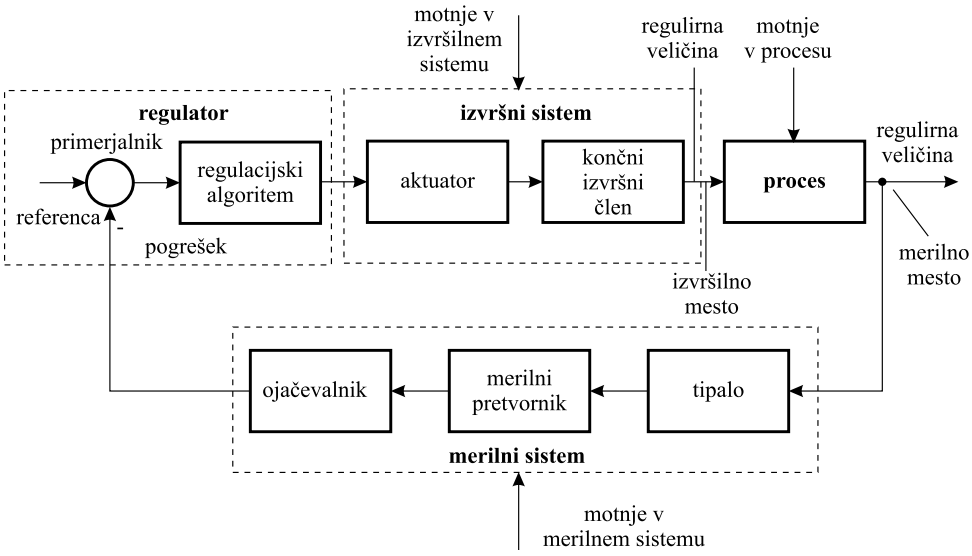
Regulacija je povratnozančno vodenje procesa, pri katerem vplivamo na proces tako, da se veličine, ki jih reguliramo, čim bolj ujemajo z želenimi (referenčnimi) veličinami ne glede na motnje, katerim je regulacijski sistem podvržen.

7.3.3.1 Osnovni pojmi v regulacijskih sistemih in zahteve pri njihovem načrtovanju

Sl. 7.59 prikazuje bločni diagram regulacijskega sistema. Le-ta nazorno prikazuje vse gradnike, ki nastopajo v regulacijski zanki.

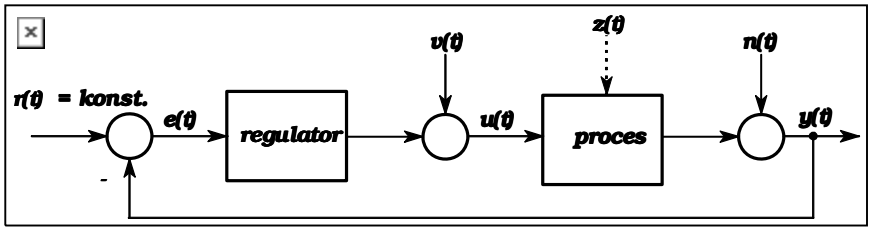
Tipalo odtipava regulirano veličino na merilnem mestu. Signal, ki ga daje tipalo, moramo pogosto pretvoriti z merilnim pretvornikom in ga nato ojačati z ojačevalnikom (npr. v standardni signal 4-20 mA). Na poti od merilnega mesta se vnašajo motnje, ki jih imenujemo motnje v merilnem sistemu (ali tudi motnje na regulirano veličino). Pretvorjena regulirana veličina vstopa v regulator in se tam primerja z želeno oz.

referenčno vrednostjo. Iz pogreška med njima regulacijski algoritem (najbolj inteligentni del v regulacijski zanki) določi signal (npr. 4-20 mA), ki preko aktuatorja (npr. motor) in končnega izvršnega člena (npr. ventil, loputa) na izvršilnem mestu deluje na proces. Na poti od regulatorja do procesa lahko delujejo motnje, ki jih imenujemo motnje v izvršnem sistemu (ali tudi motnje na regulirno veličino). Tipalo, merilni pretvornik in ojačevalnik združimo v en blok, ki ga imenujemo merilni sistem. Aktuator in končni izvršni člen pa z enim izrazom imenujemo izvršni sistem.



Sl. 7.59. Bločna shema regulacijskega sistema

Kadar pa proučujemo dinamične lastnosti regulacijske zanke, pa Sl. 7.59 preuredimo tako, da izvršni in merilni sistem vključimo v regulator oz. proces. S tem ima bločna shema le dva bloka: regulator in proces (Sl. 7.60).

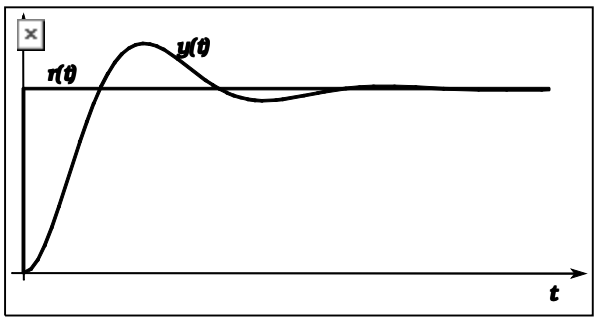


Sl. 7.60. Bločna shema regulacijskega sistema

r(t) je referenca, *e(t)* je pogrešek, *u(t)* je regulirna, *y(t)* pa regulirana veličina. *v(t)* je motnja na vhodu procesa, *n(t)* na izhodu, motnja *z(t)* pa deluje nekje v procesu.

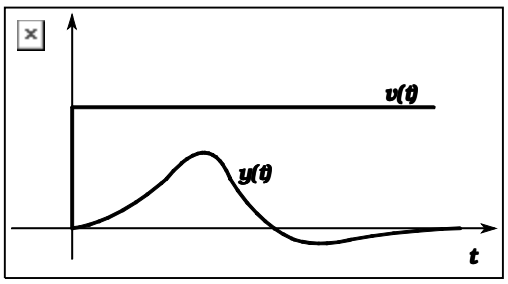
Sledilno in regulacijsko delovanje

Pri sledilnem delovanju mora biti regulacijski sistem načrtan tako, da regulirana veličina čim bolj verno sledi referenčni veličini. Tipični primeri sledilne regulacije so: zagotovitev hitrostnega profila pri delovanju dvigala, zagotovitev določenega temperaturnega profila pri številnih procesih v kemični industriji, sledenje strelne naprave tarči, regulacije zasuka vesoljskih ladij itd. Pri načrtovanju sledilne regulacije je manj pomembno, kako regulacijski sistem deluje pri morebitnih motilnih signalih. Sl. 7.61 prikazuje tipičen odziv regulacijskega sistema na stopničasto spremembo reference.



Sl. 7.61. Tipična poteka referenčne in regulirane veličine pri sledilni regulaciji

Kljub temu, da je načrtovanje sledilnega regulatorja običajno bolj podrobno obravnavano v literaturi, pa je zlasti v procesni industriji bolj pomembno, da regulator učinkovito odpravlja motnje, oz. da le-te čim manj vplivajo na regulirano veličino. Takemu načinu delovanja regulacijskega sistema pravimo regulacijsko delovanje. Učinkovitost reagiranja na spremenjen referenčni signal nas pri tej regulaciji manj zanima, ker predpostavimo, da je želena vrednost nastavljena na neko konstantno vrednost in se ne spreminja. Tipični primeri so: regulacija temperature v prostoru, ko je želena temperatura konstantna, regulator pa mora čimbolj izločiti vplive motenj (sprememba zunanje temperature, odpiranje oken, vrat, itd.). Sl. 7.62 prikazuje tipičen odziv na stopničasto motnjo *v(t)* na regulirni veličini.

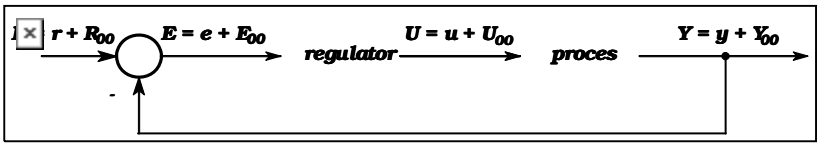


Sl. 7.62. Odziv na motnjo na regulirni veličini v regulacijskem sistemu

Za načrtovanje regulacije je bistveno, da vemo, kateri tip regulacije je pomemben oz. pomembnejši pri konkretnem problemu. Optimalni sledilni regulator ne odpravlja optimalno motenj in obratno. Pri načrtovanju optimalnega regulatorja za odpravljanje motenj je pomembno, da predvidimo tudi oblike motenj in mesto delovanja v regulacijskem sistemu.

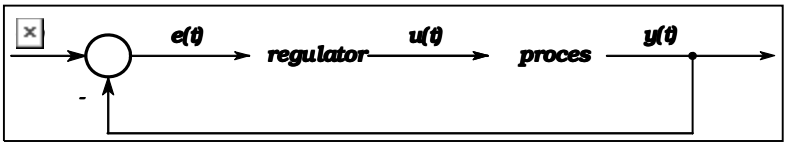
Obravnava regulacijskega sistema v delovni točki

Običajno si lahko vse signale v regulacijskih sistemih predstavljamo tako, da so sestavljeni iz nekih delovnih, enosmernih vrednosti in iz manjših sprememb v okolici delovne točke. Ustrezne oznake prikazuje Sl. 7.63.



Sl. 7.63. Obravnava regulacijskega sistema v okolici delovne točke

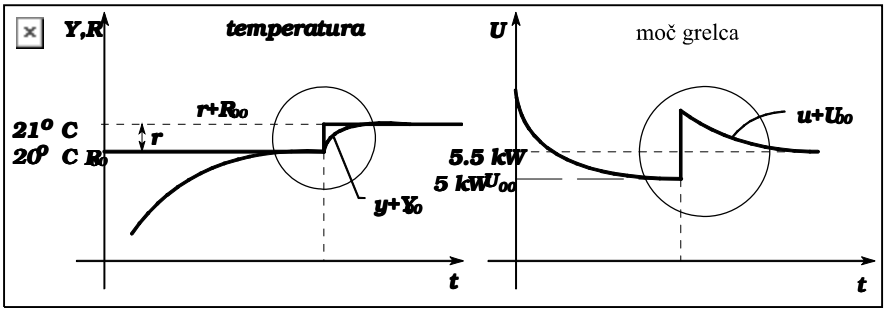
R_{00}, E_{00}, U_{00} in Y_{00} so enosmerne vrednosti signalov oz. vrednosti signalov, ki v regulacijskem sistemu določajo delovno točko. Če se v bližini te delovne točke regulacijski sistem vede približno linearno, lahko zaradi zakona superpozicije od vseh nastopajočih signalov odštejemo vrednosti delovnih točk (R_{00}, E_{00}, Y_{00}, U_{00}) in dobimo regulacijski sistem, ki ga uporabljamo pri dinamični obravnavi (Sl. 7.64). Takemu modelu regulacijskega sistema pravimo tudi *deviacijski model*.



Sl. 7.64. Deviacijski model regulacijskega sistema

Pri nadaljnji obravnavi bomo imeli ob takih oznakah, kot jih prikazuje Sl. 7.64, v mislih le spremembe signalov v okolici delovne točke in torej vrednosti delovne točke ne bomo jemali v obzir.

Za boljše razumevanje si poglejmo naslednji zgled. V prostoru imamo nastavljeno želeno temperaturo 20 ^{0}C. Ta temperatura se vzpostavi po določenem prehodnem pojavu in zahteva moč grela 5 kW. Če spremenimo želeno temperaturo za 1 ^{0}C, se po prehodnem pojavu moč grela poveča za 500 W. Prehodna pojava prikazuje Sl. 7.65.

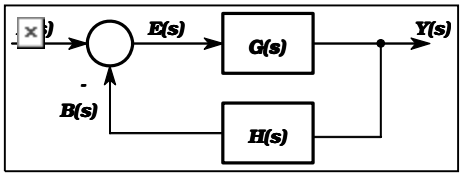


Sl. 7.65. Prehodna pojava pri regulaciji temperature v prostoru

Pri nadaljnji obravnavi regulacijskih sistemov bomo imeli v mislih vselej le prehodni pojav, ki je na Sl. 7.65 obkrožen. Posebno pogosto bomo analizirali sisteme pri enotini skočni spremembi referenčnega signala (enotina stopnica). To ne pomeni, da se je referenčni signal dejansko spremenil iz 0 na 1 problemske enote, ampak v delovni točki za neko konstantno vrednost, ki pa smo jo normirali na vrednost ena zaradi preglednejše nadaljnje obravnave, kar ne predstavlja nikakršne izgube na splošnosti.

<u>Posplošeni bločni diagram regulacijskega sistema</u>

V praksi je potrebno običajno regulirano veličino pretvoriti v neko drugo veličino, preden jo primerjamo z referenčno veličino (npr. sprememba temperature v napetost, filtriranje,…). Pretvorba lahko izkazuje tudi večje ali manjše zakasnitve, zato jo opišemo s prenosno funkcijo *H(s)*. Ni nujno, da *H(s)* vedno predstavlja prenosno funkcijo tipala in merilnega pretvornika. Lahko predstavlja del regulacijskega algoritma. Prenosno funkcijo regulatorja in procesa pa združimo v prenosno funkcijo *G(s)*. Ustrezno povratnozančno strukturo prikazuje Sl. 7.66.



Sl. 7.66. Posplošeni bločni diagram regulacijskega sistema

Glede na Sl. 7.66 definiramo:

Prenosna funkcija direktne veje je kvocient med regulirano veličino *Y(s)* in pogreškom *E(s):*

$$\text{Prenosna funkcija direktne veje} = \frac{Y(s)}{E(s)} = G(s) \qquad (7.105)$$

Odprtozančna prenosna funkcija je kvocient med povratnozančnim signalom *B(s)* in signalom pogreška *E(s)*:

$$\text{Odprtozančna prenosna funkcija} = \frac{B(s)}{E(s)} = G(s)H(s) \qquad (7.106)$$

Zaprtozančna prenosna funkcija je kvocient med regulirano veličino *Y(s)* in referenčno veličino *R(s)*.

$$\text{Zaprtozančna prenosna funkcija} = \frac{Y(s)}{R(s)} = \frac{G(s)}{1+G(s)H(s)} \qquad (7.107)$$

<u>Zahteve pri načrtovanju dinamičnih lastnosti regulacijskih sistemov</u>

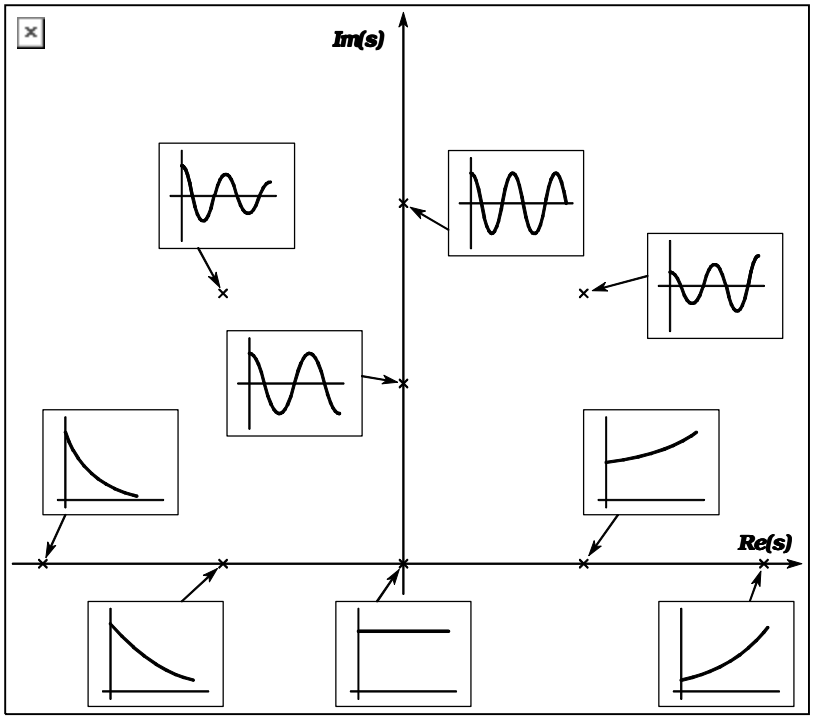
Pri načrtovanju regulatorja se srečamo s številnimi zahtevami, ki so si nemalokrat med seboj nasprotujoče. Zato so izkušnje načrtovalca izjemno pomembne. Na tem mestu bomo podali le nekatere najpomembnejše zahteve pri načrtovanju: stabilnost, razmere v ustaljenem stanju, robustnost in inženirske pokazatelje kvalitete regulacije.

Stabilnost

Regulacijski sistem je stabilen, če je regulirana veličina omejena pri kakršnem koli omejenem vhodnem signalu. Imenovalec zaprtozančne prenosne funkcije *1+G(s)H(s)* bistveno vpliva na dinamično vedenje regulacijskega sistema. Enačbo *1+G(s)H(s)=0* imenujemo *karakteristična enačba*. Njeni koreni so *poli* regulacijskega sistema. Pri analizi in načrtovanju dinamičnih lastnosti regulacijskih sistemov je lega polov najpomembnejša.

Odziv na kratkotrajni impulz (teoretično t.i. δ impulz) imenujemo naravni odziv sistema. Le-ta je sestavljen iz prispevkov, ki jih oblikujejo poli sistema. Oblike teh prispevkov v odvisnosti od lege polov v *s* ravnini prikazuje Sl. 7.67. Poli z negativnim realnim delom predstavljajo odzive, ki s časom izzvenijo, zato je v takem primeru regulacijski sistem *stabilen*. Poli v desnem delu ravnine *s* pa pomenijo nestabilno delovanje in so v regulacijskih sistemih nedopustni. Poli na realni osi pomenijo eksponencialne odzive, kompleksni poli pa uvajajo nihanja.

Stabilnost sistema je lastnost samega sistema in ni odvisna od oblike in velikosti vhodnega signala. Zagotovitev stabilnosti je prva in najpomembnejša naloga vsakega načrtovalnega postopka.



Sl. 7.67. Vpliv polov na naravni odziv sistema

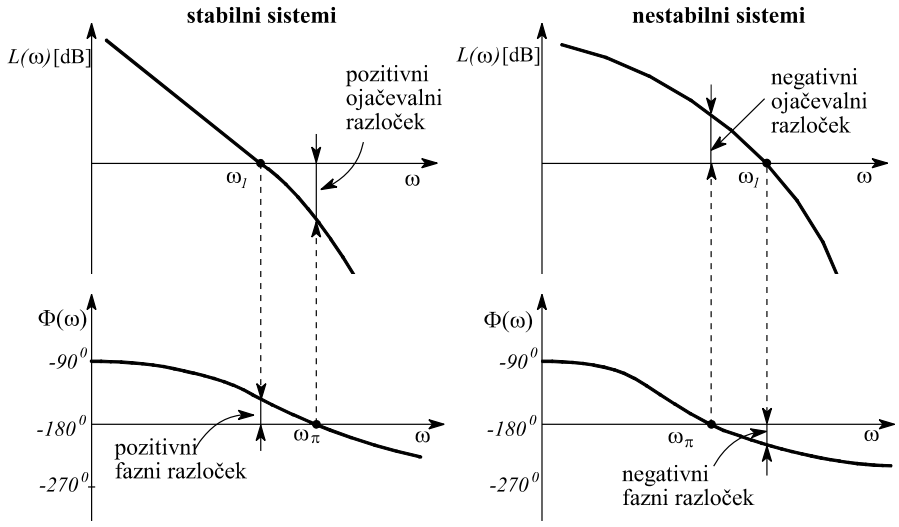
Z izbiro polov v levi polravnini pa še nismo zagotovili zadovoljivega prehodnega pojava. Za primeren prehodni pojav je potrebna ustrezna lega polov v levem delu ravnine *s*, vplivajo pa tudi ničle zaprtozančne prenosne funkcije.

Stabilnost regulacijskih sistemov pa lahko učinkovito proučujemo tudi s pomočjo frekvenčne karakteristike odprtozančne prenosne funkcije *G(s)H(s)* (npr. Bodejev diagram, enačbi (7.10) in (7.17) v 7.1.1). Regulacijski sistem je stabilen, če pri frekvenci ω_1, kjer ojačevalna karakteristika $L(\omega)$ seka 0dB, faza še ni -180° $(\phi(\omega) > -180^o)$. Z drugimi besedami mora biti ojačenje $L(\omega)$ negativno pri frekvenci ω_π, ko fazna karakteristika seka -180°.

Oddaljenost sistema od meje stabilnosti (stabilnostna rezerva) določata ojačevalni (k_m) in fazni razloček (ϕ_m)

$$\begin{aligned} k_m &= -20\,log\left|GH\left(j\omega_\pi\right)\right| \\ \phi_m &= 180^o + \phi\left(\omega_1\right) \end{aligned} \tag{7.108}$$

Stabilni sistem ima oba razločka pozitivna, nestabilni pa negativna. Proučevanje stabilnosti nazorno prikazuje Sl. 7.68.



Sl. 7.68. Proučevanje stabilnosti s pomočjo frekvenčne karakteristike

Z regulatorjem želimo običajno doseči ojačevalni razloček okoli 6dB, fazni pa 20-50°.

Ustaljeno stanje regulacijskih sistemov

Regulacijskega sistema ni potrebno analizirati le v času prehodnega pojava, ampak tudi v ustaljenem stanju. Ustaljeno stanje nastopi, ko izzveni prehodni pojav. Pri konstantnem referenčnem ali motilnem signalu je tudi ustaljeno stanje konstantno, če je le sistem stabilen. Običajno želimo, da je po določeni spremembi referenčnega signala ali po nastopu motnje pogrešek v ustaljenem stanju enak nič ali vsaj čim manjši.

Na pogrešek v ustaljenem stanju ključno vpliva t.i. vrsta sistema *G(s)H(s)*, ki je določena s številom polov prenosne funkcije *G(s)H(s)* v koordinatnem izhodišču ravnine *s*. Vsak tak pol predstavlja t.i. integrirni karakter. Enostavno se da izpeljati, da pri stopničasti spremembi reference regulacijski sistem nima ustaljenega pogreška v primeru, če ima prenosna funkcija *G(s)H(s)* vrsto 1 oziroma en pol v koordinatnem izhodišču ali enojni integrirni karakter. Ta integrirni karakter mora imeti regulator ali proces. Zato je osnovna naloga integrirnega člena v regulatorju, da odpravlja ustaljeni pogrešek.

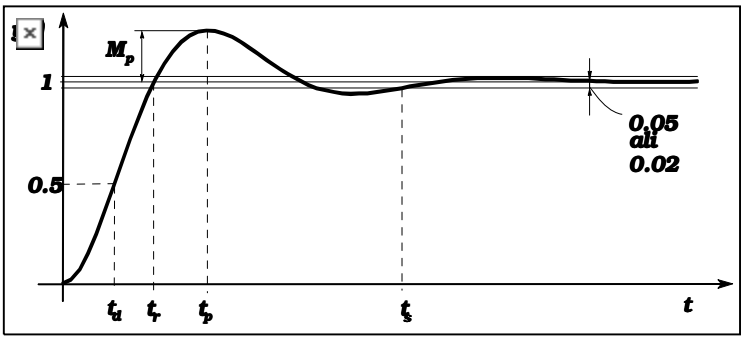
Robustnost

Parametri procesa se med obratovanjem pogosto spreminjajo. Zaradi povratne zanke se pri dobro nastavljenem regulatorju vpliv teh sprememb na regulirano veličino zmanjša. Pravimo, da je regulator robusten na spremembe parametrov.

Inženirski pokazatelji kvalitete regulacije

Lega polov sicer ključno vpliva na časovne poteke, vendar ustrezna postavitev polov

zahteva precej izkušenj in teoretičnega znanja. Zato so običajno zahteve bolj kot z lego polov podane s t.i. inženirskimi pokazatelji kvalitete. Le-te vpeljemo s pomočjo poteka regulirane veličine pri stopničasti spremembi reference (Sl. 7.69).



Sl. 7.69. Odziv regulacijskega sistema na stopničasto vzbujanje

Inženirski pokazatelji kvalitete so:

- t_d - *čas zakasnitve*. Čas zakasnitve je čas, v katerem regulirana veličina prvič doseže 50% končne vrednosti;
- t_r - *čas vzpona*. Čas vzpona je čas, v katerem odziv sistema naraste iz 10% na 90%, iz 5% na 95% ali iz 0% na 100% končne vrednosti. Za podkritično dušene sisteme se običajno uporablja kriterij 0 - 100%, za nadkritično dušene pa kriterij 10% - 90%;
- t_p - *čas maksimalnega prevzpona*. Čas maksimalnega prevzpona je čas, v katerem odziv doseže maksimalni prevzpon;
- M_p - *maksimalni (procentni) prevzpon*. Maksimalni prevzpon je za odziv na Sl. 7.69 določen z razliko med maksimalno vrednostjo odziva in vrednostjo 1. Če ustaljeno stanje ni določeno z vrednostjo ena, potem je prevzpon podan z enačbo

$$\text{Maks. proc. prevzpon} = M_p = \frac{y(t_p) - y(\infty)}{y(\infty)} \; 100\%$$

- t_s - *umiritveni čas*. Umiritveni čas je čas, ki ga potrebuje odziv, da doseže in ostane znotraj tolerančnega področja okoli ustaljene vrednosti (običajno ± *2%* ali ± *5%*). Umiritveni čas je odvisen od največje časovne konstante regulacijskega sistema.

Običajno želimo doseči pokazatelje, ki so manjši od prepisanih: čas vzpona $\leq t_{r0}$, maksimalni prevzpon $\leq M_{p0}$ in umiritveni čas $\leq t_{s0}$.

∇

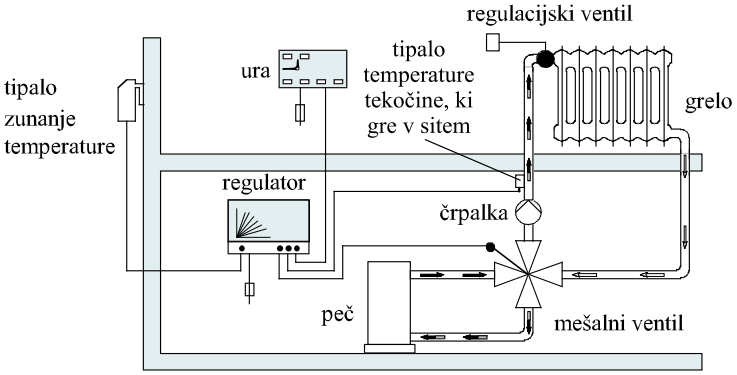
Primer 7.12: *Regulacija temperature v prostoru*

Regulacija temperature v prostoru spada med klasične, vsakomur znane, regulacijske

probleme. Zanjo obstajajo številni načini, ki v glavnem zavisijo od vrste grela, ki ga uporabljamo. V glavnem uporabljamo t.i. stopenjsko (ON-OFF) regulacijo v povezavi z električnimi greli, pa tudi bolj napredno regulacijo pri sistemih, ki imajo določeno akumulacijo toplotne energije.

Sl. 7.70 prikazuje primer zvezne regulacije temperature prostora s centralnim ogrevanjem.

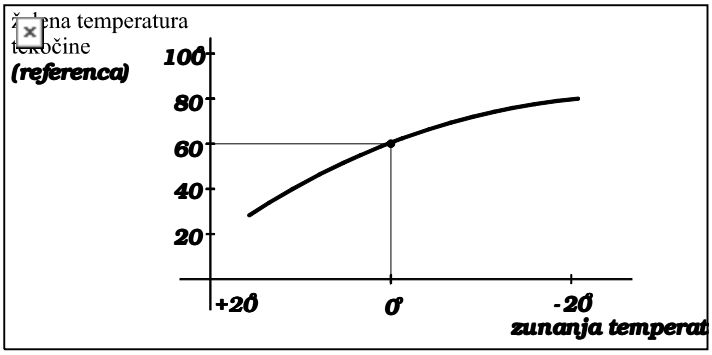
Regulacijski del sestavljajo tipalo zunanje temperature, tipalo, ki meri temperaturo tekočine, ki jo pošiljamo v sistem, regulator, ura, ki se nahaja v prostoru, ter mešalni ventil (končni izvršni člen), ki ga odpira in zapira motor (aktuator). Proces sestavljajo peč za centralno ogrevanje, črpalka, grelo v prostoru, ustrezen povezovalni sistem in seveda samo ogrevanje prostora. Osnovna zanka omogoča regulacijo temperature tekočine, ki jo pošiljamo v sistem. Regulator glede na razliko med želeno in dejansko temperaturo tekočine, ki jo meri tipalo, daje motorju ustrezno komando za zasuk ventila.



Sl. 7.70. Regulacija temperature v prostoru

Ventil je mešalnega tipa in omogoča želeno temperaturo tekočine na ta način, da v pravilnem razmerju meša tekočino, ki prihaja iz peči (ta je na konstantni, precej visoki temperaturi in regulirana z ustrezno regulacijsko zanko v kotlu) in povratno tekočino, ki se v sistemu precej ohladi. Značilnost regulacijskega sistema pa je v tem, da referenca za temperaturo tekočine ni fiksno nastavljena na regulatorju, ampak se preračuna preko t.i. družine statičnih karakteristik iz zunanje temperature. Tako pri nižjih zunanjih temperaturah polnimo sistem z bolj vročo tekočino. Tak koncept omogoča velik prihranek, saj je največji vir izgub pri centralnem ogrevanju prav v tem, da polnimo sistem s tekočino, ki je toplejša od potrebne. Eno od možnih statičnih karakteristik prikazuje (Sl. 7.71).

Uporabnik neposredno torej ne more nastaviti želene temperature, ker je le-ta odvisna od zunanje temperature, lahko pa vpliva na obliko statične karakteristike tako, da ji spreminja naklon ali jo vzporedno premika in na ta način naredi optimalno prilagoditev za svoj prostor.



Sl. 7.71. Statična karakteristika za izračun želene temperature

Ura običajno le vklaplja in izklaplja ogrevalni sistem (mešalni ventil) v želenih časovnih intervalih (npr. preklop med dnevnim in nočnim režimom), medtem ko želeno temperaturo v prostoru dosežemo z dodatnim regulacijskim ventilom na grelu.

Δ

Regulacije so v preteklih desetletjih prav gotovo predstavljale najmočnejše raziskovalno področje vodenja sistemov. Teoretične izsledke so predvsem vspodbujale potrebe vojaške in letalske industrije in šele mnogo kasneje tudi procesne industrije. Nastalo je ogromno število regulacijskih algoritmov, ki so bili dostikrat tudi prezahtevni za stopnjo tehnološkega razvoja. Podali bomo le tiste regulacijske metode, ki se danes največ uporabjajo predvsem v procesni industriji.

Na prvem mestu moramo vsekakor omeniti *proporcionalno-integrirno-diferencirno* (PID) regulacijo. Po nekaterih podatkih ta vrsta regulacije rešuje kar 90% regulacijskih problemov v procesni industriji. Znano je, da je PID regulator precej robusten na morebitne spremembe v procesu ali njegovi okolici. Če pa so te spremembe vendarle znatnejše, moramo uporabiti t.i. *adaptivne regulatorje*, ki se sproti prilagajajo tem spremembam. Nadalje lahko številne regulacijske metode uvrstime med t.i. metode vodenja na osnovi modela realnega procesa. Na tem področju so trenutno najaktualnejše *prediktivne metode*, ki s pomočjo vgrajenega optimizacijskega postopka napovedo bodoče vrednosti regulirnega signala. Posebno regulacijsko problematiko predstavljajo procesi z več vhodi in izhodi, pri čemer nastopajo navzkrižni vplivi med vhodi in izhodi. V takih primerih uporabimo *multivariabilne regulatorje*.

7.3.3.2 Proporcionalna - integrirno-diferencirna regulacija

Proporcionalno-integrirno-diferencirna regulacija (*PID* regulacija) je zlasti v procesni industriji najpogosteje uporabljeni algoritem. Odlikuje se po enostavni izvedbi, dobrih dinamičnih lastnostih in precejšnji robustnosti med delovanjem. Algoritem običajno izvedemo z vzporedno kombinacijo proporcionalnega (*P*), integrirnega (*I*) in diferencirnega (*D*) člena. Algoritem podaja enačba

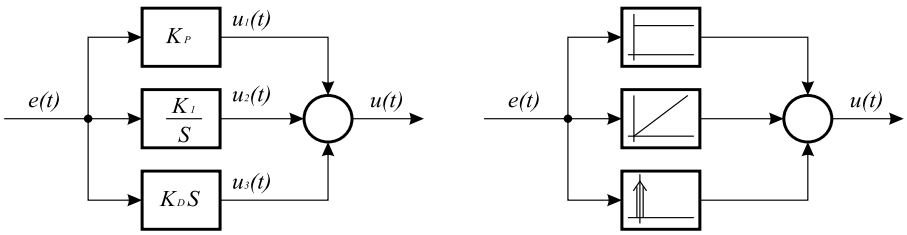
$$u(t) = K_P e(t) + K_I \int e(t)dt + K_D \frac{de(t)}{dt} = K_P \left(e(t) + \frac{1}{T_I} \int e(t)dt + T_D \frac{de(t)}{dt} \right) \quad (7.109)$$

oz. prenosna funkcija

$$G_R(s) = \frac{U(s)}{E(s)} = K_P \left(1 + \frac{1}{T_I s} + T_D s \right) \quad (7.110)$$

K_P, K_I in K_d so ojačenja proporcionalnega, integrirnega in diferencirnega člena, T_I je integrirni čas, T_D pa diferencirni čas.

Regulirna veličina je torej vsota členov, ki so sorazmerni pogrešku, integralu pogreška in diferencialu pogreška. V bločnih diagramih prikažemo regulator z enim od načinov, ki jih prikazuje Sl. 7.72. V bloke vpišemo ustrezne prenosne funkcije ali pa ikone, ki prikazujejo odziv posameznih delov regulacijskega algoritma na enotino stopnico.



Sl. 7.72. Možna bločna diagrama PID regulacije

V praksi razen popolnega *PID* algoritma uporabljamo še različne delne izvedbe: *P*, *PI* ali *PD* regulacijo. *I* člen, še bolj pa *D* člen se redko uporabljata kot samostojna člena.

PID algoritem (enačbi (7.109) in (7.110)) vsebuje t.i. idealni diferenciator (diferencirnemu členu pravimo tudi člen *D0*). V praksi *D* člena na tak način ne moremo niti nočemo realizirati, saj bi tako izveden regulator ob hitrih spremembah referenčnega signala ali motenj povzročil izdatne spremembe regulirne veličine, kar bi povzročilo nasičenje izvršnih členov. Prav tako se v takem primeru izdatno ojačujejo visokofrekvenčne motnje. Zato se v praksi uporablja *PID* regulator, ki vsebuje t.i. *D1* člen, t.j. diferencirni člen z dodatnim polom. Prenosna funkcija je v tem primeru

$$G_R(s) = K_P \left(1 + \frac{1}{T_I s} + \frac{T_D s}{T' s + 1} \right) \quad (7.111)$$

Namesto o ojačenju k_P zlasti pri regulatorjih, ki jih uporabljamo v procesni industriji, pogosto govorimo o proporcionalnem območju regulatorja (*PB* - proportional band, včasih oznaka x_P). Le-to je definirano kot območje regulirane veličine (ali pogreška), ki izkoristi 100% območje regulirne veličine. Lahko ga podajamo v enotah regulirane veličine ali v % celotnega merilnega območja regulirane veličine. Zveza med ojačenjem

in proporcionalnim območjem je obratno sorazmerna

$$K_P = \frac{100\%}{PB[\%]} \tag{7.112}$$

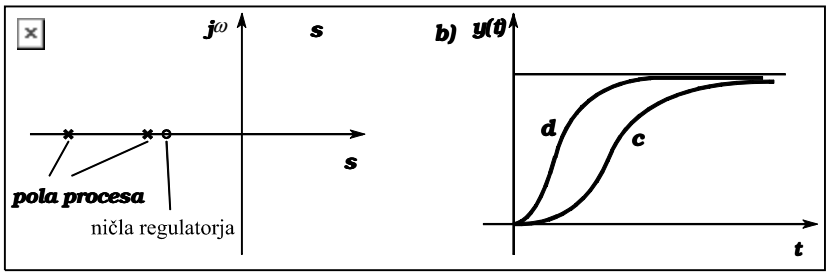
Če ima npr. regulator proporcionalno območje *PB* = 100%, to pomeni, da 100% področje regulirane veličine (npr. tokovni signal 4-20 mA) pretvori v 100% področje regulirane veličine(npr. tudi 4-20 mA), oz. da ima ojačenje $k_P=1$.

Vloga parametrov v PID algoritmu

P člen je osnovni člen v algoritmu. Omogoča hitro izreguliranje pogreška, vendar ponavadi ne odpravlja pogreška v ustaljenem stanju. Zato dodamo *I* člen. S tem regulacijski sistem sicer postane nekoliko počasnejši in manj dušen (bolj nihajoč), vendar pa tak regulator uspešno odpravlja pogrešek v ustaljenem stanju. Vlogo *D* člena lepše vidimo, če zapišemo *PD* algoritem

$$G_R(s) = K_P(1 + T_D s) \tag{7.113}$$

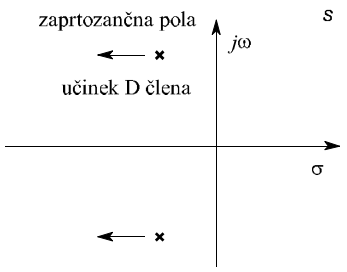
Prenosna funkcija regulatorja ima torej ničlo pri $s = -1/T_D$. *D* člen učinkuje v skladu s spremembo pogreška, deluje prediktivno in izvaja hitre popravke v regulacijski zanki, še preden pogrešek znatneje naraste. Če je proces nadkritično dušen (dominantni poli so realni), lahko ob pravilni nastavitvi *PD* člen v regulatorju s svojo ničlo (enačba (7.113)) delno kompenzira dominantni oz. počasni pol procesa, kar prikazuje Sl. 7.73.



Sl. 7.73. Kompenzacija počasnega pola procesa s PD členom regulatorja:
a) poli in ničla v s ravnini;
b) odziv procesa na stopničasto vzbujanje (krivulja c) in
c) odziv kompenziranega sistem (v odprti zanki) (krivulja d)

Ker sistem v odprti zanki postane hitrejši, postane hitrejši tudi povratnozančni sistem.

Primerno nastavljen *D* člen v *PD* ali *PID* regulatorju lahko poveča dušenje regulacijskega sistema. Če je zaprtozančni regulacijski sistem podkritično dušen (odziv je določen s konjugirano kompleksnim parom dominantnih polov), potem *D* člen pomakne korena nekoliko proti levi. Sistem zato postane bolj dušen in stabilnejši. Učinek prikazuje Sl. 7.74.



Sl. 7.74. Učinek D člena v PD ali PID regulatorju

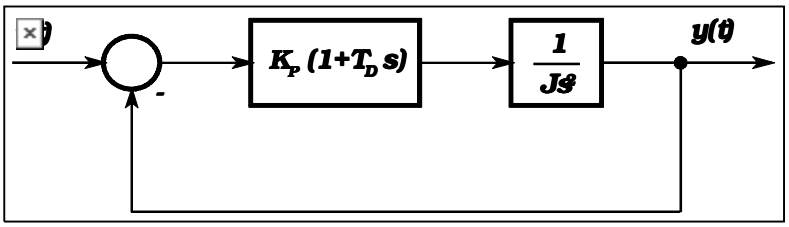
∇

Primer 7.13: *Proporcionalno - diferencirna regulacija zasuka rotacijskega mehanskega sistema brez dušenja*

Če ima rotacijski mehanski sistem zanemarljivo dušenje ($b \cong 0$), ga lahko opišemo z diferencialno enačbo

$$J\frac{d^2 y(t)}{dt^2} = T(t) \tag{7.114}$$

Pri tem je *y(t)* zasuk sistema, *T(t)* pa vzbujalni moment. Sl. 7.75 prikazuje bločno regulacijsko shemo v primeru, če bi zasuk regulirali s *PD* regulatorjem.

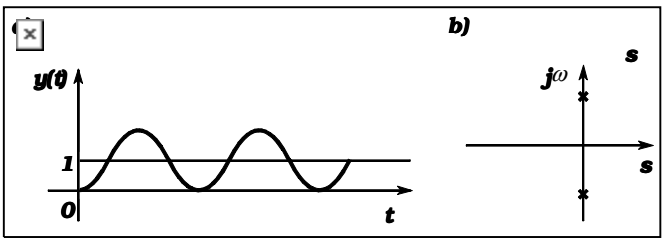


Sl. 7.75. PD regulacija zasuka

V primeru *P* regulacije ($T_D=0$) je prenosna funkcija zaprtozančnega sistema

$$\frac{Y(s)}{R(s)} = \frac{K_P}{Js^2 + K_P} \tag{7.115}$$

Regulacijski sistem ima dva imaginarna konjugirano kompleksna pola. Potek regulirane veličine pri stopničasti spremembi reference in ustrezno lego polov prikazuje Sl. 7.76.

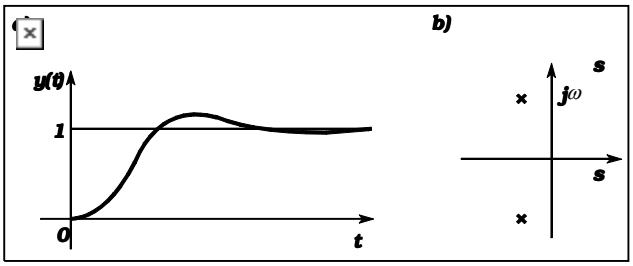


Sl. 7.76. a) časovni potek zasuka pri P regulatorju;
b) lega zaprtozančnih polov

Z ojačenjem *P* regulatorja lahko vplivamo le na frekvenco nihanja (gibanje korenov po imaginarni osi), ne pa tudi na stabilnost. Če dodamo *D* člen, pa je prenosna funkcija zaprtozančnega sistema

$$\frac{Y(s)}{R(s)} = \frac{K_P(1+T_D s)}{Js^2 + K_P T_D s + K_P} \tag{7.116}$$

Kot smo pričakovali, je *D* člen uvedel dušenje (člen $k_P T_D s$ v imenovalcu), zato sta se korena iz imaginarne osi premaknila v levo polravnino. Odziv regulacijskega sistema in lego polov za ta primer prikazuje Sl. 7.77.



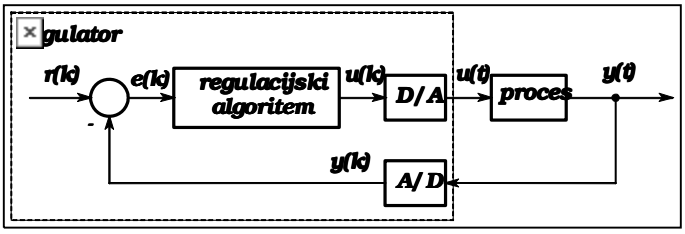
Sl. 7.77. a) časovni potek zasuka pri PD regulatorju; b) lega zaprtozančnih polov

Δ

<u>Diskretna izvedba PID algoritma</u>

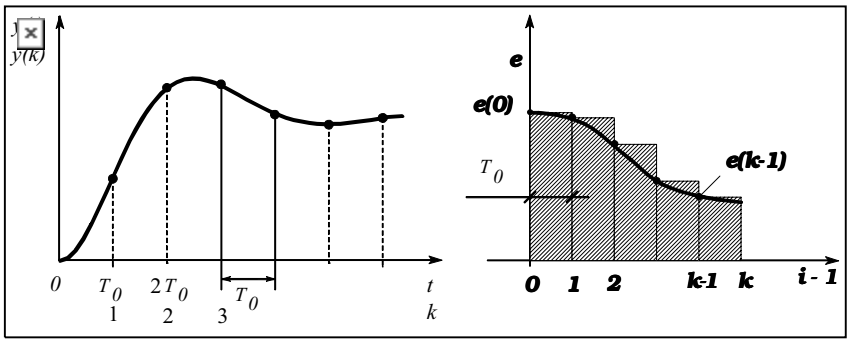
Klasične hidravlične, pnevmatske in analogne elektronske regulatorje danes vse bolj zamenjujejo diskretno (računalniško, digitalno) izvedeni regulacijski algoritmi. Taki regulatorji so izjemno fleksibilni, saj je možno s spreminjanjem podatkov ali programa enostavno spreminjati njihove parametre ali njihovo strukturo. Zato pa potrebujemo dodatne pretvorniške elemente, ki omogočajo prenos regulirane veličine (oz. njenih vzorcev) v računalnik in prenos izračunane regulirne veličine na izvršni sistem. To so A/D (analogno-digitalni) in D/A (digitalno-anologni) pretvorniki.

Ustrezno bločno shemo prikazuje Sl. 7.78, povezavo med zvezno *y(t)* in vzorčno regulirano veličino *y(k)* pa Sl. 7.79 (levo).



Sl. 7.78. Bločna shema regulacijskega sistema z diskretno izvedenim regulacijskim algoritmom

Ker A/D pretvornik jemlje vzorce s časom vzorčenja T_0, so *r(k)*, *e(k)*, *u(k)* in *y(k)* ustrezni vzorci v trenutkih 0, T_0, $2T_0$, … Da lahko programsko realiziramo regulacijski algoritem, je potrebno enačbo zveznega *PID* algoritma (enačba (7.109)) diskretizirati.



Sl. 7.79. Povezava med zvezno in vzorčeno veličino (levo) in numerično integriranje pogreška (desno)

Če zamenjamo integral z vsoto (glej Sl. 7.79 desno) odvod pa z diferenco, dobimo enačbo

$$u(k) = K_P\{e(k) + \frac{T_0}{T_I}\sum_{i=1}^{k} e(i-1) + \frac{T_D}{T_0}[e(k) - e(k-1)]\} \tag{7.117}$$

Če zapišemo enačbo (7.117) še za (*k*-1) in odštejemo obe enačbi, dobimo diskretni *PID* algoritem v obliki diferenčne enačbe

$$u(k) = u(k-1) + q_0 e(k) + q_1 e(k-1) + q_2 e(k-2) \tag{7.118}$$

oz. diskretne prenosne funkcije

$$G_R(z) = \frac{U(z)}{E(z)} = \frac{q_0 + q_1 z^{-1} + q_2 z^{-2}}{1 - z^{-1}} \tag{7.119}$$

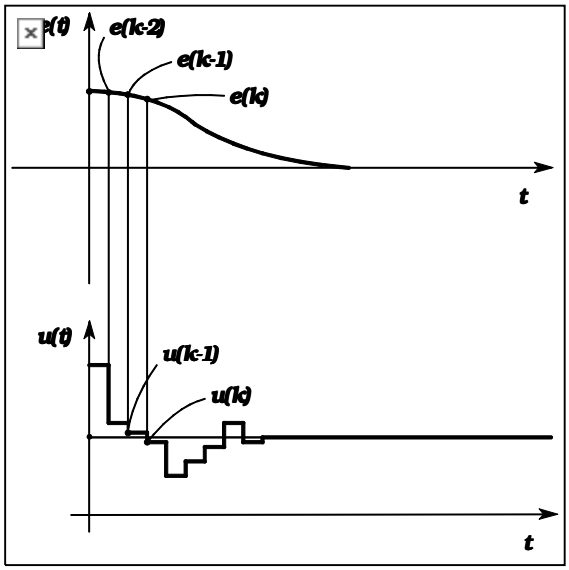
kjer so

$$q_0 = K(1+\frac{T_D}{T_0}) \quad q_1 = -K(1+2\frac{T_D}{T_0}-\frac{T_0}{T_I}) \quad q_2 = K\frac{T_D}{T_0} \tag{7.120}$$

Enačba (7.118) jasno kaže, kako se programsko realizira *PID* regulacijski algoritem. Za izračun sedanje vrednosti regulirne veličine *u(k)* potrebujemo preteklo vrednost *u(k*-1*)* ter sedanjo vrednost signala pogreška *e(k)* in dve pretekli vrednosti *e(k*-1*)*, *e(k*-2*)*. Sl. 7.80 prikazuje tipični signal pogreška in regulirni signal pri računalniškem vodenju procesov (stopničasta sprememba referenčnega signala, proporcionalni proces).

<u>Načrtovanje zvezne PID regulacije</u>

Ker imajo *PID* regulatorji vnaprej določeno strukturo, se postopek sinteze regulacijskih algoritmov poenostavi na določevanje njegovih parametrov. Postopek določanja parametrov je predvsem odvisen od tega ali imamo na voljo model procesa ali ne. Če model procesa ne obstaja, smo vezani predvsem na t.i. eksperimentalne metode, ki so znane tudi kot *nastavitvena pravila*.



Sl. 7.80. Signal pogreška in regulirni signal pri vodenju z diskretnim PID regulatorjem

Če pa je model procesa znan, imamo večjo izbiro možnosti. Lahko uporabljamo nastavitvena pravila na modelu procesa, lahko uporabljamo nekatere analitične postopke, ki temeljijo na zagotovitvi ustrezne lege zaprtozančnih polov ali na kompenzaciji v frekvenčnem prostoru, lahko pa uporabimo tudi računalniško optimizacijo.

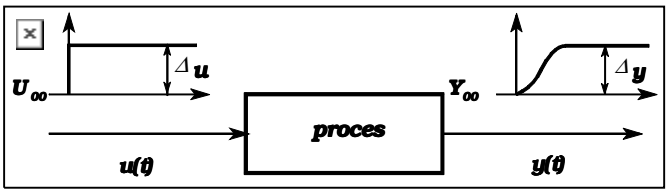
Razen zagotovitve stabilnosti kot osnovne zahteve pri vsakem načrtovanju pa moramo z regulatorjem, kot smo že omenili, izpolniti še druge zahteve za ustrezno kvaliteto

regulacije: hitro sledenje referenci, čim manjši (po možnosti ničelni) pogrešek v ustaljenem stanju, majhni prevzponi pri hitrih spremembah reference, hitro izreguliranje motenj, robustno delovanje pri spremembah parametrov procesa, manjše učinkovanje nelinearnosti v zanki, itd.

Nastavitvena pravila Zieglerja in Nicholsa

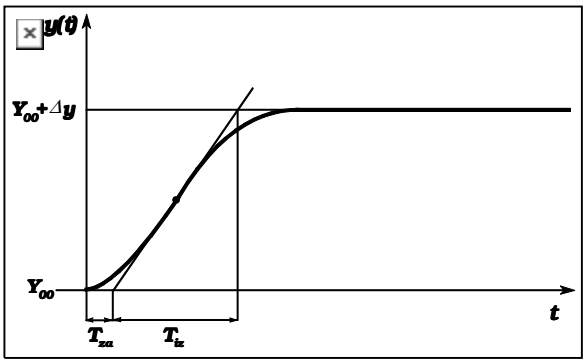
Ziegler in Nichols priporočata ustrezno nastavitev parametrov s pomočjo določenih eksperimentov na realnem objektu ali na modelu procesa. Znana sta dva postopka: določitev s pomočjo odziva procesa na stopnico in postopek z nihajnim preizkusom (Ziegler in Nichols, 1942).

Metoda s pomočjo odziva na stopničasto vzbujanje spada med t.i. odprtozančne metode, ker zahteva eksperiment na odprtozančnem procesu. Primerna je za proporcionalne procese, ki pa morajo biti nekoliko nadkritično dušeni. Proces je potrebno vzbujati z enotino stopnico in meriti odziv. Ustrezen postopek prikazuje Sl. 7.81.



Sl. 7.81. Merjenje odziva procesa na enotino stopnico

Tak posnetek je možno izvesti neposredno na objektu kar med obratovanjem. Regulator postavimo v položaj ročno, nakar vzbudimo proces s stopničasto spremembo vhodnega signala in posnamemo odziv. Običajno so dopustne majhne spremembe. Karakteristični odziv, ki je primeren za uporabo tega kriterija, prikazuje Sl. 7.82.



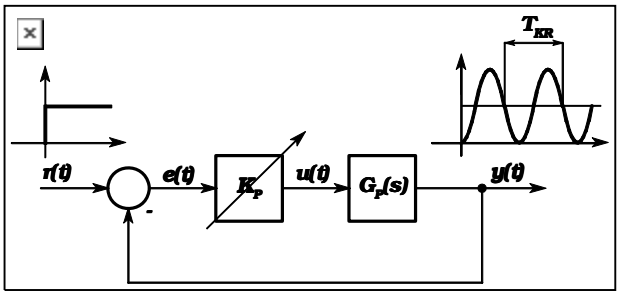
Sl. 7.82. Odziv proporcionalnega procesa

Postopek zahteva, da narišemo tangento v prevojni točki odziva. S pomočjo presečišč tangente z absciso in premico $y(t)=Y_{00}+\Delta y$ določimo čas zaostajanja T_{za} in čas izravnave T_{iz}. Ziegler in Nichols sta na osnovi teh parametrov priporočala nastavitve parametrov *PID* regulacije, kot prikazuje Tabela 7.7.

Tabela 7.7: Nastavitev parametrov PID regulacije po metodi Ziegler-Nichols s pomočjo odziva procesa na stopnico

Vrsta regulacije	K_P	T_I	T_D
P	$\frac{1}{K}\frac{T_{iz}}{T_{za}}$	∞	0
PI	$\frac{0.9}{K}\frac{T_{iz}}{T_{za}}$	$3.3\ T_{za}$	0
PID	$\frac{1.2}{K}\frac{T_{iz}}{T_{za}}$	$2\ T_{za}$	$0.5\ T_{za}$

Metoda s pomočjo nihajnega preizkusa zahteva poizkus v zaprti zanki. Pri metodi nihajnega preizkusa nastavimo $T_I=\infty$ in $T_D=0$, torej uporabimo v zanki le proporcionalni regulator. Regulacijskemu sistemu pri poljubnem vzbujanju (običajno konstantne spremembe reference ali motnje) povečujemo ojačenje K_P toliko časa, da regulirana veličina nedušeno zaniha. Ustrezen postopek prikazuje Sl. 7.83.



Sl. 7.83. Nihajni preizkus

Nihajni preizkus je možno izvesti le v primeru, če je proces vsaj tretjega reda ali če vsebuje mrtvi čas. Ojačenju, pri katerem regulacijski sistem nedušeno zaniha, pravimo kritično ojačenje K_{KR}. Iz posnetega odziva ali s pomočjo časovnega merjenja dobimo periodo nihanja, ki jo imenujemo kritična perioda T_{KR}. Na osnovi kritičnega ojačenja in kritične periode sta Ziegler in Nichols predlagala parametre *PID* regulacije, ki jih prikazuje Tabela 7.8.

Tabela 7.8 Nastavitev parametrov z nihajnim preizkusom po metodi Ziegler - Nichols

Vrsta regulacije	K_P	T_I	T_D
P	$0.5\ K_{KR}$	∞	*0*
PI	$0.45\ K_{KR}$	$0.83\ T_{KR}$	*0*
PID	$0.6\ K_{KR}$	$0.5\ T_{KR}$	$0.125\ T_{KR}$

Pravila Zieglerja in Nicholsa so narejena za cel spekter sorodnih procesov. Pri stopničasti referenci dajo 10% - 60% prevzpon. Povprečni prevzpon, ugotovljen pri uporabi na različnih procesih, je 25%. Zato je razumljivo, da nobeno nastavitveno pravilo ne da optimalnih rezultatov v konkretnem primeru. Njihov pomen pa je v tem, da dobimo okvirne vrednosti parametrov. Le-te potem običajno eksperimentalno na objektu ali simulacijsko (če imamo model) še izboljšamo. Slabost metod Zieglerja in Nicholsa pa je v tem, da ni definirano, za kakšne vhodne signale naj regulacija dobro deluje. Optimalno delovanje za stopničasto spremembo reference zahteva namreč drugačno nastavitev parametrov kot optimalno delovanje pri pojavu motnje.

Pri obravnavanih nastavitvenih pravilih smo upoštevali idealni *PID* regulacijski algoritem. Ker pa v praksi vedno uporabljamo obliko, ki jo podaja enačba (7.112), nastopa še dodatni parameter T'. Tega ponavadi nastavimo tako, da velja

$$T' \cong (0.1 \text{ do } 0.3)\ T_D \tag{7.121}$$

Nastavitvena pravila ob uporabi Bodejevega diagrama

P, *PI* ali *PID* regulator lahko uglasimo s pomočjo Bodejevega diagrama odprtozančne prenosne funkcije *G(s)H(s)* v primeru, če je le-ta vsaj drugega reda. Nastavitvena pravila temeljijo na tem, da zagotovijo potrebni fazni razloček.

P regulator

Nastavitvena pravila priporočajo, da določimo absolutno vrednost frekvenčne karakteristike pri faznem kotu -162°. Ojačenje v regulacijski zanki izberemo tako, da je

$$K_P = \frac{1}{|G(j\omega')H(j\omega')|} \qquad \angle[G(j\omega')H(j\omega')] = -162^o \tag{7.122}$$

Frekvenca, pri kateri je faza -162°, postane torej frekvenca ω_l kompenziranega sistema, fazni razloček pa je $\phi_m = 18^0$.

PI regulator

Ojačenje v regulacijski zanki izračunamo s pomočjo absolutne vrednosti frekvenčne karakteristike pri faznem kotu -144^0

$$K_P = \frac{1}{\left|G(j\omega'')H(j\omega'')\right|} \qquad \angle\left[G(j\omega'')H(j\omega'')\right] = -144^o \tag{7.123}$$

Na ta način sicer na prvi pogled izgleda, da smo dosegli večji fazni razloček, vendar pa *I* člen vnaša zaostajanje v fazno karakteristiko, kar zmanjša fazni razloček. Integrirni čas T_I izračunamo tako, da predhodno določimo frekvenco ω', kjer je fazni kot -162^0

$$T_I = \frac{2\pi}{\omega'} \qquad \angle\left[G(j\omega')H(j\omega')\right] = -162^o \tag{7.124}$$

PID regulator

Konstanti K_P in T_I določimo kot pri *PI* regulatorju. Diferencirni čas T_D pa izberemo

$$T_D = 0.1T_I \tag{7.125}$$

Optimizacija regulacijskega sistema

Optimizacija regulacijskega sistema je postopek, s pomočjo katerega načrtamo parametre regulacijskega algoritma tako, da le-ta minimizira ali maksimizira (odvisno od situacije) določeno cenilko (kriterijsko funkcijo, performančni indeks) pri določenem vzbujanju regulacijskega sistema oz. pri določenem prehodnem pojavu. Ob tem je potrebno upoštevati omejitve. Čeprav jo je v principu možno izvajati tudi neposredno na realnem objektu, jo v praksi skoraj vedno izvajamo na modelu realnega objekta. Ker je v primeru *PID* regulatorja potrebno določiti le parametre, ne pa tudi strukture, pravimo taki optimizaciji parametrska optimizacija.

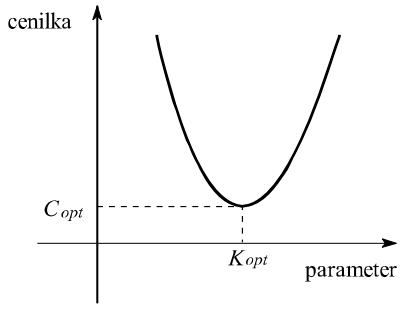
Cenilka je običajno funkcija integralskega tipa odvisna od signala pogreška in od regulirne veličine. Omejitev pa je posledica upoštevanja realnih lastnosti regulacijskih sistemov, ki jih sicer pri teoretični obdelavi včasih radi zanemarimo. To so predvsem fizične omejitve izvršnih sistemov, merilnih sistemov in regulatorjev (npr. 4 do 20 *mA*).

Samo najenostavnejše probleme, ki so ponavadi zgolj teoretičnega pomena, je možno rešiti analitično. V tem primeru cenilko, ki jo določimo v nekem prehodnem pojavu (npr. $C = \int e^2(t)dt$ pri stopničasti spremembi reference), zapišemo kot funkcijo parametrov *PID* regulacije

$$C = f(K_P, T_I, T_D) \tag{7.126}$$

Če želimo določiti optimalno ojačenje, je potrebno cenilko odvajati po ojačenju in odvod izenačiti z nič

$$\frac{\partial C}{\partial K_P} = 0 \Rightarrow K_{Popt} \tag{7.127}$$



Sl. 7.84. Vrednost cenilke v odvisnosti od parametra

Za realne probleme pa danes v glavnem uporabljamo računalniško parametrsko optimizacijo. To je poseben algoritem, ki pri ponavljajočih simulacijah prehodnih pojavov toliko časa spreminja parametre regulatorja, da na koncu zagotovi minimum cenilke (glej Sl. 7.84).

Glavna slabost postopka je velika potratnost računalniškega časa, saj vsaka nova kombinacija parametrov regulatorja zahteva nov simulacijski tek oz. nov prehodni pojav, v katerem se ovrednoti vrednost cenilke. Torej je pri reševanju realnih problemov potrebno regulacijski sistem simulirati, saj je nemogoče analitično izraziti vrednost cenilke v odvisnosti od parametrov. Ker je pri reševanju kompleksnejših problemov potrebno izvesti tudi nekaj sto poizkusov - simulacijskih tekov, je razumljivo, kako neobhoden je sposoben računalnik. Z razvojem računalnikov pa ta slabost postaja vse manj pomembna.

Praktični problemi pri delovanju regulatorjev

Do sedaj smo se v glavnem omejili na najbolj vitalni, t.j. na regulacijski algoritem. Vendar pa je potrebno za uspešno delovanje na realnem procesu upoštevati še celo vrsto dejavnikov. Ti se zlasti enostavno lahko upoštevajo pri računalniški realizaciji regulatorjev. Če analiziramo programsko opremo nekega računalniško izvedenega industrijskega regulatorja, vidimo, da regulacijski algoritem predstavlja manjši del celotnega programskega modula. Preostali del je običajno namenjen filtriranju vhodnih signalov, realizaciji preklopa ročno - avtomatsko, zaščiti pred integralskim pobegom, realizaciji direktnega ali reverznega delovanja, itd.

Filtriranje vhodnih signalov

Industrijski regulatorji delujejo v zelo motenem okolju. Zato pretvorjena regulirana veličina, ki jo regulator sprejema, običajno vsebuje motnje. Zaradi tega imajo regulatorji na vhodu nizkopasovne filtre, ki izločajo motilne signale. Pri tem je potrebno paziti, da je pasovna širina filtra dovolj velika, da filter preveč ne zmanjša hitrega učinkovanja P in D člena regulatorja.

Preklop ročno - avtomatsko

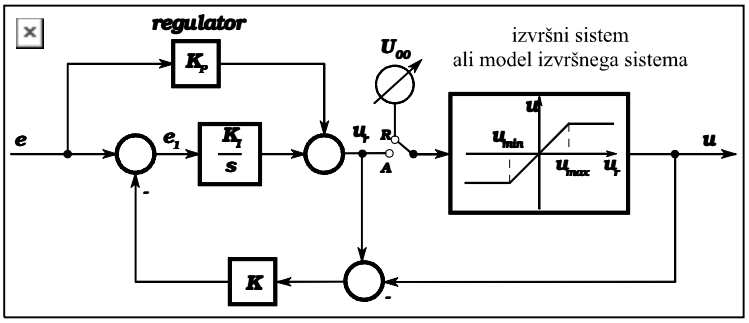
Vsak industrijski regulator omogoča, da ga lahko preklapljamo med ročnim in avtomatskim načinom. V ročnem načinu delovanja ima nadzor nad procesom operater, ki ročno s pomočjo posebnega gumba, vodi proces (preko izvršnega sistema) tako, da regulirana veličina dobi želeno (delovno) vrednost. Nato običajno preklopi regulator v stanje avtomatsko. V avtomatskem načinu delovanja pa odloča o vrednosti regulirnega signala regulacijski algoritem. Ročni način delovanja je najbolj značilen pri zagonu procesa, kajti funkcija regulatorja je običajno le, da ustrezno deluje v delovni točki.

Sodobni regulatorji omogočajo preklop med ročnim in avtomatskim delovanjem in obratno brez udarov (bumpless transfer).

Integralski pobeg in ustrezna zaščita

Dinamično področje delovanja členov izvršnih sistemov je vedno omejeno. Tipični elementi so npr. ventili, ki jih lahko reguliramo le med popolnim zaprtjem in popolnim odprtjem. Če se v povezavi s takimi elementi uporablja integrirni člen v regulatorju, lahko pride do t.i. integralskega pobega. Izvršni sistem pride med reguliranjem v nasičenje, zato se pogrešek mnogo počasneje manjša, kot če nasičenja ne bi bilo. Zaradi integracije pogreška izhod *I* člena regulatorja, in s tem regulatorja, dobi velike vrednosti, kar pa seveda nima nikakršnega učinka na regulirano veličino. Ko kljub počasnejšemu prehodnemu pojavu pogrešek postane nič, mora nekaj časa delovati še negativni pogrešek, da zmanjša vrednost na integrirnem členu regulatorja. Zato je regulirna veličina precej dolgo v nasičenju, regulirana veličina pa ima velik prevzpon v prehodnem pojavu.

Glavna ideja zaščite pred integralskim pobegom je v tem, da je potrebno zaznati, kdaj izvršni sistem preide v nasičenje. V trenutku, ko se to dogodi, je potrebno zaustaviti integracijo *I* dela regulatorja, oz. vplivati na *I* člen tako, da bo zagotavljal natančno tako regulirno veličino, ki bo pripeljala izvršni sistem na mejo nasičenja. Metode zaščite zahtevajo, da poznamo model izvršnega sistema oz. parametra u_{min} in u_{max} pri katerih pride izvršni sistem v nasičenje. Sl. 7.85 prikazuje najpogostejšo izvedbo, ki razen zaščite pred integralskim pobegom zagotavlja tudi brezudarni preklop.



Sl. 7.85. Preklop ročno - avtomatsko in zaščita pred integralskim pobegom

V stanju ročno da integrirni del regulatorja natančno tak prispevek regulirni veličini, da je vsota proporcionalnega in integrirnega signala regulatorja u_r enaka signalu U_{00} (signal za ročno vodenje rocesa) oz. signalu u (predpostavimo naklon omejitvene karakteristike 45^0). Vsaka razlika se izniči preko močne povratne vezave okoli integrirnega dela. Pri preklopu v način avtomatsko tako ni nikakršnega udara. Če je v avtomatskem delovanju izvršni sistem v nasičenju ($u_r>u_{max}$ oz. $u_r<u_{min}$), velja zaradi močne povratne zanke okoli integratorja $u=u_{max}$ oz. $u=u_{min}$. Na ta način smo preprečili integralski pobeg.

Direktno in reverzno delovanje regulatorja

V do sedaj obravnavanih regulacijskih shemah se je regulirna veličina povečevala, če se je regulirana veličina zmanjševala (če je npr. temperatura upadla, se je povečala moč grelnika). Na ta način dosežemo negativno povratno zanko in stabilno delovanje s pomočjo regulacijskega dela. Takemu delovanju regulatorja pravimo direktno delovanje. Včasih pa je potrebno povečati regulirno veličino v primeru, če regulirana veličina narašča. Tak primer imamo npr. pri hlajenju jedrskega reaktorja s hladilno tekočino. Če temperatura v reaktorju narašča, je potrebno bolj odpreti ventil za dovajanje hladilne tekočine. Takemu načinu delovanja pa pravimo reverzno delovanje regulatorja. Le-to je možno doseči tako, da definiramo pogrešek kot $e=y-r$ (doslej smo vedno upoštevali $e=r-y$), ali pa, da pomnožimo izhod regulatorja z (-1). V tem primeru regulator torej ne obrača predznaka, ker se le-ta obrača v karakteristiki aktuatorja, končnega izvršnega člena ali procesa. Sodobne regulatorje lahko konfiguriramo za direktno ali reverzno delovanje.

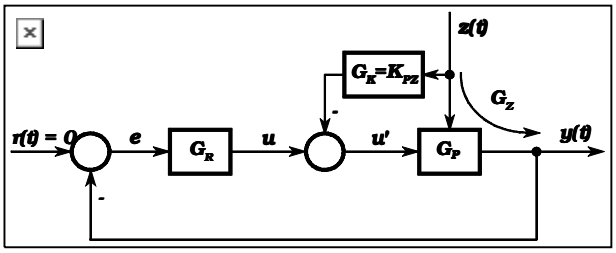
<u>Večzančni regulacijski sistemi</u>

Doslej obravnavani regulacijski sistemi so vsebovali eno regulacijsko zanko, v kateri so nastopali parametrsko optimalni regulatorji. Z optimalno nastavitvijo njihovih parametrov izčrpamo vse možnosti, ki jih daje tak način regulacije. Vendar pa lahko tudi optimalno nastavljen *PID* regulator odpove, če v procesu nastopajo velike motnje ali ima le-ta veliko časovno zakasnitev. Eno od možnih rešitev predstavlja vključitev dodatnih zank v regulacijski sistem, kar pomeni, da spremenimo strukturo regulacijskega sistema. Na kratko bomo opisali vključitev krmiljenja v regulacijsko zanko in kaskadno regulacijo. Vse metode v glavnem temeljijo na tem, da zmanjšajo zakasnitve v regulacijskem sistemu, in da učinkoviteje odpravljajo motnje. Z zmanjšanjem zakasnitev se zmanjšajo problemi stabilnosti, zato lahko povečamo ojačenje regulatorja, kar dodatno ugodno vpliva na delovanje regulacijskega sistema.

Krmiljenje z upoštevanjem motnje

Pri doslej obravnavanem načinu regulacije se učinki motnje sicer zmanjšajo, vendar šele potem, ko motnja že vpliva na regulirano veličino. Šele takrat namreč regulator dobi informacijo, da je potrebno posredovati s tako regulirno veličino, da bo le-ta zmanjšala vpliv motnje. Pri velikih časovnih konstantah procesa je tak način delovanja neučinkovit.

V smislu kvalitete regulacije bi bilo seveda bolje, da ne bi čakali na spremembo regulirane veličine, ampak bi motnjo upoštevali takoj, ko nastopi. Pri tem bi krmilnik deloval na izvršni sistem v taki obliki, da bi se čim hitreje kompenziral njen vpliv. Tak postopek je seveda možen le v primeru merljive motnje. Imenujemo ga krmiljenje z upoštevanjem motnje. V obravnavanem postopku krmiljenje nastopa le kot dodatek osnovni regulacijski zanki. Zato je krmilni sistem lahko enostaven, največkrat kar proporcionalni sistem. Ustrezno bločno shemo prikazuje Sl. 7.86.



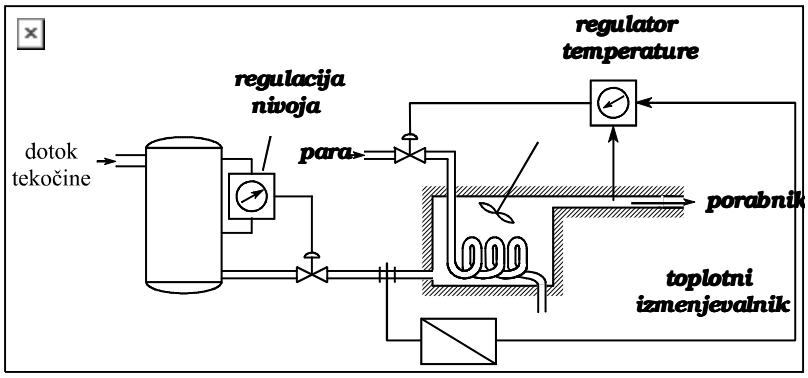
Sl. 7.86. Proporcionalno krmiljenje z upoštevanjem motnje

Naloga takega sistema je, da že na izvršnem mestu čim bolj kompenzira učinek motnje na regulirano veličino. Ker kompenzacija nikoli ni popolna, poskrbi za natančno delovanje regulacijska zanka.

∇

Primer 7.14: *Regulacija temperature v toplotnem izmenjevalniku*

Tehnološko shemo prikazuje Sl. 7.87.



Sl. 7.87. Tehnološka shema temperaturnega regulacijskega sistema

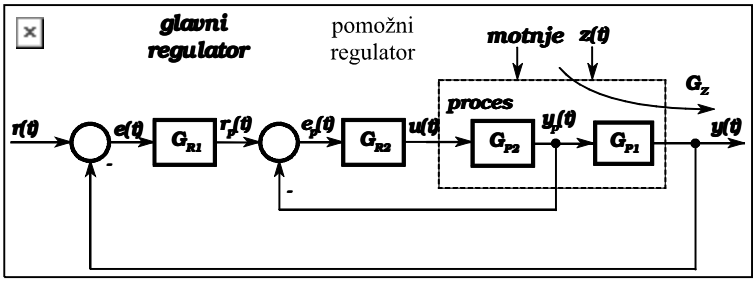
V sistemu toplotni izmenjevalnik ogreva tekočino, ki prihaja iz stolpa k porabniku. Glavni regulator meri temperaturo na izhodu in če le-ta odstopa od želene, spremeni pretok pare v toplotni izmenjevalnik. Ena od možnih motenj v sistemu je npr. motnja v pretoku, ki nastane zaradi spremenjene porabe ogrete tekočine. Zaradi spremembe pretoka se spremeni tudi temperatura tekočine na izhodu, vendar s precejšnjo zakasnitvijo. Zato regulator potrebuje veliko časa, da odpravi motnjo. V sistem lahko

učinkovito uvedemo krmiljenje s tem, da merimo motnjo t.j. spremembo pretoka. Ta sprememba se nemudoma posreduje regulatorju. Le-ta takoj po nastopu motnje spremeni dotok pare v toplotni izmenjevalnik. Krmilni sistem tako skrbi za čim manjši učinek motnje v prehodnem pojavu, regulator pa skrbi za točno nastavitev temperature in za odpravljanje preostalih, nemerljivih motenj.

Δ

Kaskadna regulacija

Kaskadna regulacija predstavlja eno izmed najpogosteje uporabljenih večzančnih regulacij. Osnovni princip prikazuje Sl. 7.88.



Sl. 7.88. Kaskadna regulacija

Razen regulirane veličine $y(t)$ merimo še pomožno regulirano veličino $y_p(t)$. Uporabimo dva regulatorja, ki sta vezana v kaskado. Samo prvi regulator ima nastavljivo referenco (local set up). S pomočjo pogreška $e(t)=r(t)-y(t)$ se izračuna izhod glavnega regulatorja, ki pa ni regulirna veličina, ampak tvori referenčni vhod v pomožni regulator (remote set up). Ta se primerja s pomožno regulirano veličino in iz tako nastalega pogreška pomožni regulator generira regulirno veličino. Pomožni regulator G_{R2} in del procesa G_{P2} tvorita pomožno regulacijsko zanko, glavni regulator pa s pomožno regulacijsko zanko in delom procesa G_{P1} tvori glavno regulacijsko zanko. Tako glavna kot pomožna zanka vsebujeta merilna sistema, vendar ima le glavni regulator neodvisno nastavljivo referenco in le pomožni regulator generira regulirno veličino. Kaskadna regulacija izkazuje boljšo kvaliteto kakor enozančna iz naslednjih vzrokov:

- vplivi motenj, ki se pojavijo v delu procesa G_{P2}, torej pred pomožnim merilnim mestom $y_p(t)$, so zaradi pomožne regulacijske zanke bistveno zmanjšani;
- zaradi pomožne regulacijske zanke je bistveno zmanjšan vpliv sprememb parametrov prenosne funkcije G_{P2}. Pri načrtovanju glavnega regulatorja moramo zato upoštevati predvsem spremembe parametrov izhodnega dela procesa G_{P1};
- zaradi pomožne regulacijske zanke je bistveno zmanjšan vpliv nelinearnosti v procesu G_{P2};
- zaradi pomožne regulacijske zanke postane ta del regulacijskega sistema hitrejši (manjše časovne konstante, višje lastne frekvence), zato tudi celotni regulacijski

sistem deluje hitreje.

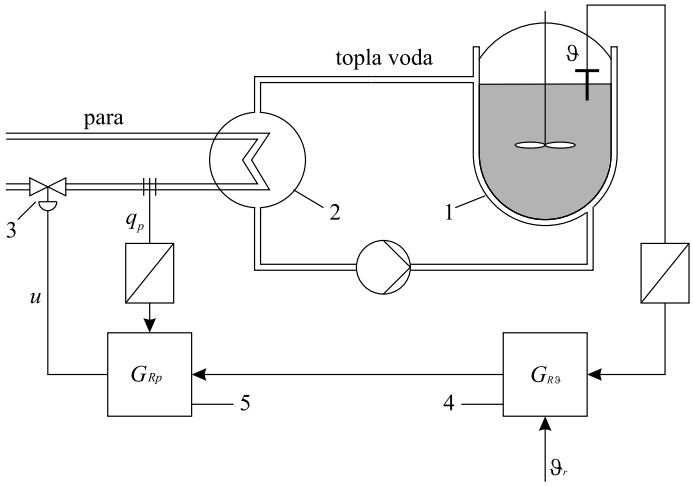
Dejanska izvedba kaskadne regulacije je odvisna predvsem od mesta nastopa motenj. Za pomožni regulator uporabljamo enostavnejšo izvedbo (*P* ali *PI*), za glavni regulator pa *PID* regulator.

∇

Primer 7.15: *Kaskadna regulacija ogrevanja kemijskega reaktorja*

Tehnološko shemo prikazuje Sl. 7.89.

Kemijski reaktor (1) ogrevamo s toplo vodo, ki se nahaja v njegovem plašču. Le-to pa ogrevamo s parnim toplotnim izmenjevalnikom (2). Ena od motenj, ki se pojavlja, je motnja v pretoku pare toplotnega izmenjevalnika. To je motnja v vhodnem delu procesa. Motnjo sicer izloči glavni regulator $G_{R\vartheta}$ (4) s spremembo položaja ventila (3), vendar je tak način precej zakasnjen predvsem zaradi ogrevanja plašča. S klasično *PID* regulacijo težko dosežemo želene rezultate. Zato vgradimo pomožno regulacijsko zanko, kjer na osnovi merjenja pretoka pomožni regulator G_{Rp} (5) hitro učinkuje na položaj ventila (3) oziroma hitro izregulira motnjo v pretoku pare.



Sl. 7.89. Kaskadna regulacija temperature kemijskega reaktorja

Δ

V dosedanjem tekstu smo na kratko obdelali osnovne principe PID regulacije, nekatere razširjene pristope in omenili najbolj pogoste metode načrtovanja. V nadaljevanju pa si oglejmo praktični primer načrtovanja regulacije in sicer za hidravlični sistem, ki smo ga predstavili in zanj razvili matematični model v prvem delu tega poglavja.

∇

Primer 7.16: *Načtovanja regulacije nivoja v hidravličnem sistemu*

Hidravlični sistem je sestavljen iz treh povezanih shranjevalnikov tekočine (glej Sl. 7.10). Postopek modeliranja in linearizacije je opisan v podpoglavju 7.2.1 (Matematično modeliranje procesov). Regulirati želimo nivo H_3 s pomočjo vhodnega pretoka ϕ_I. Razen nivoja H_3 lahko merimo tudi nivo H_1, kar bomo uporabili pri kaskadni regulaciji. Kvaliteto regulacijskega sistema bomo proučevali s simulacijo v primeru sledilnega delovanja (sprememba želenega nivoja) in v primeru regulacijskega delovanja, ko konstantna motnja H_{1m} deluje na spremembo nivoja H_1 (npr. večja masa tekočine v zelo kratkem času poveča volumen in s tem nivo v prvem shranjevalniku).

Linearizirani model opisujeta prenosni funkciji (enačbi (7.36) in (7.43), podpoglavje 7.2.1)

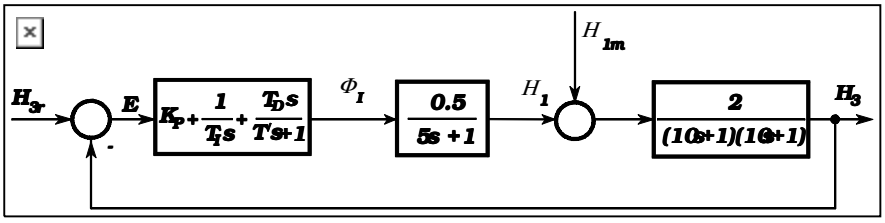
$$G_{P2}(s) = \frac{H_1(s)}{\Phi_I(s)} = \frac{R_1}{R_1 C_1 s + 1}$$

$$G_{P1}(s) = \frac{H_3(s)}{H_1(s)} = \frac{R_3}{R_1(R_2 C_2 s + 1)(R_3 C_3 s + 1)}$$

pri čemer so konstante R_1=0.5, R_2=1, R_3=1, C_1=10, C_2=10, C_3=10.

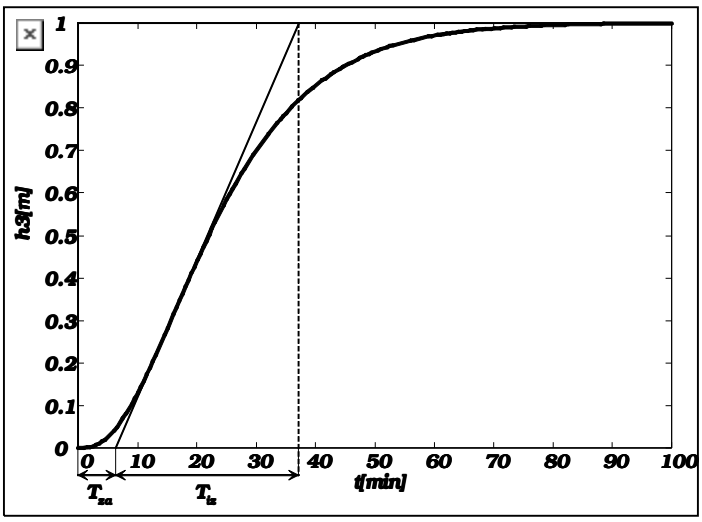
Načrtovanje enozančne regulacije

Sl 7.90 prikazuje bločno shemo pri regulaciji hidravličnega sistema z enozančnim *PID* regulacijskim algoritmom.



Sl. 7.90. Enozančna PID *regulacija*

Za določitev parametrov regulatorja po prvi Ziegler-Nicholsovi metodi potrebujemo odziv procesa (potek nivoja v spodnjem shranjevalniku) na stopničasto vzbujanje (spremembo vhodnega pretoka).



Sl. 7.91. Potek nivoja pri stopničasti spremembi vhodnega pretoka

S tangento v prevojni točki določimo čas zaostajanja in čas izravnave, s pomočjo ustaljenih vrednosti vhodnega in izhodnega signala pa še ojačenje procesa

$$T_{za} = 6.3 \qquad T_{iz} = 31 \qquad K = 1$$

S pomočjo Tabele 7.7 izračunamo parametre *P*, *PI* in *PID* regulacije. Rezultate prikazuje Tabela 7.9.

Tabela 7.9. Parametri P, PI in PID regulacije (odziv na stopničasto vzbujanje)

Vrsta regulacije	K_P	T_I	T_D
P	4.92	∞	0
PI	4.43	20.8	0
PID	5.90	12.6	3.15

Časovna konstanta diferencirnega člena *T'* je v vseh primerih $0.1T_D$.

Z nihajnim preizkusom (druga Ziegler-Nicholsova metoda) pa dobimo kritično ojačenje in kritično periodo

$$K_{KK} = 9 \qquad T_{KK} = 28.2$$

S pomočjoTabele 7.8 izračunamo parametre *P*, *PI* in *PID* regulacije. Rezultate prikazuje Tabela 7.10.

Tabela 7.10. Parametri P, PI in PID regulacije (nihajni preizkus)

Vrsta regulacije	K_P	T_I	T_D
P	4.50	∞	0
PI	4.05	23.4	0
PID	5.40	14.1	3.52

Parametre *PID* regulacije določimo še s pomočjo računalniške optimizacije, tako da minimiziramo cenilko

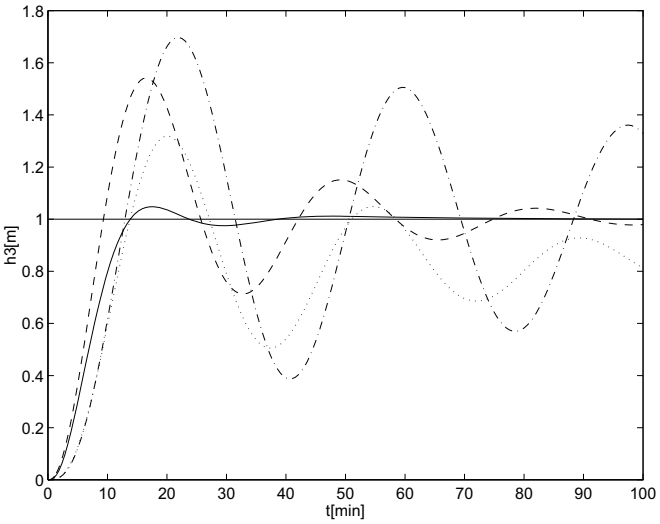
$$ITAE = \int_0^{50} t\left|e(t)\right|dt$$

pri vzbujanju s stopničasto referenco in pri motnji $H_{1m} = 0$. Na ta način dobimo naslednje parametre

$$K_P = 3.06 \qquad T_I = 20.4 \qquad T_D = 7.11$$

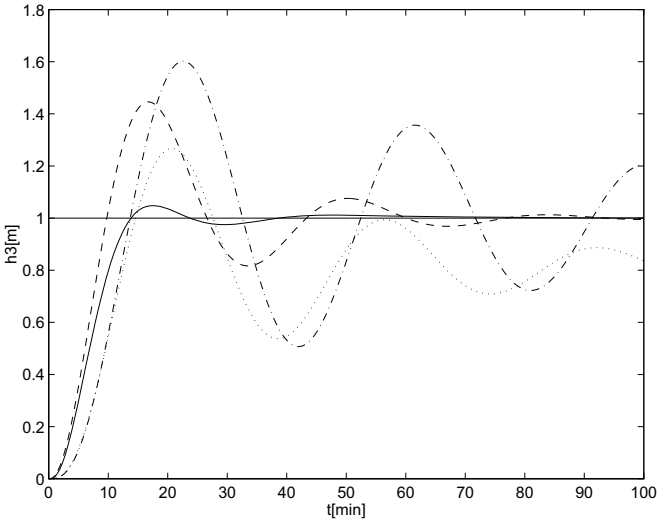
S temi parametri dosežemo cenilko ITAE = 47.2.

Sl. 7.92 prikazuje potek nivoja pri stopničasti spremembi reference pri regulaciji s *P, PI* in *PID* regulatorjem, ki smo jih načrtali iz odziva sistema na stopničasto vzbujanje. Slika prikazuje tudi odziv pri *PID* regulatorju, ki smo ga dobili z optimizacijo.



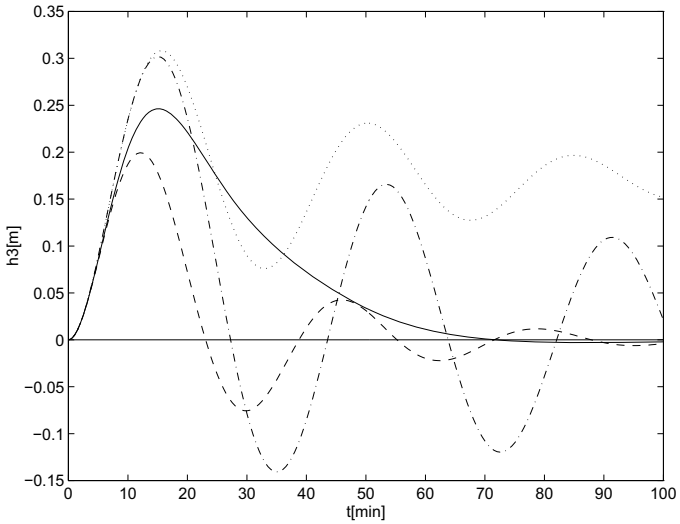
Sl. 7.92. Potek nivoja pri stopničasti spremembi reference (odziv na stopničasto vzbujanje)… P *regulator,* -.-. PI *regulator,* -- - PID *regulator,* __ *optimalni* PID *regulator*

Sl. 7.93 prikazuje potek nivoja pri stopničasti spremembi reference pri regulatorjih, ki smo jih načrtali z nihajnim preizkusom.



Sl. 7.93. Potek nivoja pri stopničasti spremembi reference (nihajni preizkus)… P *regulator,* -.-. PI *regulator,* --- PID *regulator,* __ *optimalni* PID *regulator*

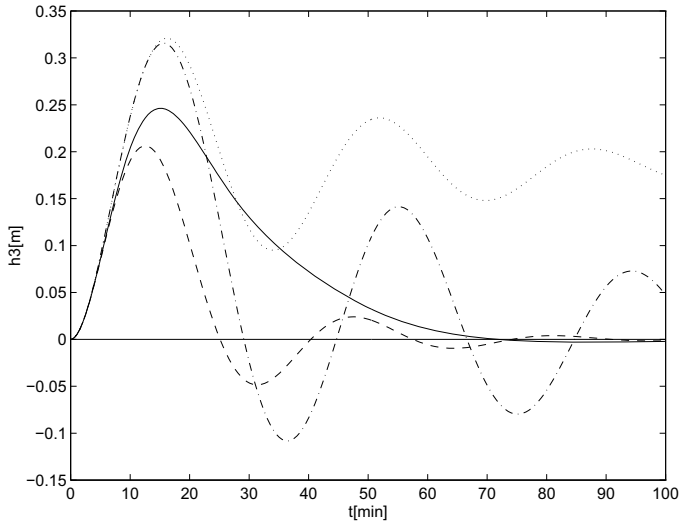
Sl. 7.94 in Sl. 7.95 prikazujeta poteke nivoja pri stopničasti motnji $H_{1m} = 0.5$.



Sl. 7.94. Potek nivoja pri stopničasti motnji (odziv na stopnico) … P *regulator,* -.-. PI *regulator,* --- PID *regulator,* __ *optimalni* PID *regulator*

Iz prikazanih simulacijskih potekov lahko zaključimo:

- metode Ziegler-Nichols niso bile najbolj uspešne, zlasti ne za sledilno delovanje, saj so poteki preveč nihajoči;
- nihajni preizkus da nekoliko uporabnejše rezultate;
- *P* regulator izkazuje pogrešek v ustaljenem stanju. *I* člen odpravi pogrešek, a odzivi postanejo še bolj nihajoči. *D* člen pohitri delovanje regulatorja in vnese dodatno dušenje, tako da kompenzira slabo stran *I* člena;
- regulator, ki smo ga načrtali z optimizacijo, je zelo uspešen pri sledilnem delovanju, pri regulacijskem delovanju pa sta uspešnejša regulatorja po Ziegler-Nicholsu.



Sl. 7.95. Potek nivoja pri stopničasti motnji (nihajni preizkus) … P *regulator,* -.-. PI *regulator,* --- PID *regulator,* __ *optimalni* PID *regulator*

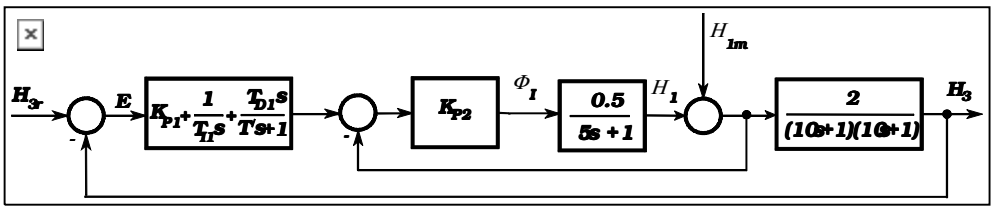
Krmiljenje z upoštevanjem merljive motnje

Iz simuliranih odzivov lahko opazimo, da regulacijski sistem odpravi motnjo v približno 80 *min.* Če je motnja merljiva, lahko precej izboljšamo kvaliteto regulacije. Uporabili smo shemo, ki jo prikazuje Sl. 7.86. Pri tem smo izbrali ojačenje krmilnika

$$K_{PZ} = 2$$

Kaskadna regulacija

Kvaliteto sledilnega in regulacijskega delovanja lahko izboljšamo s kaskadno regulacijo. Uporabimo shemo, ki jo prikazuje Sl. 7.96.



Sl. 7.96. Kaskadna regulacija hidravličnega sistema

Za pomožni regulator G_{P2} smo izbrali *P* regulator z ojačenjem

$$K_{P2} = 2$$

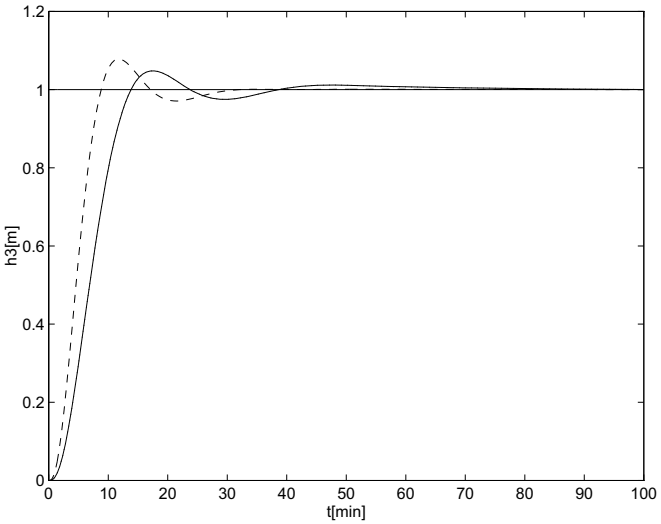
Pomožni regulator smo načrtali tako, da smo dosegli časovno konstanto pomožne regulacijske zanke 2.5, kar je dvakrat manj, kot je časovna konstanta dela procesa G_{P2}.

Glavni regulator G_{P1} smo določili z optimizacijo na enak način in ob enaki cenilki, kot prej optimalni enozančni regulator, le da je bila pri simulaciji že vključena pomožna regulacijska zanka. Dobili smo naslednje parametre:

$$K_{P1} = 4.62 \qquad T_{I1} = 20.6 \qquad T_{D1} = 5.52$$

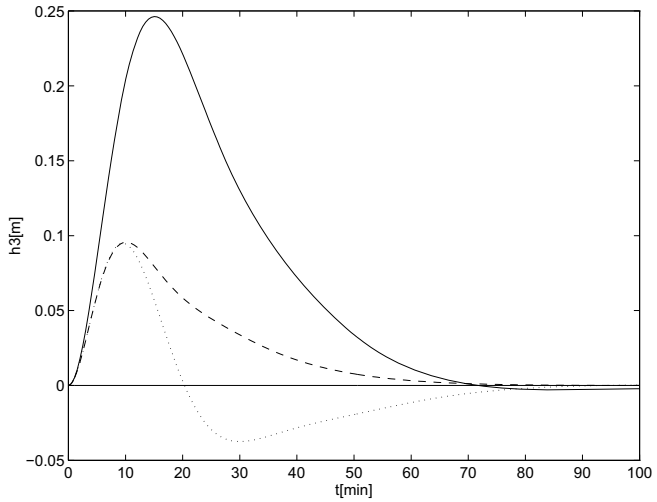
S temi parametri dosežemo cenilko ITAE = 23.4, kar je dvakrat manj kot pri enozančni regulaciji.

Sl. 7.97 prikazuje poteka nivoja tekočine pri optimalnem enozančnem in optimalnem kaskadnem regulatorju. Oba regulatorja sta bila optimirana za sledilno delovanje. Kaskadni regulator je nekoliko hitrejši.



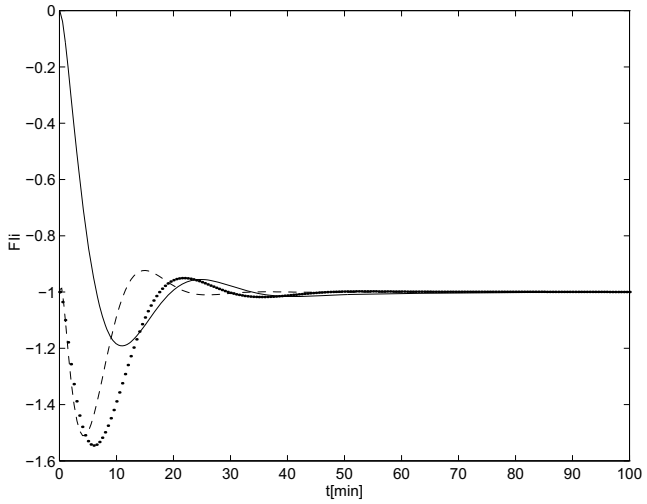
Sl. 7.97. Potek nivoja pri enozančni (__) in kaskadni (---) regulaciji

Še večjo izboljšavo pa predstavlja kaskadna regulacija pri regulacijskem delovanju. Sl. 7.98 prikazuje poteke nivoja pri stopničasti motnji H_{1m} = 0.5 pri optimalnem enozančnem regulatorju, pri krmiljenju z upoštevanjem motnje in pri optimalnem kaskadnem regulatorju. V vseh primerih je *PID* regulator optimiran za sledilno delovanje.



Sl. 7.98. Potek nivoja pri enozančni (__) regulaciji, pri vključenem krmiljenju (...) in pri kaskadni (---) regulaciji

Vidimo, da sta metoda z vključenim krmiljenjem in kaskadna regulacija približno enako uspešni in precej bolj učinkoviti od enozančne regulacije.



Sl. 7.99. Potek nivoja pri enozančni (__) regulaciji, pri vključenem krmiljenju (...) in pri kaskadni (---) regulaciji

Zanimiv je tudi prikaz regulirne veličine ϕ_l pri obravnavanih primerih. Simulirane poteke prikazuje Sl. 7.99.

Opazimo, da sta si signala v primeru krmiljenja in kaskadne regulacije zelo podobna, zato je tudi učinek podoben. V obeh primerih je začetna vrednost regulirne veličine enaka ustaljeni vrednosti -1, t.j. vrednosti, ki v ustaljenem stanju kompenzira motnjo.

Δ

7.3.3.3 Adaptivna regulacija

Adaptivna regulacija je tista regulacija, ki se prilagaja spremembam v svoji okolici. Pravzaprav vsaka regulacija upošteva te spremembe, saj povratna zanka izloča motnje in zmanjša vpliv spreminjajočih se parametrov. Čeprav še vedno ni enoveljavne definicije adaptivne regulacije, so strokovnjaki na tem področju soglasni, da ***je adaptivna regulacija tista regulacija, ki prilagaja parametre regulatorja spremembam v svoji okolici***.

Začetki adaptivne regulacije segajo v zgodnja petdeseta leta tega stoletja in na področje aeronavtike. Letala (predvsem visokozmogljiva) letijo namreč na različnih višinah in z različnimi hitrostmi, zato se parametri njihovega dinamičnega obnašanja zelo spreminjajo. Svoj razcvet so adaptivni sistemi doživeli v sedemdesetih letih, ko je razvoj tehnologije (mikroračunalniki) omogočil njihovo enostavno izvedbo. Področje njihove uporabe se je razširilo tudi na procesno industrijo, saj parametri tovrstnih procesov zelo zavisijo od delovne točke. Tipični primeri so nelinearni izvršni členi, kjer parametri lineariziranega procesa zavise od položaja izvršnega člena. Podoben primer je spreminjanje časovnih konstant s pretokom. Znano je, da je časovna konstanta koncentracije v neki posodi (reaktorju) odvisna od njene prostornine ter pretoka skozi njo (v prvem približku je kar njun kvocient). Pretok pa je odvisen od produkcije in se seveda spreminja. Za doseganje dobre kvalitete regulacije se morajo regulatorji takšnim spremembam prilagajati.

Poleg parametrov procesa pa se lahko spreminja tudi značaj motenj. Tipični primer je krmarjenje ladij, saj se lastnosti motenj (valovi, veter), to je njihov frekvenčni spekter, zelo spreminja.

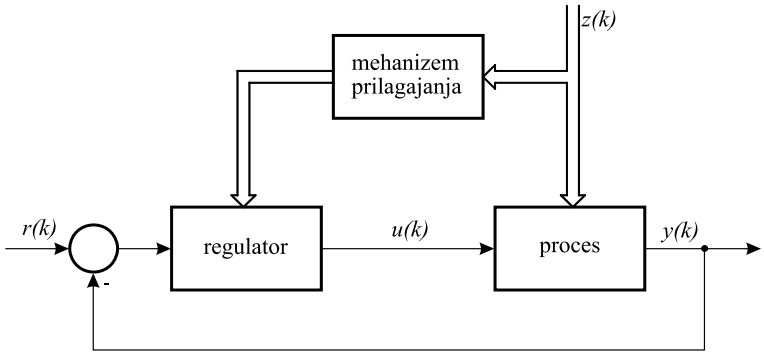
Ker adaptivni sistemi spreminjajo parametre regulatorjev, so v svoji naravi nelinearni sistemi. To otežuje njihovo matematično analizo, predvsem analizo stabilnosti. Zato je potrebno uporabiti metode za analizo nelinearnih sistemov, npr. Ljapunovo teorijo stabilnosti.

Kot smo že omenili, se adaptivni sistemi samodejno prilagajajo spremembam v svoji okolici. Te spremembe lahko nastanejo zaradi znanih (merljivih) vplivov, ali pa so posledica neznanih oz. nemerljivih vplivov. V prvem primeru uporabimo odprtozančni adaptivni sistem, v drugem pa zaprtozančnega.

Odprtozančni adaptivni sistemi

Odprtozančni adaptivni sistemi temeljijo na merjenju vplivov, ki spremene lastnosti procesa. Zato imenujemo tovrstno vodenje tudi *adaptivno krmiljenje*.

Princip tovrstnih adaptivnih sistemov prikazuje Sl. 7.100., kjer lahko vidimo, da spreminjamo regulator glede na merljivi vpliv *z(k)*. Tovrstni sistemi vsebujejo samo eno povratno zanko, t.j. povratno zanko osnovnega regulacijskega sistema. Sama adaptacija regulatorja pa se izvrši v odprti zanki. Primer odprtozančnega adaptivnega sistema je spreminjanje parametrov regulatorja letala glede na hitrost in višino leta. V procesni industriji, kjer, kot smo pokazali zgoraj, zavisijo časovne konstante od pretokov, lahko prilagajamo parametre regulatorja glede na spremembe pretoka.

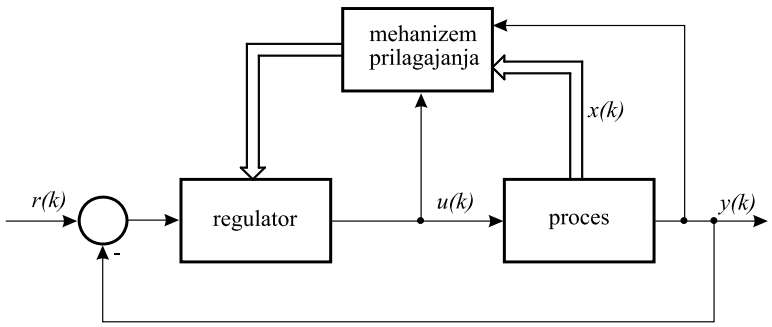


Sl. 7.100. Adaptivno krmiljenje.

Realizacije odprtozančnih adaptivnih sistemov so znane pod imenom *razporejanje parametrov* (angl. gain scheduling). V tem primeru načrtamo regulator za vse možne kombinacije vplivov in tako načrtane regulatorje shranimo v obliki tabele. Dejansko uporabljeni regulator je tisti, ki pripada trenutnim dejanskim vplivom. Dobra stran adaptivnega krmiljenja je v hitrem odzivu na spremembe parametrov procesa, ker je obnašanje procesa znano vnaprej. Druga prednost je v tem, da se pri njih ne srečamo s problemom stabilnosti adaptivne zanke, ker pač adaptivne povratne zanke ni (to seveda ne pomeni, da ne more postati nestabilna osnovna povratna zanka, če smo slabo načrtali regulator, oz. če uporabljeni regulator ne ustreza dejanskemu procesu). Slabost adaptivnega krmiljenja je v tem, da običajno ne moremo zajeti vseh vplivov in smo zato prisiljeni k zanemaritvam, kar negativno vpliva na kakovost vodenja.

Povratnozančni adaptivni sistemi

Če sprememb procesa ni možno določiti na osnovi opazovanja merljivih signalov oziroma, če je merjenje teh signalov predrago, moramo uporabiti povratnozančne adaptivne sisteme. Princip teh sistemov prikazuje Sl. 7.101.



Sl. 7.101. Povratnozančni adaptivni sistemi

Bistvo povratnozančnih adaptivnih sistemov je v zbiranju informacij o spremembah v procesu z opazovanjem vhodov in izhodov procesa (ter, če je izvedljivo, tudi z opazovanjem njegovih stanj). Na osnovi teh informacij se potem izvrši prilagoditev regulatorja. Poleg osnovne povratne zanke vsebujejo tovrstni regulacijski sistemi še drugo, t.i. adaptivno povratno zanko. Adaptivna povratna zanka, je običajno nelinearna.

Zbiranje informacij o spremembah v procesu je možno na več načinov, zato ločimo tudi več vrst povratnozančnih adaptivnih sistemov.

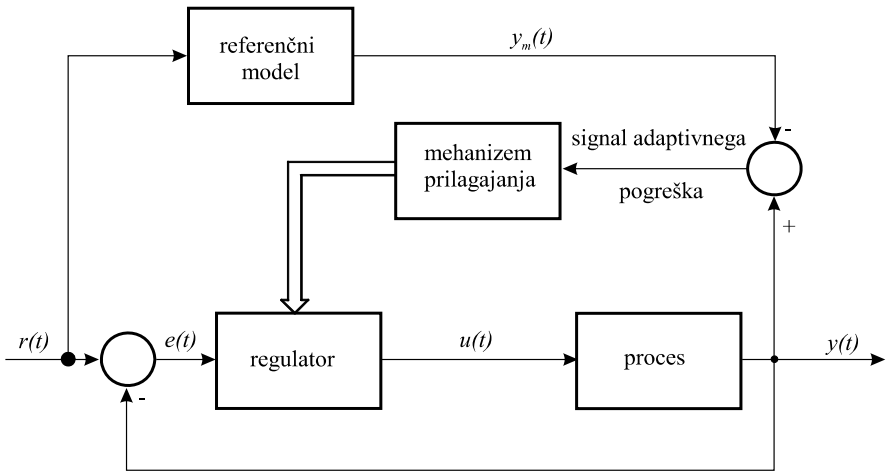
Dualni adaptivni regulacijski sistemi

Spremembe v procesu je z opazovanjem vhodnih in izhodnih signalov možno določiti le, če je proces dovolj vzbujan, torej če so opazovani signali frekvenčno dovolj bogati. Po drugi strani pa je naloga regulacije držati regulirano veličino čim bližje želeni vrednosti, ki je običajno (vsaj relativno dolg časovni interval) konstantna. Obe zahtevi sta si v nasprotju. Dualne adaptivne regulacijske sisteme načrtamo tako, da optimiramo kriterij, ki vsebuje ceno zbiranja informacij in kvalitete regulacije. Tako dobljeni sistemi omogočajo maksimalno možno zbiranje informacij ob optimalnem regulacijskem obnašanju. Pri svojem delovanju morajo upoštevati tako pretekle kakor tudi prihodnje signale. Ta naloga ni enostavna, zato so tovrstvi regulatorji doživeli le maloštevilne realizacije.

Nedualni adaptivni regulacijski sistemi

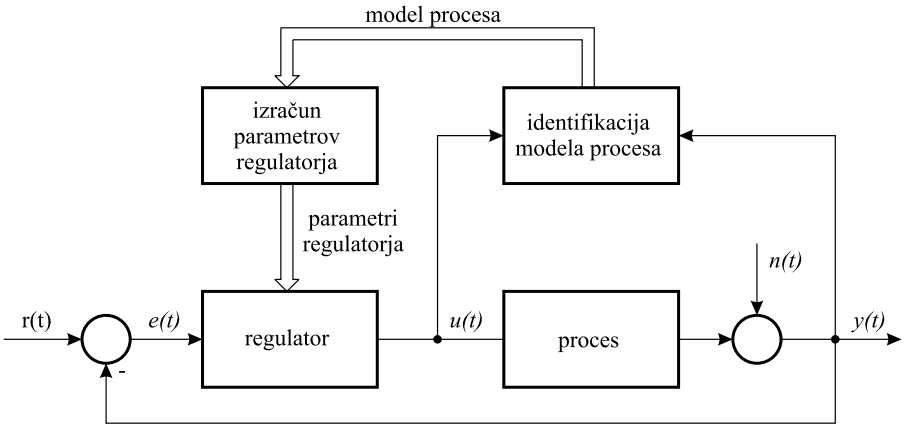
Tovrstni sistemi vršijo regulacijsko funkcijo le na osnovi trenutne ocene sprememb v procesu, torej na osnovi trenutnih in preteklih signalov. Pri tem pa sta prisotni dve filozofiji, tako da delimo nedualno adaptivne regulacijske sisteme na *modelno referenčne* in *modelno identifikacijske* adaptivne regulacijske sisteme.

- *Modelno referenčne* adaptivne regulacijske sisteme načrtamo tako, da predpišemo želeno zaprtozančno obnašanje sistema v obliki referenčnega modela. Regulator v osnovni regulacijski zanki spreminjamo tako, da dejanska regulirana veličina sledi izhodni veličini referenčnega modela. Princip modelno referenčnih adaptivnih regulacijskih sistemov prikazuje Sl. 7.102.



Sl. 7.102. Modelno referenčni adaptivni regulacijski sistemi.

- *Modelno identifikacijski* adaptivni regulacijski sistemi temeljijo na principih *ločitve* (separacije) in *nedvomne ekvivalence* (angl. certainty equivalence). Strukturo modelno identifikacijskih adaptivnih sistemov prikazuje Sl. 7.103.



Sl. 7.103. Modelno identifikacijski adaptivni regulacijski sistemi.

Glede na princip ločitve vsebuje adaptivna povratna zanka dve funkciji, in sicer identifikacijo in modifikacijo regulatorja. Glede na princip nedvomne ekvivalence predpostavimo, da je dejanski model procesa enak identificiranemu; regulator določimo torej na osnovi identificiranega modela.

Poznamo tudi več vrst modelno identifikacijskih adaptivnih regulacijskih sistemov, tu omenimo le *parametrsko adaptivne*, ki so se najbolj uveljavili v praksi. Tovrstni adaptivni regulacijski sistemi predpostavljajo nespremenljivo strukturo;

identifikacijski postopek se tako skrči na ocenjevanje parametrov, modifikacija regulatorja pa na spremembo njegovih parametrov.

V industrijski praksi se je adaptivno vodenje začelo uveljavljati sredi osemdesetih let tega stoletja, in sicer tako v obliki samonastavljivih regulatorjev, katerih osnovni namen je adaptivno vodenje, kakor tudi kot sestavni del velikih sistemov za procesno vodenje.

Danes vsebujejo že tudi standardni PID regulatorji minimalne funkcije adaptacije, npr. avtomatsko nastavitev parametrov pri zagonu. Tovrstna nastavitev (angl. autotune) je v veliko pomoč pri parametriranju regulatorjev, ki je predvsem pri procesih z velikimi časovnimi konstantami dolgotrajen postopek. Po uspešni nastavitvi parametrov regulatorja, ostanejo le-ti nespremenjeni; pravimo, da jih "zamrznemo".

∇

Primer 7.17: *Adaptivna regulacija hidravličnega sistema.*

Oglejmo si adaptivno regulacijo procesa prvega reda z neznanim ojačenjem (ki pa mora biti različno od 0) in z neznano časovno konstanto. Tipični primer takšnega sistema je hidravlični proces z enim shranjevalnikom (npr. zgornji shranjevalnik na Sl. 7.10, podpoglavje 7.2.1, opisan z enačbo 7.35).

$$\frac{H_1(s)}{\phi_I(s)} = \frac{Y(s)}{U(s)} = \frac{R_1}{R_1C_1s+1} \tag{7.128}$$

Diskretna prenosna funkcija takšnega procesa je

$$G_p(z) = \frac{Y(z)}{U(z)} = \frac{bz^{-1}}{1+az^{-1}} \tag{7.129}$$

kjer sta b in a neznana. Veljati mora $b \neq 0$. Teoretično sicer za a ni omejitev, vendar pa mora v praksi veljati $0<a<1$ (proces mora biti stabilen in mora imeti zvezni ekvivalent). $Y(z)$ je nivo tekočine v shranjevalniku, $U(z)$ pa dotok tekočine v shranjevalnik.

Načrtali bomo modelno identifikacijski adaptivni regulacijski sistem, ki je principielno podan na Sl. 7.103. Tak adaptivni sistem vsebuje dva postopka in sicer identifikacijo in pa nastavljanje regulatorja.

Identifikacija

Identifikacija diskretnih sistemov je v svojem bistvu, kot smo to spoznali že v podpoglavju o modeliranju, reševanje sistemov enačb. Časovni zapis procesa (7.129) je

$$y(k) = bu(k-1) - ay(k-1) \tag{7.130}$$

Če to enačbo zapišemo še za časovne trenutke k-1, k-2, …. k-n jih lahko združimo v matrični zapis, ki podaja znani predoločen sistem enačb

$$\mathbf{y} = \mathbf{\Psi} \cdot \vartheta \tag{7.131}$$

kjer je

$$\mathbf{y} = \begin{bmatrix} y(k) \\ y(k-1) \\ \dots\dots\dots\dots \\ y(k-n+1) \end{bmatrix} \quad \mathbf{\Psi} = \begin{bmatrix} u(k-1) & -y(k-1) \\ u(k-2) & -y(k-2) \\ \dots\dots & \dots\dots\dots\dots \\ u(k-n) & -y(k-n) \end{bmatrix} \quad \boldsymbol{\vartheta} = \begin{bmatrix} b \\ a \end{bmatrix} \tag{7.132}$$

Vektor $\boldsymbol{\vartheta}$ je vektor neznanih parametrov, ki jih lahko v vsakem trenutku izračunamo po formuli

$$\vartheta = \left(\mathbf{\Psi}^T \mathbf{\Psi}\right)^{-1} \mathbf{\Psi}^T \mathbf{y} \tag{7.133}$$

Pogoj za ta izračun je nesingularnost matrike $\mathbf{\Psi}$, ki vsebuje vhodne in izhodne signale procesa. Ti signali ne smejo biti identično enaki 0 in tudi ne smejo biti medsebojno linearno odvisni. Ta pogoj imenujemo pogoj stalnega vzbujanja.

Na ta način torej dobimo oceno neznanih parametrov *b* in *a*.

Nastavljanje regulatorja

Načrtali bomo regulator, ki izloča motnjo na izhodu hidravličnega sistema (npr. dodaten iztok vode iz rezervoarja) z vnaprej predpisano dinamiko; predpisali bomo torej pole oz. karakteristično enačbo zaprte zanke.

Izbrani regulator je regulator tipa PI s prenosno funkcijo

$$G_R(z) = \frac{q_o + q_1 z^{-1}}{1 - z^{-1}} \tag{7.134}$$

Prenosna funkcija, ki določa vpliv motnje na izhod procesa je

$$G_u(z) = \frac{Y(z)}{N(z)} = \frac{1}{1 + G_R(z) G_P(z)} = \frac{1}{1 + \frac{q_0 + q_1 z^{-1}}{1 - z^{-1}} \frac{b z^{-1}}{1 + a z^{-1}}} = \frac{(1 - z^{-1})(1 + a z^{-1})}{1 + (a - 1 + b q_0) z^{-1} + (b q_1 - a) z^{-2}} \tag{7.135}$$

Karakteristični polinom zaprte zanke je

$$D(z^{-1}) = 1 + (a - 1 + b q_0) z^{-1} + (b q_1 - a) z^{-2} \tag{7.136}$$

Želimo, da je ta polinom, ki opisuje dinamiko izločanja motnje konstanten, torej

$$D(z^{-1}) = 1 + \alpha_1 z^{-1} + \alpha_2 z^{-2} \tag{7.137}$$

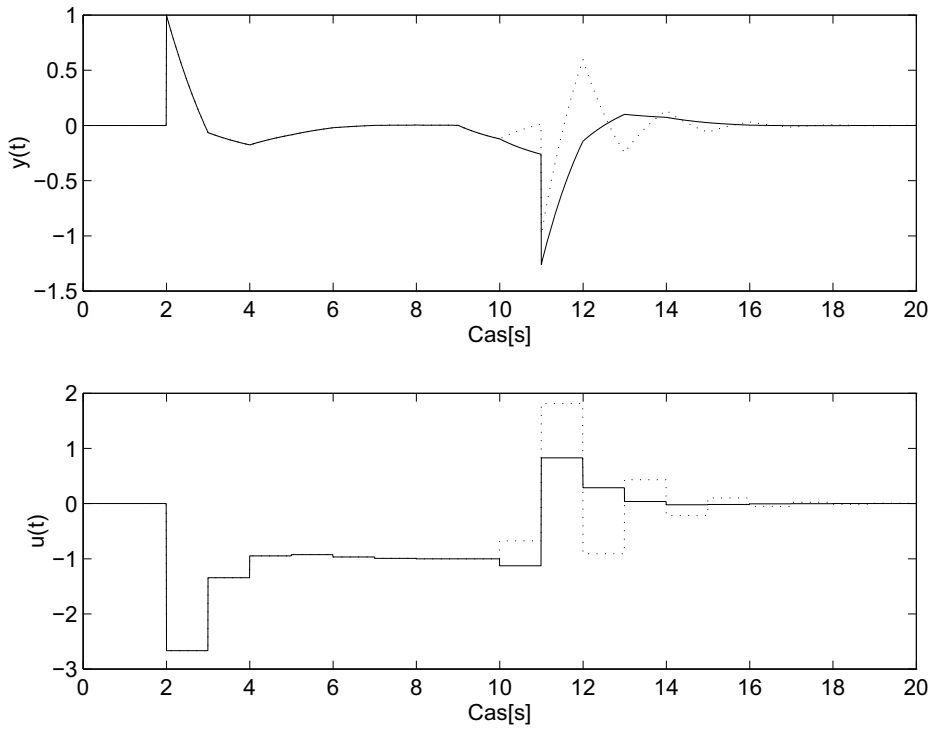
kjer sta konstanti α_1 in α_2 izbrani tako, da zagotavljata robustno in hitro izločanje motenj.

Primerjava enačb (7.136) in (7.137) nudi zakon za nastavljanje parametrov regulatorja

$$q_o = \frac{\alpha_1 - a + 1}{b}$$
$$q_1 = \frac{\alpha_2 + a}{b} \tag{7.138}$$

S tem smo zaključili izpeljavo regulacijskega zakona. Adaptivni sistem deluje tako, da z identifikacijo oceni parametre procesa, postopek nastavljanja parametrov regulatorja pa z upoštevanjem le-teh nastavi regulator tako, da ostane dinamika za izločanje motnje nespremenjena za kakršnekoli vrednosti b in a. V praksi je seveda potreben še nadzor ocenjenih parametrov, kajti lahko se zgodi, da pri prehodnih pojavih le-ti zavzamejo nesmiselne vrednosti, kar lahko privede do nestabilnosti adaptivnega sistema.

Sl. 7.104 prikazuje odziv fiksnega (prekinjena črta) in adaptivnega (neprekinjena črta) regulatorja na motnjo.



Sl. 7.104. Odziv fiksnega (prekinjena črta) in adaptivnega (neprekinjena črta) regulatorja na motnjo

Fiksni regulator je uglašen na nominalne parametre procesa (ojačenje 1 in časovna konstanta 1,96 s, kar ustreza diskretnim parametrom b=0.4, a=-0.6 pri času vzorčenja 1s). Parametri procesa se po 9 s spremenijo (ojačenje 1.25, časovna konstanta 1.5 s, kar ustreza diskretnim parametrom b=0.6, a=-0.5134).

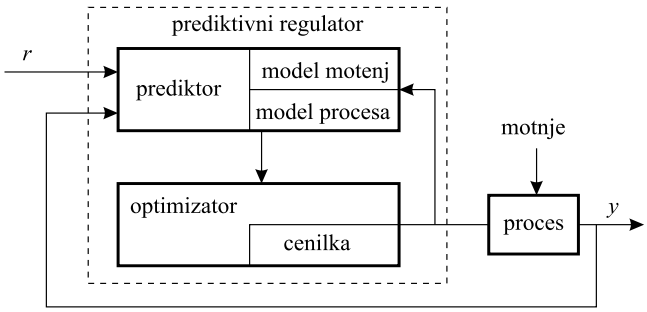
Odziv na motnjo, ki nastopi v 2. sekundi, je zato enak v obeh primerih. Odziv na drugo motnjo pri 11. sekundi pa se pri fiksnem regulatorju poslabša, adaptivni regulator pa se spremembi procesa prilagodi.

Δ

7.3.3.4 Prediktivna regulacija

Prediktivna regulacija je podvrsta t.i. regulacije na osnovi modela (angl.: model based control). Pravzaprav mora vsako vodenje upoštevati model procesa, vendar le-ta običajno v regulatorju ni eksplicitno izražen.

Princip prediktivne regulacije prikazuje Sl. 7.105.



Sl. 7.105. Princip prediktivnega vodenja

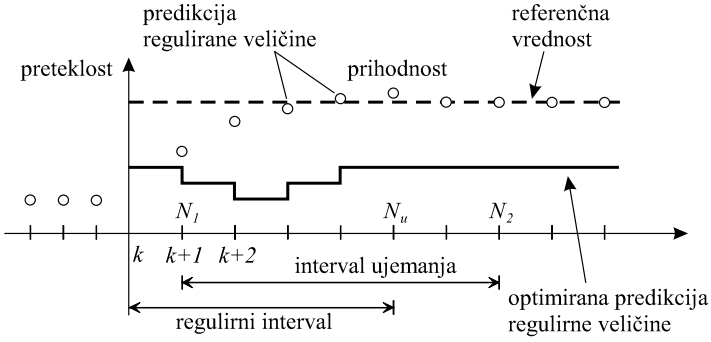
Prediktivna regulacija temelji na predpostavljenem modelu procesa, predpostavljenem modelu motenj in predpostavljenem scenariju želenih (referenčnih) vrednosti. Na osnovi teh predpostavk določimo bodoče regulirne signale tako, da optimiramo izbran kriterij-cenilko. Predpostavimo tudi, da vodimo proces s časovno diskretnim regulatorjem tako, da predstavimo bodoče regulirne signale z množico vrednosti regulirnega signala v bodočih trenutkih vzorčenja.

Princip delovanja prediktivnega regulatorja je naslednji: optimizator izračuna optimalno vrednost regulirnega signala s pomočjo prediktorja, ki izračunava bodoče vrednosti reguliranega signala na osnovi modela procesa, modela motenj, trenutne vrednosti izhodnega signala in želenih bodočih vrednosti signala.

Značilen potek signalov lahko vidimo na Sl. 7.106.

Tipična cenilka upošteva odstopanje napovedi reguliranega signala od želene reference na izbranem časovnem intervalu, ki ga imenujemo interval ujemanja (N_2-N_1). Ta interval se običajno ne prične takoj, temveč šele v bližnji prihodnosti, saj bo trenutna vrednost regulirnega signala vplivala na regulirani signal šele v prihodnosti (N_1). Poleg odstopanja reguliranega signala od želene reference pa upošteva cenilka še stroške, ki so potrebni za sledenje referenčni vrednosti, izražajo pa se z odstopanji regulirnega signala

od njegove ustaljene vrednosti. Tudi ti stroški se izračunavajo na izbranem časovnem obzorju, ki pa se prične takoj, imenujemo pa ga regulirni interval (0-N_u).



Sl. 7.106. Tipični poteki signalov pri prediktivnem vodenju

Poleg teh odstopanj lahko cenilka upošteva še razne omejitve, kot so omejitve reguliranega (npr. prevzpon) oz. regulirnega signala (npr. končni hod izvršnih členov). Ko so znane optimalne vrednosti trenutnega in prihodnjih regulirnih signalov, uporabimo trenutni regulirni signal kot aktivni trenutni regulirni signal na procesu, vse ostale pa zavržemo. V naslednjem času vzorčenja celotni postopek ponovimo, in sicer tako, da ostane relativna lega obeh intervalov (obzorij) glede na trenutni čas nespremenjena. Obe obzorji se tako pomikata, zato ju imenujemo tudi pomični obzorji.

Znanih je več metod prediktivnega vodenja. Razlika med njimi je predvsem v uporabljenem modelu procesa oz. motenj in upoštevanju omejitev. Prva praktično uporabna metoda je Model Algoritmic Control (MAC), ki uporablja model procesa v obliki končnega impulznega zaporedja. Druga znana metoda je Dynamic Matrix Control (DMC), ki uporablja model procesa v obliki končnega odziva na stopnico. Obe metodi uporabljata model motenj v obliki stopnice. Znane metode na osnovi neparametričnih modelov so še Interval Model Control (IMC) in Qudratic DMC, ki upošteva omejitve in je danes največkrat praktično uporabljena metoda predvsem v petrokemijski industriji.

Prediktivni regulatorji, ki temeljijo na parametričnih modelih procesa in motenj (prenosni funkciji oz. spektra motenj) so znani pod imenom Model Based Predictive Control (MBPC). Najbolj znana metoda je Generalized Predictive Control (GPC).

Metode, ki uporabljajo linearni model procesa brez omejitev, lahko rešijo optimizacijski problem analitično. V tem primeru ni potreben iterativni postopek optimizacije, ampak lahko izrazimo zaporedje regulirnih signalov kot funkcijo procesnih parametrov in preteklih vrednosti reguliranega in regulirnega signala.

Pri nelinearnih procesih in splošnih omejitvah pa je potrebna realizacija v originalni obliki, ki je prikazana na Sl. 7.105. V zadnjem času se kot modeli uporabljajo mehki modeli in pa nevronske mreže (glej podpoglavje 7.4). Možne so tudi kombinacije obeh metod, tako da nelinearni model sprotno lineariziramo (oz. sprotno izračunavamo

njegov impulzni odziv oz. odziv na stopnico) ter nato uporabimo analitično izračunane regulirne zakone.

∇

Primer 7.18: *Prediktivna regulacija hidravličnega sistema.*

Oglejmo si prediktivno vodenje procesa prvega reda z ojačenjem 1 in s časovno konstanto 1.96 s oz. z diskretno prenosno funkcijo (pri času vzorčenja 1 s)

$$G_p(z) = \frac{Y(z)}{U(z)} = \frac{0.4z^{-1}}{1-0.6z^{-1}} \tag{7.139}$$

Uporabimo torej enak model kot pri adaptivni regulaciji (podpoglavje 7.3.3.4). Če pomnožimo to prenosno funkcijo z operatorjem $\Delta = 1 - z^{-1}$ v števcu in imenovalcu, dobimo

$$(1-0.6z^{-1})(1-z^{-1})Y(z) = 0.4\Delta U(z) \tag{7.140}$$

oz. v časovnem prostoru

$$y(k) = 1.6y(k-1) - 0.6y(k-2) + 0.4\Delta u(k-1) \tag{7.141}$$

Izbrano cenilko, ki zajema interval ujemanja med eno in štirimi sekundami ter regulirni interval do dveh sekund lahko zapišemo v naslednji obliki

$$J = \sum_{j=1}^{3}[y(k+j) - r(k+j)]^2 + \lambda\sum_{j=0}^{1}[\Delta u(k+j)]^2 \tag{7.142}$$

kjer je λ parameter, ki uravnava razmerje med kvaliteto vodenja in za to potrebnimi stroški.

Iz enačbe (7.141) sledijo predikcije reguliranega signala

$$\begin{aligned}
\hat{y}(k+1) &= 1.6y(k) - 0.6y(k-1) + 0.4\Delta u(k) \\
\hat{y}(k+2) &= 1.6\hat{y}(k+1) - 0.6y(k) + 0.4\Delta u(k+1) = \\
&= 1.96y(k) - 0.96y(k-1) + 0.4\Delta u(k+1) + 0.64\Delta u(k) \\
\hat{y}(k+3) &= 1.6\hat{y}(k+2) - 0.6\hat{y}(k+1) + 0.4\Delta u(k+2) = \\
&= 2.1760y(k) - 1.1760y(k-1) + 0.4\Delta u(k+2) + 0.64\Delta u(k+1) + 0.784\Delta u(k)
\end{aligned} \tag{7.143}$$

oziroma v matrični obliki ob upoštevanju dejstva, da se regulirna veličina po intervalu vpliva ne spreminja več $(\Delta u(k+2) = 0)$

$$\begin{bmatrix} \hat{y}(k+1) \\ \hat{y}(k+2) \\ \hat{y}(k+3) \end{bmatrix} = \begin{bmatrix} 0.4 & 0 \\ 0.64 & 0.4 \\ 0.7840 & 0.64 \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \end{bmatrix} + \begin{bmatrix} 1.6 & -0.6 \\ 1.96 & -0.96 \\ 2.1760 & -1.1760 \end{bmatrix} \begin{bmatrix} y(k) \\ y(k-1) \end{bmatrix} \tag{7.144}$$

Skrajšani zapis te enačbe je

$$\mathbf{y}^{+} = \mathbf{G}\Delta\mathbf{u}^{+} + \mathbf{F}\mathbf{y}^{-} \tag{7.145}$$

kjer oznake + pomenijo, da gre za signale v prihodnosti (ki so neznani) in oznaka -, da gre za pretekle (znane) vrednosti signalov. Če bi imeli sistem višjega reda, bi se na desni strani zgornje enačbe pojavil še izraz $\mathbf{\Gamma}\Delta\mathbf{u}^{-}$, ki ga bomo zaradi popolnosti pri izpeljavi regulacijske zakonitosti upoštevali. V našem primeru velja $\mathbf{\Gamma}$=0.

Cenilko (7.142) lahko zapišemo v obliki

$$\begin{aligned} J &= (\mathbf{y}^{+} - \mathbf{r}^{+})^{T}(\mathbf{y}^{+} - \mathbf{r}^{+}) + \lambda\Delta\mathbf{u}^{+T}\Delta\mathbf{u}^{+} = \\ &= \left[\mathbf{G}\Delta\mathbf{u}^{+} + \mathbf{\Gamma}\Delta\mathbf{u}^{-} + \mathbf{F}\mathbf{y}^{-} - \mathbf{r}^{+}\right]^{T} \cdot \left[\mathbf{G}\Delta\mathbf{u}^{+} + \mathbf{\Gamma}\Delta\mathbf{u}^{-} + \mathbf{F}\mathbf{y}^{-} - \mathbf{r}^{+}\right] + \lambda\Delta\mathbf{u}^{+T}\Delta\mathbf{u}^{+} \end{aligned} \tag{7.146}$$

kjer vektor $\mathbf{r}^{+}$ zajema tri prihodnje vrednosti referenčnega signala. Poiščimo vektor prihodnjih vrednosti regulirnega signala, ki minimizira cenilko, tako da odvajamo cenilko po regulirnem signalu:

$$\frac{\partial J}{\partial\Delta\mathbf{u}^{+}} = 2\mathbf{G}^{T}\mathbf{G}\Delta\mathbf{u}^{+} + 2\mathbf{G}^{T}\left[\mathbf{\Gamma}\Delta\mathbf{u}^{-} + \mathbf{F}\mathbf{y}^{-} - \mathbf{r}^{+}\right] + 2\lambda\Delta\mathbf{u}^{+} = 0 \tag{7.147}$$

iz česar sledi

$$\Delta\mathbf{u}^{+} = \left[\mathbf{G}^{T}\mathbf{G} + \lambda\mathbf{I}\right]^{-1}\mathbf{G}^{T}\left[\mathbf{r}^{+} - \mathbf{\Gamma}\Delta\mathbf{u}^{-} - \mathbf{F}\mathbf{y}^{-}\right] \tag{7.148}$$

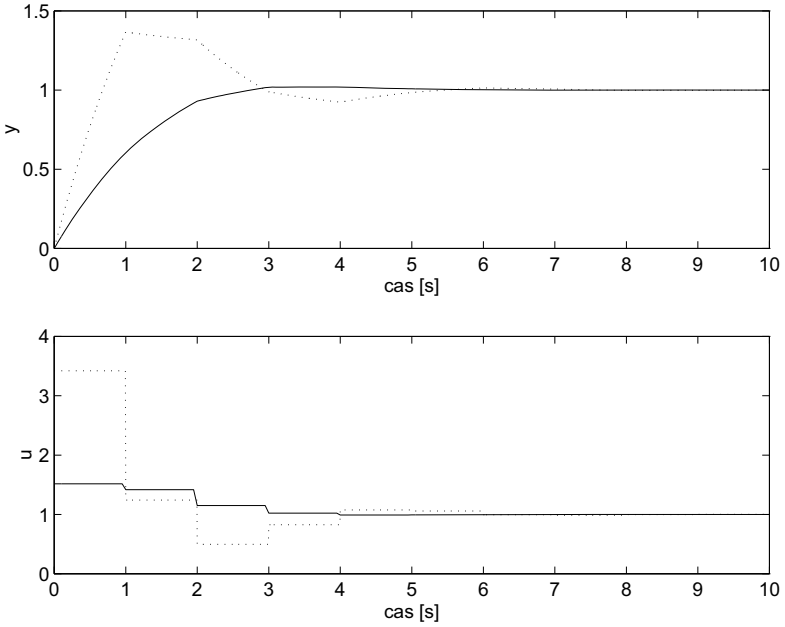
Po vstavitvi matrik **G** in **F** (iz enačbe (7.144) ter $\mathbf{\Gamma}$ (v našem primeru 0) in izbranem λ=0.1 lahko zapišemo prvo vrstico enačbe (7.148) v naslednji obliki.

$$\Delta u(k) = -2.6648y(k) + 1.1485y(k-1) + 0.9374r(k+1) + 0.4390r(k+2) + 0.1400r(k+3)$$

Ta enačba predstavlja regulirni zakon. Opazimo lahko, da ta zakon zahteva poznavanje prihodnjih vrednosti referenčnega signala. Če le-teh ne poznamo, jih v praksi zamenjamo z njegovo trenutno vrednostjo. V tem primeru sledi izhod optimalno zakasnjenemu referenčnemu signalu.

Sl. 7.107 prikazuje regulirni in regulirani signal sistema s PID (prekinjena črta) in s prediktivnim regulatorjem (neprekinjena črta) pri stopničasti spremembi želenega nivoja v shranjevalniku (regulirana signala-*y* in regulirna signala-*u*). PID regulator je bil načrtan tako, da ima zaprtozančni sistem približno dvakrat krajšo časovno konstanto kot hidravlični proces. Prediktivni regulator smo spremenili v kavzalnega, tako da smo predpostavili *r(k+1)= r(k+2) = r(k+3)=r(k)*

Vidimo lahko, da ima prediktivni regulator bolj umirjen regulirni signal in manj prenihanja.



Sl. 7.107. Odziv predikcijskega in PID regulatorja.

Δ

7.3.3.5 Multivariabilna regulacija

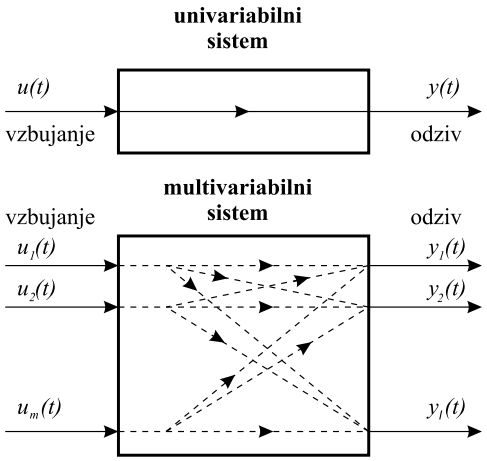
Pregled literature s področja vodenja prav hitro pokaže, da je precejšen delež posvečen tudi problematiki procesov, katerih značilnost je, da hkrati vplivamo na več vhodnih veličin zato, da bi dosegli želeno obnašanje večih izbranih izhodnih veličin. Omenjena literatura tudi dokazuje, da so se številne, nemalokrat zahtevne tehnike načrtovanja, ki obravnavajo takšne multivariabilne sisteme, izkazale kot neobhodne tudi v praktičnih aplikacijah za zadostitev kompleksnejših ciljev načrtovanja. Zgodi pa se, da je določeno kvaliteto vodenja mogoče doseči tudi pri multivariabilnih sistemih s pomočjo tehnik, ki so poznane za procese z enim vhodom in enim izhodom. Na osnovi takšnega razmišljanja si seveda kaj hitro zastavimo vprašanja, kot so:

- Kdaj lahko pri sistemih multivariabilne narave, to je pri sistemih, kjer si želimo, da bi več med seboj povezanih izhodnih veličin izkazovalo predpisano obnašanje, uporabimo univariabilno vodenje ali več univariabilnih regulatorjev v posameznih povratnih zankah?
- Na kakšne težave moramo biti pri uporabi takšnega vodenja pripravljeni?
- Kaj pridobimo, če sploh kaj, če namesto tako imenovanih klasičnih regulatorjev,

uporabimo multivariabilne?

Osnovne značilnosti multivariabilnih sistemov

Da bi osvetlili odgovore na zastavljena vprašanja, prikažimo najprej bistvene razlike med univariabilnimi in multivariabilnimi sistemi s stališča, ki je pomembno za načrtovanje vodenja. Razmere so ilustrirane na Sl. 7.108.



Sl. 7.108. Bistvena razlika med univariabilnimi in multivariabilnimi sistemi se kaže v številu obravnavanih vhodnih in izhodnih spremenljivk

Medtem ko je značilnost univariabilnih sistemov ta, da je izhod sistema, ki ga označimo z *y(t)*, odvisen od samo enega vhoda *u(t)*, pa je pri multivariabilnem sistemu vsak od izhodnih signalov $y_1(t)$, $y_2(t)$, … , $y_\ell(t)$ lahko odvisen od vsakega od vhodnih signalov $u_1(t)$, $u_2(t)$, … , $u_m(t)$ ali z drugimi besedami, vsak od vhodnih signalov v splošnem lahko povzroči spreminjanje vsakega od izhodov. Tako običajno v bolj kompaktni obliki zapisa multivariabilnega sistema predstavimo vhodne oz. izhodne signale s pripadajočima vektorjema $\mathbf{y}(t)$ in $\mathbf{u}(t)$.

Nadaljnja razlika, ki je posledica razlike med številom vhodnih in izhodnih spremenljivk, se kaže tudi v najpogosteje uporabljanih oblikah zapisov sistemov. Če se omejimo na linearne, časovno nespremenljive sisteme s konstantnimi parametri, potem znana oblika zapisa modela sistema s prenosno funkcijo za univariabilni sistem v primeru multivariabilnega sistema dobi obliko:

$$\mathbf{y}(s) = \mathbf{G}(s)\mathbf{u}(s) \tag{7.149}$$

oziroma:

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \\ \vdots \\ Y_\ell(s) \end{bmatrix} = \begin{bmatrix} G_{1,1}(s) & G_{1,2}(s) & \cdots & G_{1,m}(s) \\ G_{2,1}(s) & G_{2,2}(s) & \cdots & G_{2,m}(s) \\ \vdots & \vdots & \cdots & \vdots \\ G_{\ell,1}(s) & G_{\ell,2}(s) & \cdots & G_{\ell,m}(s) \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \\ \vdots \\ U_m(s) \end{bmatrix} \tag{7.150}$$

Prenosno funkcijo *G(s)* je v tem primeru nadomestila matrika prenosnih funkcij **G**(s) katere elementi so prenosne funkcije $G_{i,j}(s)$, ki opisujejo razmerje med *i*-tim izhodnim in *j*-tim vhodnim signalom. Če so vsi elementi matrike prenosnih funkcij obravnavanega sistema različni od nič, potem to seveda pomeni, da vsak od vhodov sistema vpliva na vsakega od izhodov oz., da je vsak izhod odvisen od vseh vhodnih signalov. Če je kateri od elementov matrike prenosnih funkcij enak nič, potem pripadajoča povezava med določenim vhodom in določenim izhodom ne obstaja. Prav poseben primer predstavlja situacija, ko so od nič različni le elementi na glavni diagonali matrike prenosnih funkcij. Če zaradi enostavnosti predpostavimo, da je število vhodov enako številu izhodov (m = ℓ), potem lahko takšen sistem opišemo z:

$$\begin{bmatrix} Y_1(s) \\ Y_2(s) \\ \vdots \\ Y_\ell(s) \end{bmatrix} = \begin{bmatrix} G_{1,1}(s) & 0 & \cdots & 0 \\ 0 & G_{2,2}(s) & \cdots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & G_{\ell,m}(s) \end{bmatrix} \begin{bmatrix} U_1(s) \\ U_2(s) \\ \vdots \\ U_m(s) \end{bmatrix} \tag{7.151}$$

ali

$$\begin{aligned} Y_1(s) &= G_{1,1}(s)U_1(s) \\ Y_2(s) &= G_{2,2}(s)U_2(s) \\ &\vdots \\ Y_m(s) &= G_{m,m}(s)U_m(s) \end{aligned} \tag{7.152}$$

Situacija, kot jo opisuje podana skupina enačb, pomeni, da je vsak od izhodov sistema odvisen le od enega samega vhoda, oziroma da vsak vhodni signal vpliva le na en sam izhod procesa. Takšen multivariabilen sistem, ki ga zaradi pripadajoče matrike prenosnih funkcij imenujemo tudi diagonalen oz. neinteraktiven, predstavlja s stališča načrtovanja direkten ekvivalent univariabilnim sistemom. Številne tehnike načrtovanja vodenja multivariabilnih sistemov se tako skušajo v prvem koraku približati tej idealni situaciji, medtem ko lahko poteka načrtovanje v drugem koraku za vsak podsistem ločeno.

V primeru, ko izvendiagonalni elementi matrike prenosnih funkcij niso identično enaki nič, pa zapiranje posamezne povratne vezave v splošnem spremeni razmere tudi v preostalem delu sistema. Vplivi se seveda razširjajo preko tako imenovanih križnih povezav oz. prenosnih funkcij na izvendiagonalnih mestih. Od tod pa seveda sledi, da pri načrtovanju vodenja multivariabilnih sistemov neupoštevanje križnih povezav lahko pripelje tudi do nezaželenega obnašanja zaprtozančnega sistema. Vidimo torej, da so

ravno križne povezave tiste, ki onemogočajo direktno uporabo tehnik za univariabilne sisteme na multivariabilnih primerih. Poudariti je potrebno, da celo takrat, ko se zavestno odločimo za uporabo univariabilnih regulatorjev, nikakor ne smemo pozabiti na obstoj križnih povezav.

Omenimo še eno obliko zapisa sistema, tj. zapis v prostoru stanj. V primeru multivariabilnega sistema z m - vhodi in ℓ - izhodi ima le-ta obliko:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \end{aligned} \tag{7.153}$$

kjer je $\mathbf{x}$(t) vektor spremenljivk stanja dimenzije $n \; x \; 1$, matrike $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ in $\mathbf{D}$ pa so konstantne matrike dimenzij $n \; x \; n$, $n \; x \; m$, $x \; n$ in $\ell \; x \; m$. Povezavo med zapisom v prostoru stanj in matriko prenosnih funkcij lahko opišemo z:

$$\mathbf{G}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D} \tag{7.154}$$

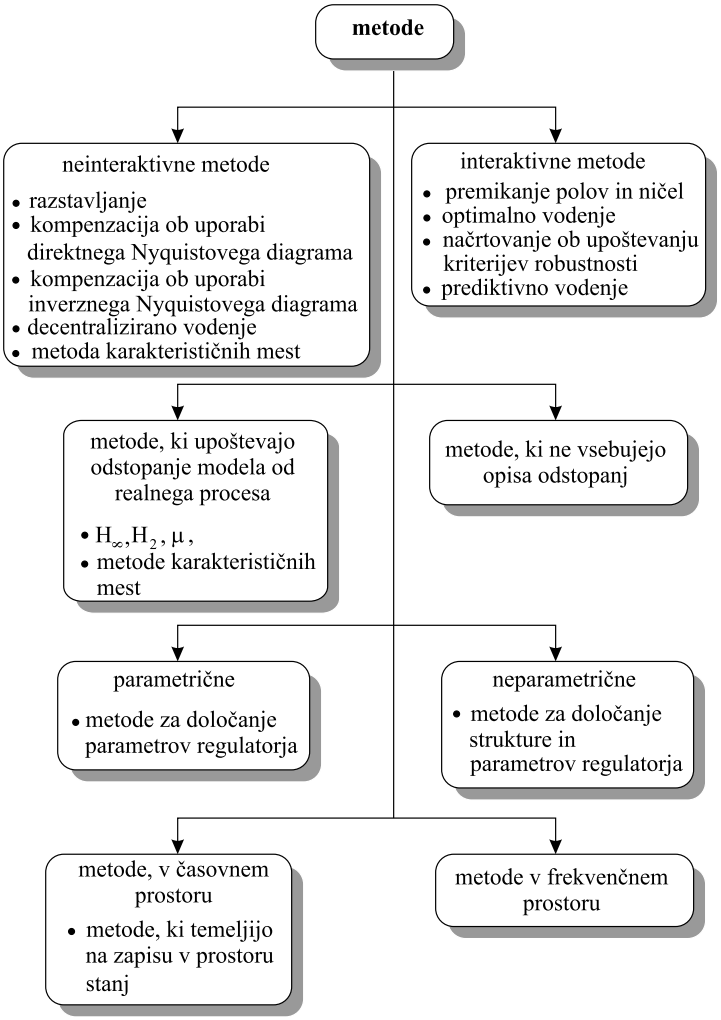
Če poznamo zapis sistema v prostoru stanj, potem izračun pripadajoče matrike prenosnih funkcij oz. prenosne funkcije ni problematičen, medtem ko obratna preslikava ni enolična in je v primeru multivariabilnega sistema lahko tudi dokaj problematična.

Pregled metod multivariabilne regulacije

V zadnjih desetletjih je bilo razvito veliko število metod za načrtovanje vodenja multivariabilnih sistemov. Čeprav je bil v začetku razvoj posvečen pretežno univariabilnim sistemom, pa lahko danes trdimo, da so v glavnem vse metode posplošene tudi na multivariabilne sisteme. Poleg tega pa multivariabilno vodenje zaradi svoje specifičnosti pozna tudi nekaj njemu lastnih pristopov.

Razvoj metod načrtovanja vodenja multivariabilnih sistemov se je pričel z idejami popolnega izničenja (Wolovich, 1974, Atanasijević, 1984) oz. zmanjševanja (Rosenbrock, 1979, Milanović, 1983) vpliva križnih povezav, tako da je v nadaljevanju mogoče uporabiti tehnike za univariabilne procese na posameznih podsistemih, ki imajo pri tako preoblikovanem multivariabilnem sistemu dominantno vlogo. Tovrstne pristope zaradi omenjenih razlogov uvrščamo v tako imenovane neinteraktivne pristope za razliko od interaktivnih (Karba, 1981), ki v načrtovanje vključujejo multivariabilni sistem kot celoto, pri čemer zmanjševanje križnih povezav ni vključeno med cilje načrtovanja. Tipični cilji načrtovanja so pri tovrstnih pristopih poleg stabilnosti še želena kvaliteta zaprtozančnega obnašanja pri različnih vplivih, ki utegnejo motiti delovanje procesa.

V zadnjih letih pa postajajo vse pomembnejši tudi aspekti robustnega načrtovanja, ki omogoča ohranjanje lastnosti stabilnosti oz. lastnosti obnašanja zaprtozančnega sistema ob upoštevanju razkoraka med delovanjem realnega procesa in idealizirano situacijo, ki je povzeta z opisom modela sistema (Patel in Munro, 1982, Maciejowski, 1989, Kocijan, 1995).



Sl. 7.109. Delitev metod načrtovanja vodenja multivariabilnih sistemov

Prav oblika modela tudi lahko predstavlja enega od možnih pokazateljev za delitev metod. Tako lahko ločimo med algoritmi, ki omogočajo načrtovanje vodenja linernih oz. nelinearnih procesov, procesov z mrtvim časom oz. brez njega, procesov s koncentriranimi ali porazdeljenimi parametri. Pri tem lahko izračuni potekajo v časovnem ali frekvenčnem (zveznem oz. diskretnem) prostoru.

Nadaljnji korak v razvoju obravnavane metodologije pa predstavljajo možnosti uporabe metod umetne inteligence, ki odpirajo pot tudi pristopom, kjer poznavanje

matematičnega modela procesa ni nujno potrebno. Obstoj velikega števila najraznovrstnejših postopkov načrtovanja pa je nadalje odprl tudi možnosti kombiniranja le-teh, in sicer tako kombiniranja med samimi pristopi načrtovanja kot tudi prehajanje med predstavitvami v časovnem in frekvenčnem prostoru, ki omogoča učinkovitejšo obravnavo doseženih lastnosti zaprtozančnega obnašanja in s tem tudi učinkovitejše načrtovanje.

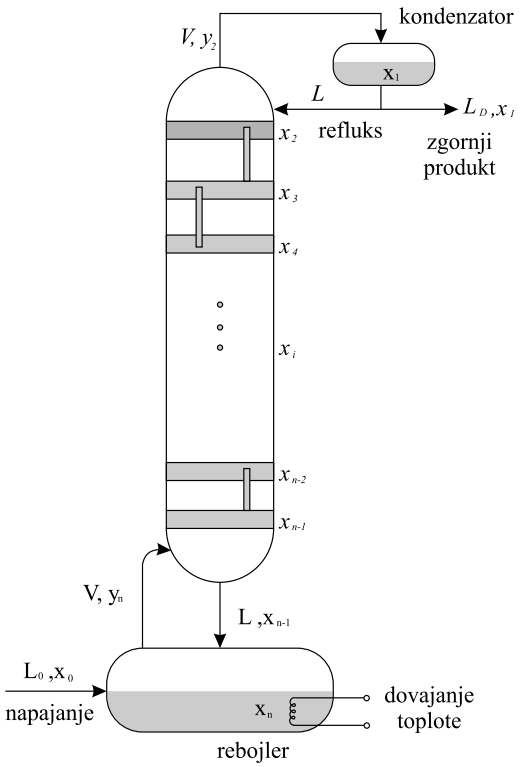
Zelo groba klasifikacija metod je podana na Sl. 7.109.

∇

Primer 7.19: *Regulacija multivariabilnega procesa destilacije*

Kot primer multivariabilnega sistema bomo obravnavali proces destilacije, ki predstavlja enega najpogostejših kemijskih postopkov, s katerim se srečamo v procesni industriji (Matko in ostali, 1992). Pri tem želimo ločevati pomešane komponente na osnovi različne hlapnosti. Obstaja seveda veliko možnih načinov delovanja destilacijskih kolon, kar je predvsem odvisno od vrste in količine predelovane snovi. Eno od možnosti shematično prikazuje Sl. 7.110, kjer gre za destilacijo dvokomponentne zmesi metanol - voda (Atanasijević, 1984).

Zmes dovajamo v rebojler na dnu kolone s sestavo x_0 in pretokom L_0, ki je enak odtoku L_D na vrhu kolone, tako da je v napravi konstantna količina snovi. Rebojler ogrevamo in posledica tega je, da hlapi potujejo po koloni navzgor s pretokom *V(t)*. Ker hlapi laže hlapna komponenta, ki je v našem primeru metanol, hitreje kot teže hlapna, se sestava po prekatih x_i za laže hlapno komponento po koloni navzgor viša. Na vrhu kolone mora doseči predpisano oz. želeno vrednost. Hlape na vrhu kolone v kondenzatorju ohlajamo. Del tako dobljenega produkta predstavlja končni izdelek, del pa vodimo nazaj v kolono z namenom, da vzdržujemo predpisano sestavo destilata. Pri tako urejeni napravi smo torej kot vhodni veličini v sistem izbrali refluksni pretok *L(t)* in tok hlapov *V(t)* na katerega vplivamo posredno s količino dovedene toplote, izhodni veličini pa sta sestava laže hlapne komponente na prvem prekatu ($y_1(t) = x_2(t)$ na Sl. 7.110) in na zadnjem prekatu v koloni ($y_2(t) = x_{n-1}(t)$ na Sl. 7.110). Pri tako definiranem delovanju smo določili linearizirani in poenostavljeni model sistema v obliki, kot jo podajata naslednji enačbi:



Sl. 7.110. Semišaržna destilacijska kolona

$$
\begin{aligned}
\dot{\mathbf{x}}(t) &= \begin{bmatrix} -0.43524 & 0.43822 & 0.01722 & -0.01941 \\ -0.12286 & 0.12105 & -0.00923 & 0.01039 \\ -0.19811 & 0.19309 & -0.34307 & 0.35350 \\ 0.10171 & -0.09697 & -0.16981 & 0.16108 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0.12589 & -0.09741 \\ 0.11821 & -0.08017 \\ 0.29233 & -0.11676 \\ 0.21985 & -0.09320 \end{bmatrix} \mathbf{u}(t) \\
\mathbf{y}(t) &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \mathbf{u}(t)
\end{aligned}
\tag{7.155}
$$

Poudariti velja, da poenostavitev v tem primeru pomeni znižanje reda sistema, ki smo jo izvedli s pomočjo identifikacije. Posledica je, da predstavlja poenostavljeni model ekvivalent dejanski situaciji s stališča vhodno - izhodnega obnašanja, medtem ko stanja modela sedaj ne predstavljajo dejanskih stanj procesa.

Iz zapisa v prostoru stanj lahko dobimo ekvivalenten zapis z matriko prenosnih funkcij:

$$\mathbf{G}(s) = \frac{1}{(s+3.122\,10^{-2})(s+2.110^{-3})(s+0.231 \pm j4.72\,10^{-2})} *$$

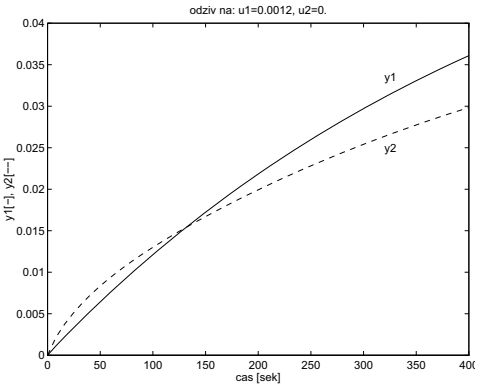
$$* \begin{bmatrix} 0.125\,(s+0.199)(s+3.1110^{-2})(s+0.25) & -9.74\,10^{-2}(s+3.1110^{-2})(s+0.196 \pm j6.36\,10^{-2}) \\ 0.292(s+0.155)(s+1.34\,10^{-2})(s+0.024) & -0.116(s+0.193 \pm j0.115)(s+1.62\,10^{-2}) \end{bmatrix}$$

(7.156)

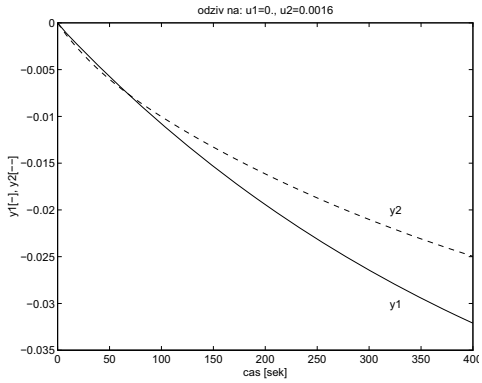
Iz podanega opisa je razvidno, da vsebuje sistem tudi tako imenovane križne interakcije. Včasih tudi pravimo, da je sistem interaktiven glede na oba vhoda. V našem primeru to pomeni, da spreminjanje refluksnega pretoka vpliva tako na sestavo zmesi na vrhu, kot tudi na dnu kolone, prav tako pa ima spreminjanje toka hlapov vpliv na sestavo zmesi po vsej koloni in s tem tudi na oba izbrana izhoda.

O prisotnosti križnih povezav se lahko prepričamo tudi s pomočjo simulacije. Sl. 7.111 prikazuje odziva sistema pri stopničasti spremembi amplitude 0.0012 prvega vhoda, pri čemer je drugi vhod enak 0, medtem ko prikazuje Sl. 7.112 odziva v primeru, ko smo izvedli stopnico amplitude 0.0016 na drugem vhodu, pri čemer je prvi vhod enak 0.

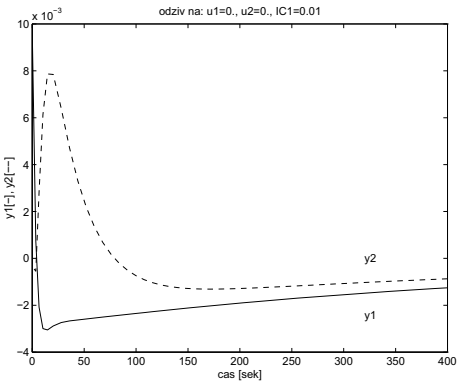
Poleg vhodno - izhodne interaktivnosti se v primeru multivariabilnih sistemov srečamo tudi s pojmom interaktivnosti glede na izhode sistema. To pomeni, da odziv na začetno stanje na določenem izhodu povzroči spreminjanje tudi na ostalih izhodih sistema. Tovrstne interakcije ilustrirata Sl. 7.114 in Sl. 7.113 za začetno stanje 0.01 na prvem oz. na drugem izhodu procesa.



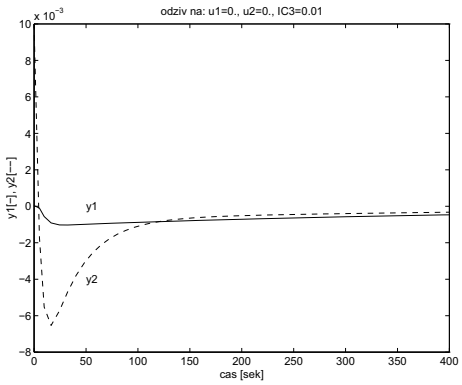
Sl. 7.111. Odziv procesa na prvi vhod. (interakcija je $y_1(t)$)



Sl. 7.112. Odziv procesa na drugi vhod. (interakcija je $y_1(t)$)



Sl. 7.114. Odziv procesa na začetno stanje na prvem izhodu. (interakcija $y_2(t)$)



Sl. 7.113. Odziv procesa na začetno stanje na drugem izhodu. (interakcija je $y_1(t)$)

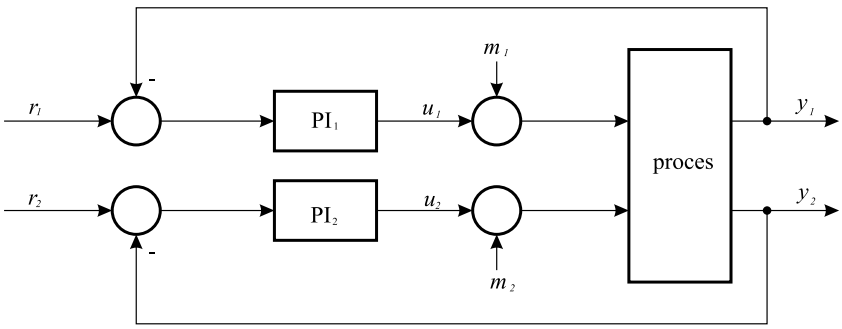
Primerjava učinkov različnih načinov vodenja

V nadaljevanju bi želeli podati primerjavo različnih učinkov nekaterih najpogosteje uporabljanih regulacijskih struktur.

Predpostavimo najprej razmere, kot so ilustrirane na Sl. 7.115.

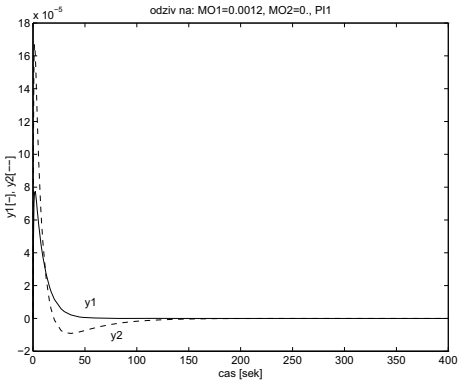
V tem primeru reguliramo proces z dvema univariabilnima proporcionalno - integrirnima regulatorjema, delovanje procesa pa je moteno z aditivnima motnjama (m_1 in m_2) na obeh vhodih v proces. Pri tako definirani zaprtozančni strukturi nadalje predpostavimo, da bomo v prvi fazi študirali samo vpliv regulatorja v prvi od obeh zank, medtem ko bomo drugi regulator začasno izklopili. Nastavimo torej ojačenja regulatorjev na naslednje vrednosti:

$$K_{P1} = 13.33 \quad K_{P2} = 0$$
$$K_{I1} = 1.333 \quad K_{I2} = 0$$

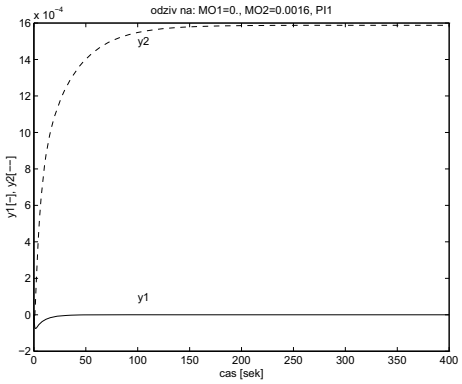


Sl. 7.115. Univariabilno vodenje procesa.

Preverimo delovanje tako definiranega zaprtozančnega sistema. Razmere ilustrirata Sl. 7.116 in Sl. 7.117. Na Sl. 7.116 sta prikazana oba odziva sistema za primer, ko smo realizirali stopničasto motnjo amplitude 0.0012 na prvem vhodu, medtem ko je motnja na drugem vhodu enaka nič, Sl. 7.117 pa prikazuje razmere, če deluje stopničasta motnja amplitude 0.0016 na drugem vhodu pri nemotenem prvem vhodu. Vidimo, da je realizacija regulatorja v prvi zanki zagotovila stabilno obnašanje sistema in popolno izničenje motnje, če motnja deluje v delu sistema, kjer je realiziran regulator. Če pa je motnja nastopila v drugi zanki, je regulator v prvi zanki uspel izregulirati vpliv te motnje na nič, medtem ko drugi izhod sistema izkazuje precejšen pogrešek v ustaljenem stanju.



Sl. 7.116. Odziv procesa na motnjo na prvem vhodu. (PI - regulator v prvi zanki)



Sl. 7.117. Odziv procesa na motnjo na drugem vhodu. (PI - regulator v prvi zanki)

Sedaj preverimo razmere, ko regulator uporabimo le v drugi zanki. Ojačenja obeh regulatorjev tako nastavimo na naslednje vrednosti:

$$K_{P1} = 0 \quad K_{P2} = -0.10$$
$$K_{I1} = 0 \quad K_{I2} = -0.01$$

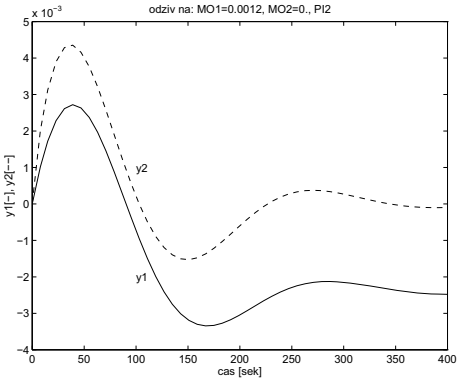
Razmere so ilustrirane na Sl. 7.118 in Sl. 7.119 za enak način vzbujanja kot v prejšnjem primeru. Vidimo, da smo ponovno zagotovili stabilno obnašanje zaprtozančnega sistema, le da se sedaj občuten pogrešek v ustaljenem stanju pojavi na prvem izhodu v primeru, ko je proces moten v zgornji zanki.

Uporabimo regulatorja, ki sta relativno uspešno delovala vsak posebej, sedaj sočasno. To pomeni, da nastavimo parametre uporabljane regulacijske strukture kot:

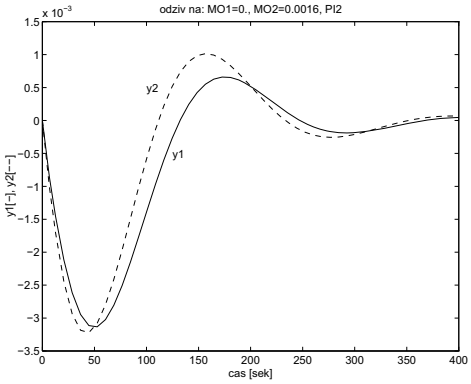
$$K_{P1} = 13.33 \quad K_{P2} = -0.10$$
$$K_{I1} = 1.333 \quad K_{I2} = -0.01$$

Delovanje tako definiranega zaprtozančnega sistema pri stopničastih motnjah na prvem oz. drugem vhodu ilustrirata Sl. 7.120 in Sl. 7.121. Vidimo, da je zaprtozančni sistem

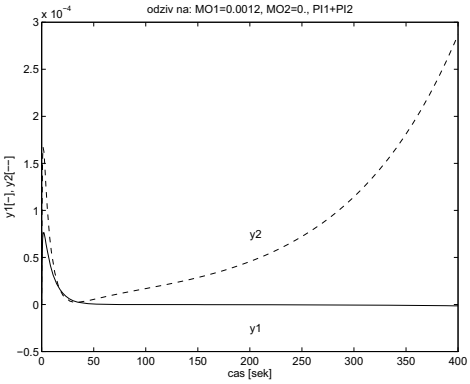
nestabilen, čeprav je bilo delovanje zaprtozančnega sistema pri vsakem regulatorju posebej stabilno. Vzrok je seveda treba iskati v obstoju križnih povezav multivariabilnega sistema, ki so povzročile nasprotujoče si delovanje regulirnih signalov oz. pozitivne povratnozančne vplive. Četudi bi se odločili za uporabo prikazanih univariabilnih regulatorjev, bi bilo potrebno enega od obstoječih ponovno uglasiti ob že realizirani drugi zaprti zanki. Problem, ki se pojavi pri takšni situaciji, in za katerega v splošnem rešitev ni poznana, je izbira zanke, ki jo uglašujemo najprej, oz. vrstni red zapiranja posameznih zank. Uglaševanje izbrane zanke je, kot je razvidno tudi iz prikazanega primera, močno odvisno od trenutnega stanja. Optimalno pot načrtovanja v takšnih primerih običajno določimo s poskušanjem, kar pa je lahko tudi dokaj zamudno, posebej v primerih, ko ima sistem večje število vhodnih in izhodnih spremenljivk.



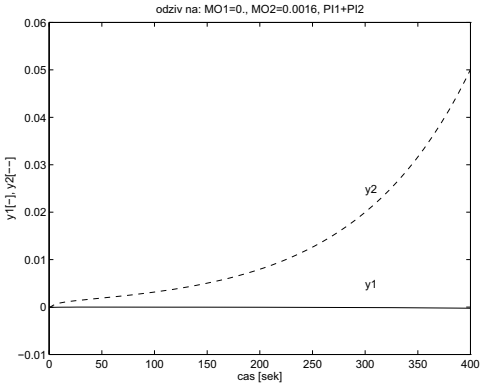
Sl. 7.119. Odziv procesa na motnjo na prvem vhodu. (PI - regulator v drugi zanki)



Sl. 7.118. Odziv procesa na motnjo na drugem vhodu. (PI - regulator v drugi zanki)

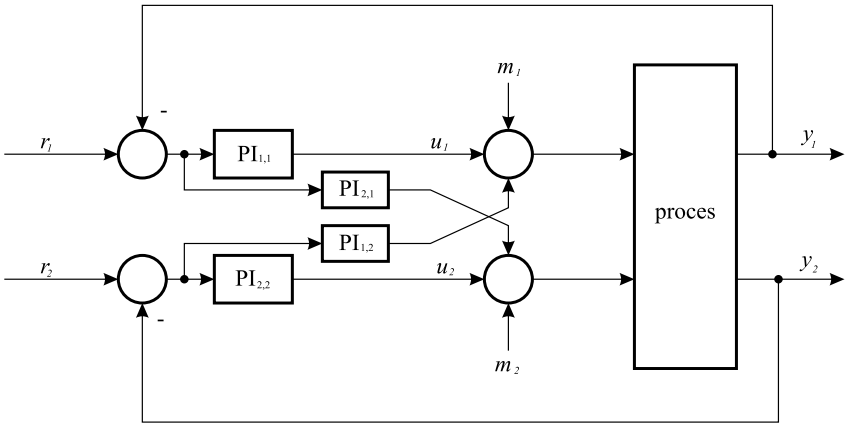


Sl. 7.121. Odziv procesa na motnjo na prvem vhodu. (hkratno delovanje obeh univariabilnih PI regulatorjev)



Sl. 7.120. Odziv procesa na motnjo na drugem vhodu. (hkratno delovanje obeh univariabilnih PI regulatorjev)

Sedaj pa prikažimo možnost, ko namesto dveh univariabilnih regulatorjev uporabimo polni multivariabilni proporcionalno - integrirni regulator, kot prikazuje Sl. 7.122.

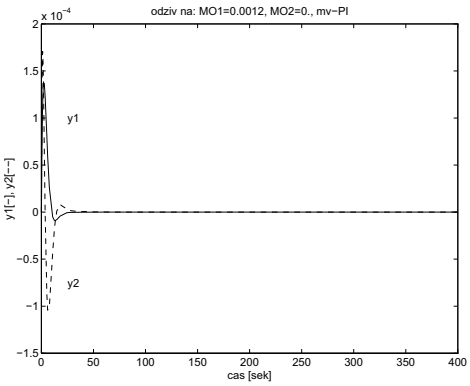


Sl. 7.122. Multivariabilno vodenje procesa.

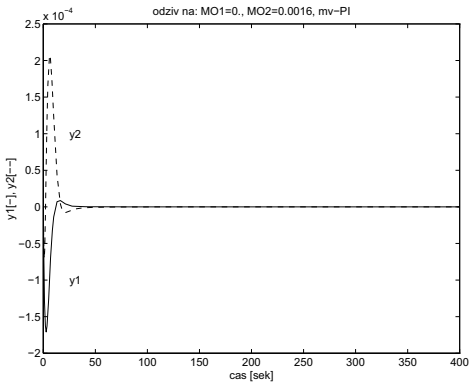
Parametre prikazane regulacijske strukture smo načrtali po metodi premikanja polov (Karba,1981) kot:

$$\mathbf{K_P} = \begin{bmatrix} 1.381 & 5.874 \\ -1.609 & 5.842 \end{bmatrix}, \quad \mathbf{K_I} = \begin{bmatrix} 2.297 & 1.590 \\ 0.0384 & 1.471 \end{bmatrix}$$

Preverimo učinke tako definiranega regulatorja. Razmere so ilustrirane na Sl. 7.123 za primer stopničaste motnje na prvem vhodu in na Sl. 7.124 za primer stopničaste motnje na drugem vhodu. Že na prvi pogled je očitno, da smo poleg stabilnosti zagotovili tudi zelo dobre regulacijske učinke za obe regulirani veličini.



Sl. 7.123. Odziv procesa na motnjo na prvem vhodu. (uporaba multivariabilnega PI regulatorja)



Sl. 7.124. Odziv procesa na motnjo na drugem vhodu. (uporaba multivariabilnega PI regulatorja)

Na osnovi podanih rezultatov zaprtozančnega delovanja obravnavane destilacijske kolone lahko zaključimo naslednje.

1. Izkoriščanje le dela spremenljivk sistema za namene vodenja (uporaba enega samega univariabilnega regulatorja na sicer multivariabilnem sistemu) je lahko upravičena oz. sprejemljiva, če so izpolnjeni naslednji pogoji:

   - obravnavani proces je stabilen, zahtevana kvaliteta zaprtozančnega obnašanja pa je naravnana pretežno na en izhod procesa. To pomeni, da lahko brez večjih dvomov določimo ustrezno mesto realizirane zaprte zanke;
   - pričakujemo, da bodo na sistem vplivale motnje takšnega značaja, da jih bo mogoče zadovoljivo izregulirati samo z enim regulirnim signalom. Tipično to pomeni, da bo motnja delovala v najbližji okolici regulacijske zanke. V nasprotnem primeru moramo pričakovati, da bo zaradi obstoja križnih povezav v sistemu prišlo do razširjanja motnje po celotnem procesu, kar se bo odražalo na vseh izhodih sistema. Čeprav tega do sedaj nismo posebej poudarjali, je prav, da omenimo tudi problem omejenosti regulirnih signalov. Razmisliti moramo ali je uporabljeni aktuator dovolj zmogljiv glede na značaj pričakovanih motenj.

   V kolikor katerikoli od naštetih pogojev ni izpolnjen, je vsekakor smiselna oziroma neobhodno potrebna razširitev regulacijske strukture.

2. Hkratna uporaba večih univariabilnih regulatorjev na multivariabilnem sistemu zahteva naslednje razmisleke:

   - najprej moramo ustrezno izbrati pare vhodno - izhodnih spremenljivk. Ta izbira je v splošnem realizirana tako, da določenemu izhodnemu signalu pripišemo tisti vhodni signal, ki nanj najmočneje vpliva. Lahko se seveda primeri, da so omenjeni učinki podobni, in v takšni situaciji običajno izbiramo na osnovi poznavanja lastnosti procesa (kot npr. v obravnavanem primeru, glede na bližino vhodno - izhodnih spremenljivk);
   - sledi izbira vrstnega reda zapiranja posameznih zank, kjer ponovno nimamo zagotovila, katera od možnih kombinacij bo dala najučinkovitejšo zaprtozančno strukturo;
   - nadalje pa je potrebno zaradi prisotnosti križnih povezav pri uglaševanju vsakega zančnega regulatorja upoštevati učinke že načrtanih regulatorjev. V nasprotnem primeru lahko pride do nasprotujočih si učinkov posameznih regulatorjev, kar lahko povzroči nestabilno in s tem nesprejemljivo obnašanje zaprtozančnega sistema.

   Takšno načrtovanje regulacijske strukture v primerjavi s prej omenjano možnostjo navadno povzroči dvig kvalitete obnašanja celotnega procesa pa tudi manjše zahteve, ki se nanašajo na posamezno regulirno spremenljivko, saj se regulirna moč razporedi med vse uporabljene signale.

3. Čeprav o algoritmih načrtovanja multivariabilnih sistemov nismo natančneje govorili (natančnejši opis številnih možnosti je mogoče najti v citirani literaturi) pa je prav, da omenimo nekaj najsplošnejših lastnosti v primerjavi s prej omenjanima možnostima:

   - metode načrtovanja multivariabilnih sistemov, ki obravnavajo tovrstne procese kot celoto in rezultirajo tudi v multivariabilnih regulacijskih strukturah, običajno ne zahtevajo optimale izbire vhodno - izhodnih parov spremenljivk. Upoštevanje tovrstnih lastnosti je vsebovano v ustrezni nastavitvi parametrov določene strukture;
   - stabilnost navadno ni zahtevana lastnost. Večina algoritmov zelo dobro obvladuje tudi nestabilne procese;
   - tipična situacija je tudi ugodnejša raporeditev regulacijske moči in sicer tudi v primerjavi z uporabo večih univariabilnih regulatorjev. V splošnem namreč lahko pričakujemo, da bo pri približno enaki doseženi kvaliteti vodenja amplitudno območje regulirnih signalov manjše, če bomo uporabljali polni multivariabilni regulator. Posledica tega je, da pri uporabi multivariabilnega vodenja lahko pričakujemo višjo dosegljivo kvaliteto obnašanja;
   - zagotovo drži, da je načrtovanje multivariabilnih regulatorjev nekoliko kompleksnejša naloga, pa tudi cena realizacije je lahko zaradi kompleksnosti višja.

7.3.4 Hibridno vodenje

Pri hibridnem vodenju upoštevamo dvojno naravo večine tehniških sistemov. Po eni strani v sistemu, npr. proizvodnem procesu, tečejo številni fizikalni procesi, ki jih opisujejo zvezne fizikalne veličine, po drugi strani pa v marsikaterem procesu najdemo tudi diskretna stanja in prehode med njimi. ***Sistem, v katerem nastopajo tako zvezne kot diskretne spremenljivke, njihov potek pa opisujejo enačbe, ki so načeloma odvisne od obeh vrst spremenljivk, imenujemo hibridni sistem. Hibridno vodenje pa je vodenje takšnega sistema, pri čemer vodenje vključuje tako zvezno vodenje kot diskretno logično odločanje.***

Stanje hibridnega sistema vključuje tako diskretne kot zvezne spremenljivke. Razvoj stanja sistema pa sledi enačbam, ki so odvisne tako od zveznih, kot od diskretnih spremenljivk. V sistemu se tako prepleta zvezna in diskretna dinamika. Zvezno dinamiko običajno opisujejo diferencialne enačbe, v primeru diskretno vzorčenih sistemov pa diferenčne enačbe. Diskretno dinamiko lahko opišemo na različne načine, vedno pa gre za nek sistem diskretnih stanj in prehodov med njimi, npr. končni avtomat ali Petrijevo mrežo. Zvezna in diskretna dinamika vplivata druga na drugo ob nastopu določenih dogodkov. To so t.i. dogodki stanja, ko neka zvezna veličina doseže določeno vrednost. Dogodek lahko povzroči spremembo diskretnega stanja, ta pa lahko vpliva na zvezno dinamiko.

Zvezni opis je v večini primerov ustreznejši, kadar opisujemo fizikalno dogajanje v nekem procesu. Na najnižjem nivoju vodenja industrijskega procesa tako običajno teče več zveznih regulacijskih zank, ki so nadgrajene s sistemom odločanja, ki deluje diskretno. Značilen primer je robotska roka, kjer je potrebno zvezno regulirati pomik, hitrost in prijemalno silo, medtem ko je zaporedje prijemanj, pomikov in spuščanj, s kakršnim robot izvede zahtevano delovno operacijo, povsem diskreten proces.

Tudi pri značilnih zveznih proizvodnih procesih, kot npr. proizvodnja naftnih derivatov, nastopajo elementi diskretnega vodenja. To je predvsem v fazi zagona, ko se morajo podprocesi pognati v ustreznem vrstem redu, in še več, nekateri deli procesa se lahko poženejo šele, ko so doseženi ustrezni pogoji (npr. ustrezna temperatura, tlak) v drugih podprocesih. Podobno je pri zaustavitvi proizvodnje. Takšne zagonske in zaustavitvene sekvence pogosto izvajajo izurjeni operaterji, vendar pa je s stališča varnosti in zanesljivosti bolje, če tudi to nalogo prevzame sistem avtomatskega vodenja.

Takorekoč vsi sodobni sistemi vodenja vsebujejo tako zvezno zaprtozančno vodenje kot diskretno odločitveno logiko, ki določa način delovanja regulacijskega sistema. S hibridnim vodenjem se tako srečujemo povsod tam, kjer imamo sistem vodenja organiziran v hierarhično strukturo, hibridne lastnosti moramo upoštevati na stiku nivojev vodenja, ki delujejo večinoma zvezno, in nivojev, ki so po naravi diskretni. To je predvsem na nadzornem nivoju, medtem ko na višjih nivojih, npr. nivoju planiranja in vodenja proizvodnje, prevladuje povsem diskreten pogled na proizvodni proces. Primeri hibridnih sistemov so ekspertni sistemi, ki nadzorujejo in nastavljajo parametre klasičnih regulatorjev, sistemi planiranja, ki nastavljajo želene vrednosti lokalnih regulatorjev, nadzorni sistemi, ki preklapljajo med več regulacijskimi algoritmi in odločajo, kateri od njih bo uporabljen za vodenje procesa v določenem območju, itd. Možna pa so še druga področja uporabe hibridnega vodenja, npr.:

- vodenje šaržnih sistemov;
- vodenje močno nelinearnih zveznih sistemov;
- vodenje sistemov s spremenljivo strukturo;
- vodenje avtonomnih sistemov.

7.3.4.1 Osnovni pristopi k hibridnim sistemom

Hibridne sisteme torej sestavlja kombinacija zveznega sistema in sistema diskretnih dogodkov, ki vplivata drug na drugega. Za zvezni proces to pomeni nenadne spremembe v zvezni dinamiki, ki lahko nastopijo zaradi prehodov diskretnih stanj. Takšna sprememba je lahko npr. preklop dvopoložajnega ventila iz zaprtega v odprto stanje.

Zvezni proces pa vpliva na sistem diskretnih dogodkov z generiranjem diskretnih dogodkov. Ti lahko nastanejo ob spremembi izhodnega signala diskretnega odjemnika, npr dvopoložajnega nivojskega stikala. Ko nivo doseže določeno vrednost, stikalo preklopi, prehod stanja stikala pa predstavlja diskretni dogodek.

Izhodišče za načrtovanje vodenja takšnega sistema je izgradnja modela hibridnega sistema. Pri tem ločujemo med tremi osnovnimi pristopi:

- diskretne lastnosti sistema poskušamo zajeti v zveznem modelu;
- zvezne lastnosti sistema poskušamo vključiti v diskretni model;
- zvezni in diskretni del modeliramo ločeno, njun medsebojni vpliv pa zajamemo z vmesnikom med njima.

Pri prvem pristopu lahko diskretne lastnosti sistema zajamemo v obliki nelinearnosti (običajno nezveznih) in sistem modeliramo kot nelinearni zvezni sistem. Značilen primer je dvopoložajna regulacija zveznega procesa. Možne so tudi razširitve zveznih modelov s preklopnimi elementi, takšni so npr. preklopni Bond grafi. Prednost tega pristopa je v uporabi obstoječih teoretičnih orodij, slabost pa so omejene možnosti predstavitve diskretnega obnašanja.

Pri vključevanju zveznih lastnosti v diskretni model se najpogosteje uporabljajo različne razširitve Petrijevih mrež. Vsem je skupno to, da poleg diskretnih prehodov med stanji vpeljejo tudi različne oblike zveznih prehodov. Dobljeni model lahko do neke mere analiziramo podobno kot čisti diskretni model, vendar pa so z vpeljavo zveznih prehodov možnosti analize zmanjšane. V to skupino spadajo tudi hibridni avtomati, to je razširitev diskretnih avtomatov, kjer vsakemu diskretnemu stanju priredimo tudi nabor zveznih spremenljivk in pripadajočih enačb. Ko se spremeni stanje hibridnega avtomata, se spremeni tudi značaj zveznega dela sistema.

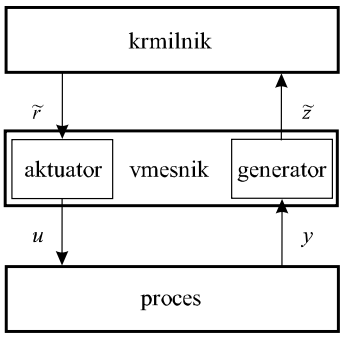
Pri tretjem pristopu uporabimo za modeliranje zveznega in diskretnega dela sistema običajne zapise. Za zvezni del so to diferencialne ali diferenčne enačbe, za diskretni del pa končni avtomati ali Petrijeve mreže. Vmesnik pretvarja simbolični izhod podsistema diskretnih dogodkov v ustrezne krmilne in regulirne signale ter tvori dogodke stanj, ki nastopajo ob določenih vrednostih zveznih veličin. Prednost takšnega pristopa je v tem, da ohranimo vso opisno moč in možnosti analize obstoječih pristopov k modeliranju. Seveda pa lahko neposredno analiziramo le tisti del dogajanja v sistemu, ki je omejen na zvezni ali diskretni podsistem. V nadaljevanju bomo ta pristop k modeliranju opisali še nekoliko podrobneje.

7.3.4.2 Kombinirani model hibridnega sistema

Osnovno idejo kombiniranega modela povzemamo po (Antsaklis in soavtorja, 1993). Sistem vodenja sestavljajo trije nivoji, kot prikazuje Sl. 7.125.

Krmilnik je predstavljen kot sistem diskretnih dogodkov, ki sprejema, obdeluje in daje na izhod dogodke, ki so predstavljeni s simboli. Proces je zvezni sistem, ki je opisan z diferencialnimi ali diferenčnimi enačbami. Proces sprejema, pretvarja in daje na izhod spremenljivke, ki so vsaj odsekoma zvezne. Krmilnik in proces komunicirata preko vmesnika, ki pretvarja vrednosti izhodnih spremenljivk procesa v simbole na vhodu krmilnika in simbole na izhodu krmilnika v zvezne vhodne signale procesa. Lahko si

predstavljamo, da je vmesnik sestavljen iz dveh delov: generatorja, ki spremlja izhode procesa in generira dogodke stanj ter aktuatorja, ki pretvarja simbolične ukaze krmilnika v odsekoma konstantne vhode procesa.



Sl. 7.125. Model hibridnega sistema.

Model procesa zapišemo v prostoru stanj kot

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t) \tag{7.157}$$

$$\mathbf{y} = \mathbf{g}(\mathbf{x}, t) \tag{7.158}$$

kjer je **x** vektor stanja, **u** je vektor vhodnih signalov, **y** pa vektor izhodnih signalov procesa. **f** je sistemska funkcija in **g** izhodna funkcija procesa. Zaradi enostavnejše obravnave bomo v nadaljevanju privzeli, da so vsa stanja merljiva, zato bomo kot vhod v generator privzeli kar vektor stanj **x**.

Krmilnik lahko opišemo kot deterministični avtomat. To je sistem, ki ga sestavlja množica diskretnih stanj, prehodi med njimi pa nastopajo kot posledica zunanjih dogodkov. Avtomat je determinističen, če trenutno stanje in zunanji dogodek enolično določata, katero bo naslednje stanje. Zunanje dogodke ponazorimo s simboli na vhodu avtomata - v našem primeru jih bomo imenovali simboli procesa. Poleg tega je vsakemu stanju prirejen tudi simbol na izhodu avtomata, ki se pojavi tedaj, ko avtomat preide v to stanje. Simbole na izhodu bomo imenovali krmilni simboli.

Matematično takšen avtomat opišemo z zbirko množic in preslikav, ki jih združimo v petorko

$$A = \left\{\tilde{S}, \tilde{Z}, \tilde{R}, \delta, \phi\right\} \tag{7.159}$$

Pri tem je $\tilde{S}$ množica stanj, $\tilde{Z}$ je množica simbolov procesa, $\tilde{R}$ je množica krmilnih simbolov, $\delta : \tilde{S} \times \tilde{Z} \to \tilde{S}$ je funkcija prehajanja stanj avtomata in $\phi : \tilde{S} \to \tilde{R}$ je izhodna funkcija avtomata. Delovanje krmilnika lahko opišemo z dvema diferenčnima enačbama

$$\tilde{s}[n] = \delta(\tilde{s}[n-1], \tilde{z}[n]) \tag{7.160}$$

$$\tilde{r}[n] = \phi(\tilde{s}[n]) \tag{7.161}$$

kjer je $\tilde{s}[n] \in \tilde{S}$, $\tilde{z}[n] \in \tilde{Z}$ in $\tilde{r}[n] \in \tilde{R}$. Enačba (7.160) določa, v katero stanje bo prešel krmilnik ob nastopu določenega simbola procesa, enačba (7.161) pa, kateri krmilni simbol se bo ob tem pojavil na izhodu krmilnika. Spremenljivka n pri tem ponazarja medsebojno časovno lego stanj in simbolov v sekvenci.

Vmesnik opišemo z dvema preslikavama. Prva preslika zaporedje izhodnih simbolov krmilnika v odsekoma konstanten vhod procesa:

$$\mathbf{u}(t) = \gamma(\tilde{r}[n]) \tag{7.162}$$

Vhodni signal v proces lahko zavzame le določene konstantne vrednosti, ki ustrezajo posameznim izhodnim simbolom. Signal se lahko spremeni le tedaj, ko se na izhodu krmilnika pojavi simbol (ob nastopu ustreznega dogodka).

Druga preslikava preslika prostor stanj procesa v množico vhodnih simbolov krmilnika. Zapišemo jo kot

$$\tilde{z}[n] = \alpha(\mathbf{x}(t)) \tag{7.163}$$

Iz enačbe (7.163) bi lahko sklepali, da se ob spreminjanju stanja procesa neprestano generirajo procesni simboli. Temu ni tako, ker funkcija α temelji na razdelitvi prostora stanj na več področij. Vsakemu področju ustreza en simbol, ki se generira le takrat, ko stanje procesa vstopi v prirejeno področje.

Opisani model je zelo enostaven. Proces je čisti zvezni proces, krmilnik je edini diskretni podsistem. Pogosto so že v samem procesu prisotne diskretne značilnosti, npr. pri šaržnih procesih. Zato nekateri pristopi, npr. (Lennartson in soavtorji, 1996), opisani koncept razširjajo tako, da se že znotraj procesa nahaja tudi diskretni podsistem. Sam proces ima torej podobno strukturo, kot je prikazana na Sl. 7.125, le da diskretni del ni več krmilnik, temveč predstavlja notranja diskretna stanja procesa. Na ta način lahko modeliramo različno dinamiko procesa v različnih delovnih področjih, binarne senzorje ter dvo- ali večpoložajne diskretne aktuatorje. Krmilnik v tem primeru nastopa kot dodatni diskretni sistem, ki preko vmesnika komunicira tako z zveznim, kot z diskretnim delom procesa.

Če gledamo vmesnik in proces skupaj kot eno celoto, dobimo v vseh načinih kombiniranega modeliranja sistem diskretnih dogodkov, ki ga imenujemo diskretni proces - tega krmilnik vidi in poskuša voditi. Krmilnik torej vidi abstraktni model procesa oziroma višjenivojski model, ki je manj podroben od opisa z diferencialnimi ali diferenčnimi enačbami. Kompleksnost tega abstraktnega modela je odvisna od izvedbe vmesnika. Lahko si predstavljamo, da bolj podrobno ko razdelimo prostor stanj (na čim

več področij), bolj je pripadajoči diskretni model kompleksen. Ob tem velja omeniti še to, da za razliko od krmilnika, diskretni model procesa ni vedno determinističen. Lahko se zgodi, da je iz določenega področja prostora stanj možen prehod v različna področja, pač glede na to, v kateri točki se sistem trenutno nahaja.

Vse doslej smo obravnavali predvsem modeliranje hibridnih sistemov. Končni namen takšnega modeliranja je načrtovanje vodenja sistema. Pri načrtovanju vodenja sistemov diskretnih dogodkov ločimo med dvema temeljnima pristopoma:

- uporaba tehnik verifikacije;
- algoritmična sinteza vodenja.

Pri prvem pristopu načrtamo vodenje intuitivno, nato pa preverjamo, če je želeno delovanje modeliranega sistema doseženo ob uporabi načrtanih pravil vodenja. Glavni problem tega pristopa so pri večini sistemov zelo zahtevni postopki formalne verifikacije. Le redko lahko na enostaven način formalno dokažemo, da ima vodeni sistem res želene lastnosti. V večini primerov si pomagamo s simulacijo, vendar se moramo zavedati, da pri simulaciji običajno zajamemo le majhen del možnih dogodkov v sistemu, in zato uspešna simulacija še ne zagotavlja pravilnosti vodenja.

Pri drugem pristopu izpeljemo pravila vodenja iz opisa procesa in zahtev za delovanje. Takšne zahteve so pogosto podane v obliki omejitev vektorja stanja ($\mathbf{x}(t)$ mora npr. ostati znotraj nekega področja). Načrtano vodenje zagotavlja, da bodo zahteve izpolnjene, in je seveda močno odvisno od definicije le-teh. Vodenje načrtujemo z orodji teorije sistemov diskretnih dogodkov, zato proces z vmesnikom pretvorimo v diskretni model. Ta model dobimo z razdelitvijo zveznega prostora stanj na področja, katerih meje so v tesni zvezi s postavljenimi zahtevami. Če npr. zahtevamo, da temperatura v procesu ostaja v pasu 40±5°C, je smiselno, da eno mejo področja potegnemo pri 35°C in drugo pri 45°C.

Ta pristop je blizu inženirski praksi. Delitev prostora stanj je v praksi običajno dosežena z namestitvijo binarnih senzorjev ali z določitvijo mejnih vrednosti izhodov zveznih senzorjev. S tem definiramo tudi dogodke, ki vstopajo v krmilnik. Vendar pa je sinteza krmiljenja, ki temu sledi, v praksi običajno intuitivna. Poglavitni problem pri tem pristopu je, da so obstoječe metode formalne sinteze omejene na posebne, relativno enostavne primere, ob tem pa tudi računsko precej zahtevne.

Za oba pristopa velja, da so matematične metode, ki so na voljo, zahtevne in presegajo okvir inženirske prakse. Zato se na tem mestu ne bomo spuščali v podrobnosti, temveč bomo le nakazali razvoj modela in intuitivno sintezo vodenja na enostavnem primeru.

∇

Primer 7.20: *Polnjenje mešalne posode*

Oglejmo si mešalno posodo na Sl. 7.126, ki je del šaržnega procesa. V posodi se mešata

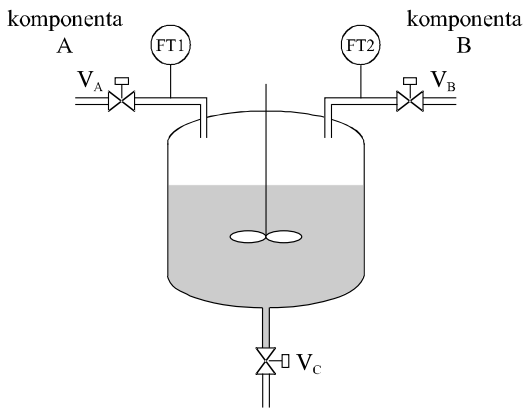
dve komponenti (A in B), ki vstopata preko dvopoložajnih ventilov V_A in V_B. Količino natočene snovi merimo s pomočjo merilcev pretoka FT1 in FT2. Predpostavimo, da sta pripadajoča volumska pretoka Φ_A in Φ_B konstantna, če je ustrezni ventil odprt.

Označimo z $V_A(t)$ in $V_B(t)$ delna volumna komponent A in B. Delna volumna predstavljata količino posamezne snovi v mešalni posodi ob času t. Spremembi delnih volumnov obeh komponent v posodi sta določeni z vhodnima pretokoma:

$$\begin{aligned} \dot{V}_A(t) &= \Phi_A(t) \\ \dot{V}_B(t) &= \Phi_B(t) \end{aligned} \tag{7.164}$$

Skupni volumen snovi v mešalni posodi je

$$V(t) = V_A(t) + V_B(t) \tag{7.165}$$



Sl. 7.126. Mešalna posoda.

Če izberemo oba delna volumna za spremenljivki stanja, vhodna pretoka kot vhoda v proces, skupni volumen pa kot izhod, lahko zapišemo zvezni model procesa kot

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{bmatrix} \Phi_A(t) \\ \Phi_B(t) \end{bmatrix} \\ y &= \begin{bmatrix} 1 & 1 \end{bmatrix} \cdot \mathbf{x} \end{aligned} \tag{7.166}$$

Cilj vodenja je natočiti v posodo določeno količino (V_{Af}) komponente A in določeno količino (V_{Bf}) komponente B. Ker pri tem obstaja nevarnost neželenih reakcij, moramo paziti, da medsebojno razmerje komponent ostaja v določenih mejah. To zahtevo zapišemo kot omejitev stanja sistema v obliki

$$k_1 V_A(t) \le V_B(t) \le k_2 V_A(t) \tag{7.167}$$

Pri tem moramo upoštevati še, da vhodna pretoka pri odprtih ventilih nista enaka, temveč tedaj velja

$$\Phi_B < k_1 \Phi_A \tag{7.168}$$

Množico krmilnih simbolov $\widetilde{R}$ sestavimo glede na možne kombinacije stanj obeh ventilov. Dobimo

$$\widetilde{R} = \left\{ \widetilde{r}_A, \widetilde{r}_B, \widetilde{r}_{AB}, \widetilde{r}_0 \right\} \tag{7.169}$$

Indeks simbola pri tem označuje, kateri ventil bo po nastopu simbola odprt (ventil V_A, ventil V_B, oba ventila hkrati ali noben).

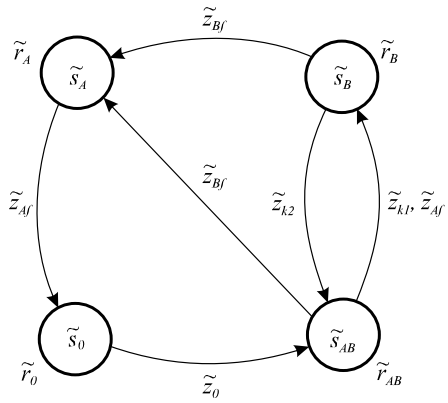
Množico simbolov procesa $\widetilde{Z}$ pa sestavimo glede na zahteve za delovanje sistema. V danem primeru iz zahtev sledi

$$\widetilde{Z} = \left\{ \widetilde{z}_{Af}, \widetilde{z}_{Bf}, \widetilde{z}_{k1}, \widetilde{z}_{k2}, \widetilde{z}_0 \right\} \tag{7.170}$$

Prva dva simbola pri tem pomenita, da je dosežena meja V_{Af} oziroma V_{Bf}, druga dva, da sta bili kršeni omejitvi (7.167), zadnji simbol pa pomeni zahtevo za start polnjenja (generira se npr. takrat, ko je posoda prazna). Funkcija α torej generira

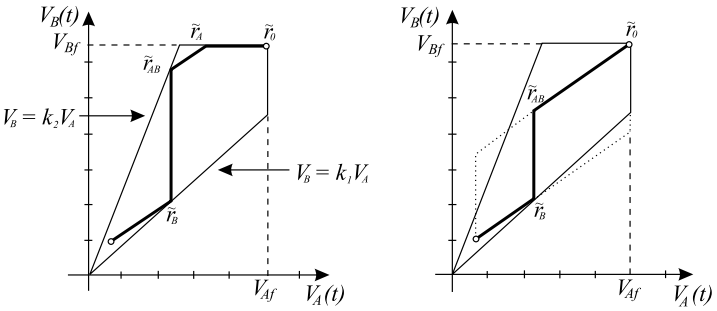
$$\begin{aligned} &\widetilde{z}_{Af}, \quad \text{ko} \quad V_A(t) \ge V_{Af} \\ &\widetilde{z}_{Bf}, \quad \text{ko} \quad V_B(t) \ge V_{Bf} \\ &\widetilde{z}_{k1}, \quad \text{ko} \quad V_B(t) < k_1 V_A(t) \\ &\widetilde{z}_{k2}, \quad \text{ko} \quad V_B(t) > k_2 V_A(t) \end{aligned} \tag{7.171}$$

Pri pogoju, da sta začetna volumna v mešalni posodi znotraj zahtevanih meja, lahko definiramo krmilnik, kot prikazuje Sl. 7.134.



Sl. 7.127. Krmilnik

Krmilnik na sliki ima 4 stanja, ob prehodih med stanji so označeni procesni dogodki, ki te prehode prožijo, ob stanjih pa so označeni tudi krmilni dogodki, ki nastanejo ob vstopu krmilnika v določeno stanje. Pri tem definiramo začetno stanje sistema kot stanje, v katerem sta oba ventila odprta ($\tilde{s}_{AB}$). Sl. 7.128a prikazuje polnjenje mešalne posode v fazni ravnini. Vrisani sta omejitvi stanja (7.167) in trajektorija, ki jo dobimo pri načrtanem krmilniku. Očitno je, da ta rešitev ni časovno optimalna. Časovno optimalen potek trajektorije stanja prikazuje Sl. 7.128b. V tem primeru je zanimivo, da iz množice $\tilde{R}$ uporabimo le simbola $\tilde{r}_B$ in $\tilde{r}_{AB}$, ventil V_B je torej stalno odprt. S pikčasto črto sta označeni optimalni trajektoriji, če v sistemu ne bi imeli dodatnih omejitev stanja.



Sl. 7.128. Polnjenje mešalne posode

Δ

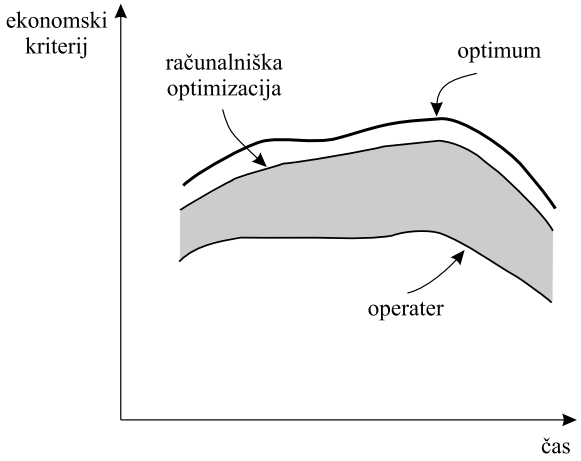
Hibridno vodenje je področje raziskovanja, ki se intenzivno razvija v zadnjih letih. V akademskih krogih se porajajo številni novi pristopi, le redki pa so že dozoreli za industrijsko prakso. Medtem ko je koncept modeliranja in simulacije hibridnih sistemov že dokaj dodelan, pa so metode sinteze ustreznih algoritmov vodenja še precej nedodelane. Obstajajo pristopi k načrtovanju za posebne primere, splošnih rešitev pa zaenkrat še ni. V večini praktičnih primerov hibridnih sistemov vodenja je tako edina rešitev intuitivno načrtovanje, pri čemer sta lahko v veliko pomoč modeliranje in simulacija, s katero lahko do neke mere preverjamo ustreznost načrtanega vodenja. Številna sodobna simulacijska orodja imajo vgrajeno podporo hibridni simulaciji. Takšna orodja so npr. G2, Dymola, Matlab/Simulink.

7.3.5 Optimiranje

Z optimizacijo smo se že srečali pri obravnavi nižjih nivojev vodenja, kjer smo z optimizacijo poskušali doseči čim boljše delovanje samega algoritma vodenja. Pri regulaciji smo npr. z optimizacijo želeli izboljšati delovanje regulatorjev, kot so PID regulator, adaptivni regulator, prediktivni regulator, itd. Optimiranje, ki ga predstavljamo v tem poglavju, pa se nanaša na optimizacijo delovanja procesa kot celote in predstavlja eno od funkcij višjih nivojev vodenja.

7.3.5.1 Definicija in značilnosti optimiranja

Optimiranje v računalniškem sistemu vodenja vključuje postopke vodenja, s katerimi iščemo oz. želimo doseči čim bolj optimalno delovanje procesa glede na ekonomske rezultate obratovanja, t.j. profit oz. izplen. Optimiranje je tesno povezano s cilji obratovanja industrijskega procesa. Le-ti običajno vključujejo kvaliteto produktov, količino proizvodnje, porabo energije, onesnaževanje okolja, varnost obratovanja, ipd. Doslej obravnavane funkcije računalniškega vodenja (zbiranje podatkov o sistemu, sekvenčno vodenje, regulacija, nadzor) posredno že prispevajo k navedenim ciljem vodenja, s funkcijo optimiranja pa želimo neposredno vplivati na cilje obratovanja na ta način, da premostimo razkorak med cilji vodenja industrijskega obrata na eni strani in specifičnimi nastavitvami vhodov v sistem procesnega vodenja na drugi strani. Ta naloga je zelo pogosto prepuščena tehnologom oz. operaterjem, ki se ob upoštevanju razmeroma širokega varnostnega območja obratovanja procesa uspejo bolj ali manj približati teoretičnemu optimumu. Kot je prikazano na Sl. 7.129., je prav razlika med krivuljo operaterja in krivuljo optimuma potencial za uvedbo funkcije optimiranja (Savas, 1965).



Sl. 7.129. Krivulja optimiranja

Optimiranje torej za vzdrževanje obratovanja procesa ni potrebno, potrebno pa je za čim boljše obratovanje. Vpeljevanje funkcije optimiranja v računalniški sistem vodenja mora biti zato ekonomsko upravičeno. Tako v določenih primerih že samo osnovne funkcije z nadzorom upravičijo računalniško vodenje, spet drugje pa so osnovne funkcije vodenja samo vmesni nivo, ki se ekonomsko obrestuje šele dolgoročno z vpeljavo optimiranja. Pri nekaterih procesih optimiranje ni potrebno, ker proces obratuje na meji omejitev.

Optimiranje je posebej primerno za procese, kjer se pogoji obratovanja pogosto spreminjajo (npr. pri spremembah lastnosti vhodnih surovin, pri spreminjanju proizvodnih zahtev zaradi tržišča ali medsebojnega usklajevanja posameznih enot proizvodnega procesa), pri proizvodnji več različnih produktov, itd. Možnosti

optimiranja je še posebej smiselno proučiti pri procesih, ki izkazujejo naslednje značilnosti (Latour, 1979):

- na voljo je dovolj neodvisnih spremenljivk oz. prostostnih stopenj za vplivanje na proces;
- učinkovitost obratovanja procesa je v zadostni meri odvisna od spreminjanja neodvisnih spremenljivk;
- v procesu so prisotne pogoste in opazne motnje, ki imajo pomemben vpliv na učinkovitost procesa;
- obstajajo spremenljivke, katerih vrednosti ostajajo dalj časa fiksirane in jih ne spreminjamo predvsem zaradi večje enostavnosti vodenja in obratovanja. V tem primeru je smiselno proučiti spreminjanje teh spremenljivk s ciljem povečati učinkovitost obratovanja.

Zelo pogosto se optimiranje nanaša na iskanje optimalnih obratovalnih pogojev, kar običajno vključuje:

- *optimiranje tehnoloških parametrov obratovanja;*
- *iskanje optimalnih vrednosti krmilnih spremenljivk* ali
- *optimalno nastavitev želenih (referenčnih) vrednosti regulacijskih sistemov*.

Poznamo pa tudi druge vrste optimiranja, npr.:

- *optimizacijo dodeljevanja strežnih naprav* različnim enotam proizvodnega procesa (ang. allocation) in
- *optimizacijo razporejanja*, ki vključuje časovno razporejanje tehnoloških operacij (ang. sequencing, scheduling).

Pogost primer optimiranja je tudi *koordinacija*, ki nastopa zlasti v obsežnejših industrijskih postavitvah v procesni industriji, kjer je potrebno medsebojno koordinirati obratovanje več enot ali podprocesov, in pri procesih, kjer zasledujemo več različnih in medsebojno nasprotujočih si ciljev vodenja.

7.3.5.2 Glavni pristopi k optimiranju

Pri funkciji optimiranja je potrebno dejanski problem vodenja formulirati kot optimizacijski problem. Optimizacija se nanaša na cenilko, ki jo želimo minimizirati ali maksimizirati na ta način, da z ustreznim algoritmom optimizacije določimo vrednosti spremenljivk, ki dajo optimalno vrednost cenilke. Pri sistemih za vodenje z optimizacijo določimo vrednosti vhodnih (neodvisnih) spremenljivk vodenja, s katerimi želimo vplivati na izhodne (odvisne) spremenljivke vodenja. Slednje so v relaciji z ekonomskimi učinki obratovanja procesa, ki so definirani kot cilji vodenja in so matematično podani s cenilko.

Pri formulaciji optimizacijskega problema je torej potrebno opredeliti naslednje elemente:

- določiti vhodne (neodvisne) spremenljivke vodenja;
- določiti izhodne (odvisne) spremenljivke vodenja;
- definirati cilje vodenja v obliki ustrezne cenilke;
- izbrati ustrezen algoritem optimizacije.

Pri optimiranju procesov se običajno pokaže, da je v konkretni izvedbi pomembna predvsem definicija problema z izbiro ustreznih spremenljivk ter ciljev vodenja, ki pa je v veliki meri specifična za vsak problem posebej.

Izbira algoritma za optimizacijo zavisi od vrste in postavitve problema. Za probleme sprotne optimizacije industrijskih procesov so primerni robustni algoritmi, ki vedno dajo kot rezultat sprejemljivo rešitev. Pogosto uporabljene metode so npr. linearno programiranje, nelinearno oz. dinamično programiranje, gradientne metode optimizacije, metode z Lagrange-ovimi multiplikatorji ter posebej načrtani hevristični postopki (Matko, 1993).

Pri optimiranju procesov v splošnem ločimo več pristopov, ki so predstavljeni v nadaljevanju: direktno optimizacijo procesa in optimizacijo s pomočjo modela, statično in dinamično optimizacijo.

Direktna optimizacija procesa

Direktna optimizacija je metoda optimiranja, kjer med normalnim obratovanjem in neposredno na procesu izkustveno določimo optimalno točko obratovanja. Pri tem ne potrebujemo modela procesa, potrebno pa je zato nenehno eksperimentiranje na procesu. Postopek se ciklično ponavlja in sestoji iz naslednjih faz:

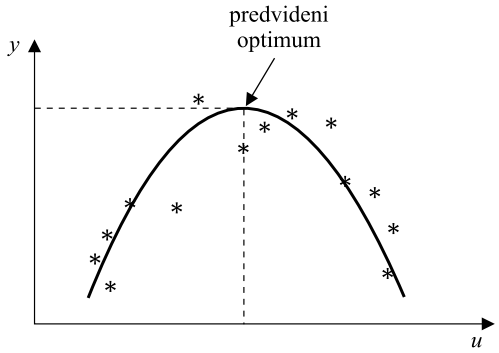
- spoznavanje oblike prostora cenilke;
- estimacija gradientov v prostoru cenilke z regresijsko analizo in
- določitev nove (boljše) točke obratovanja s premikom v smeri gradienta.

Spoznavanje oblike prostora cenilke, t.j. odvisnosti med vhodnimi spremenljivkami procesa in učinki obratovanja, poteka z eksperimentiranjem na procesu. Pri tem z namenskim izvajanjem sprememb v nastavitvah vhodnih spremenljivk opazujemo posledične spremembe rezultatov obratovanja. Izvajanje eksperimentov mora biti skrbno načrtovano, tako da je končni produkt kljub spremembam vhodov še vedno sprejemljiv. Po zadostnem številu eksperimentov se na osnovi dobljenih podatkov izvede regresijska analiza. Pri tem se določi regresijski model, ki matematično opiše dobljene vhodno-izhodne relacije prostora cenilke; z ustrezno matematično funkcijo skušamo doseči čim boljše ujemanje med meritvami in regresijskim modelom (Sl. 7.130). V nadaljevanju je na osnovi gradienta regresijske krivulje mogoče oceniti kombinacijo vhodnih veličin, kjer leži optimum regresijske krivulje z verjetnim boljšim delovanjem procesa. Izračun se preveri z eksperimenti v točki predvidenega optimuma.

Prednost direktne optimizacije je, da se le-ta vrši na obstoječem procesu, tak kot pač je, in da ne prihaja do omejitev, ki so povezane s (pomanjkljivim) teoretičnim znanjem o

procesu. Zato je primerna predvsem za procese, ki jih slabo razumemo, in kjer je težko priti do ustreznega modela.

Metoda pa ima tudi resne pomanjkljivosti, ki omejujejo njeno uporabo. Predvsem se pri procesih z velikim številom spremenljivk povečuje število eksperimentov, posebej še, če upoštevamo ponavljanje eksperimentov, ki je potrebno, da se izloči šum meritev.



Sl. 7.130. Postavitev regresijskega modela na osnovi merjenih podatkov

Čas eksperimentiranja se povečuje tudi zaradi dinamike procesa, saj je pri izvajanju eksperimentov potrebno (glede na časovno konstanto procesa) čas eksperimentiranja podaljšati, tako da izhod procesa ob spremembi vhodov doseže novo ustaljeno stanje. Pristop tudi ni primeren za procese z dolgim časom eksperimentiranja in kjer se pogosto pojavljajo motnje.

Optimizacija s pomočjo modela

Prav zaradi omenjenih pomanjkljivosti je v mnogih primerih ustreznejši pristop k optimiranju s pomočjo modela. Pri tem ločimo dve vrsti modelov: model procesa in ekonomski model. Model procesa predstavlja fizikalno-kemijski vidik in opisuje obnašanje procesa v odvisnosti od procesnih veličin, kot so temperatura, pretok, sestava, koncentracije, itd. Ekonomski model pa opisuje učinek obratovanja na poslovanje; podan je s cenilko, ki je pogosto nelinearna in zajema medsebojno nasprotujoče si zahteve vodenja.

Pri pristopih k optimiranju, ki temeljijo na modelih, postane uspešnost optimiranja v veliki meri odvisna od modeliranja procesa, saj je od natančnosti in reprezentativnosti modela odvisna natančnost določitve optimuma. Pri optimizaciji s pomočjo modela, ki je bil dobljen na osnovi empiričnega ali teoretično-empiričnega pristopa k modeliranju, je tudi pomembno sprotno popravljanje modela, tako da se zagotovi čim boljše ujemanje procesa in modela. S popravljanjem modela skušamo kompenzirati pomanjkljivosti, ki so nastopile pri modeliranju, kot npr. poenostavitev strukture modela, ocena parametrov iz meritev v ozkem območju obratovanja procesa, šum meritev, itd. Za namene sprotne optimizacije se pogosto uporabljajo pristopi, kjer gre za kombinacijo znanega fizikalno-

kemijskega modela z osnovnimi konstantami, ter sprotne identifikacije izkustvenih parametrov modela na osnovi meritev.

Uporaba modela in simulacije omogoča tudi:

- opredelitev obnašanje procesa pri različnih pogojih obratovanja in na ta način ocenitev potencialnih doprinosov optimizacije;
- ocenitev vpliva omejitev na obratovanje procesa;
- iskanje globalnega optimuma (za razliko od lokalnega optimuma) z izvedbo optimizacij iz različnih začetnih točk;
- ocenitev oblike prostora cenilke, kar vpliva tudi na izbiro optimizacijskega algoritma.

Statična optimizacija

Pri statični optimizaciji ne upoštevamo dinamične narave procesa, to pomeni, da čas ne nastopa v optimizacijskih izračunih. Tak način optimiranja je primeren za procese, ki obratujejo pri razmeroma konstantnih obratovalnih pogojih, kjer je pogostost motenj majhna, proces pa se relativno hitro odzove na spremembe obratovanja. Pri statični optimizaciji zato zadostujejo statični modeli, saj je bistvo takega modela predvsem odnos vzrok-posledica med neodvisnimi spremenljivkami, ki jih spreminjamo s sistemom vodenja, ter odvisnimi spremenljivkami, ki vplivajo na učinke vodenja.

Dinamična optimizacija

Pri dinamični optimizaciji pa je nasprotno dinamika procesa pomembna, vrednosti krmilnih ali regulirnih spremenljivk vodenja pa so določene kot funkcije časa. Dinamična optimizacija zahteva tudi dinamičen model procesa.

∇

Primer 7.21: *Optimizacija kvalitete šaržnega procesa s prilagodljivimi recepti*

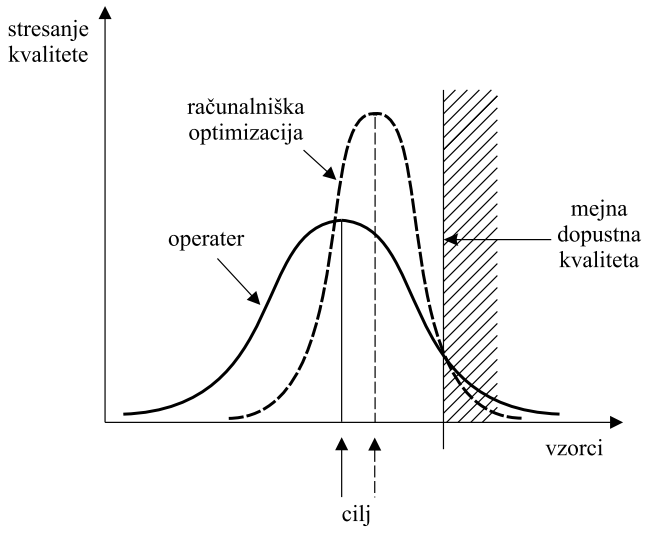
Pogost primer optimiranja je zmanjšanje stresanja kvalitete končnega produkta. Z računalniško optimizacijo je v nekaterih primerih mogoče doseči manjše stresanje kvalitete in s tem ožjo porazdelitveno krivuljo, kot je prikazano na Sl. 7.131.

V takem primeru se ciljna vrednost kvalitete lahko premakne bliže k mejni vrednosti, ki predstavlja še dopustno kvaliteto in je določena z zahtevami trga. Tak premik lahko pomeni predvsem boljši izkoristek surovin, krajši čas proizvodnje, manjšo porabo energije, itd.

Na področju šaržnih procesov je eden od pristopov k optimiranju t.i. koncept prilagodljivih receptov (Rijnsdorp, 1991; Verwater-Lukszo, 1995).

Proizvodnja pri šaržnih procesih poteka po receptu, ki predpisuje korake in pogoje, ki so potrebni za izdelavo izdelka z želenimi lastnostmi. Recept je običajno predpisan s strani

načrtovalca sistema in predstavlja optimalni potek šarže za določen vhod v proces in določene obratovalne pogoje. Recept in parametri recepta pogosto ostajajo nespremenjeni, čeprav se lastnosti reaktantov, njihovo doziranje in obratovalni pogoji spreminjajo od šarže do šarže, spremenijo pa se lahko tudi cilji vodenja. V takih primerih se poslabšajo rezultati obratovanja (kvaliteta, izkoristek), zato so navedene motnje tudi vzrok za optimiranje. Ideja prilagodljivih receptov je, da se znana odstopanja vhodov v šaržo, obratovalnih pogojev med šaržo in zahtev vodenja kompenzira s spremembo parametrov recepta, s ciljem doseči optimalno obratovanje. Kot parametri recepta nastopajo doziranje vhodnih komponent, časi trajanja posameznih faz procesa, ipd. Zelo pogosto se optimizacija nanaša na fazo transformacije (t.j. fazo šarže, v kateri npr. pri kemijskih procesih poteče reakcija). V tem primeru se optimizacija nanaša na določitev optimalnih potekov procesnih spremenljivk za to fazo (t.j. temperature, tlaka, koncentracije, ipd.) in se rešuje z metodami dinamične optimizacije podobno kot pri zveznih procesih.

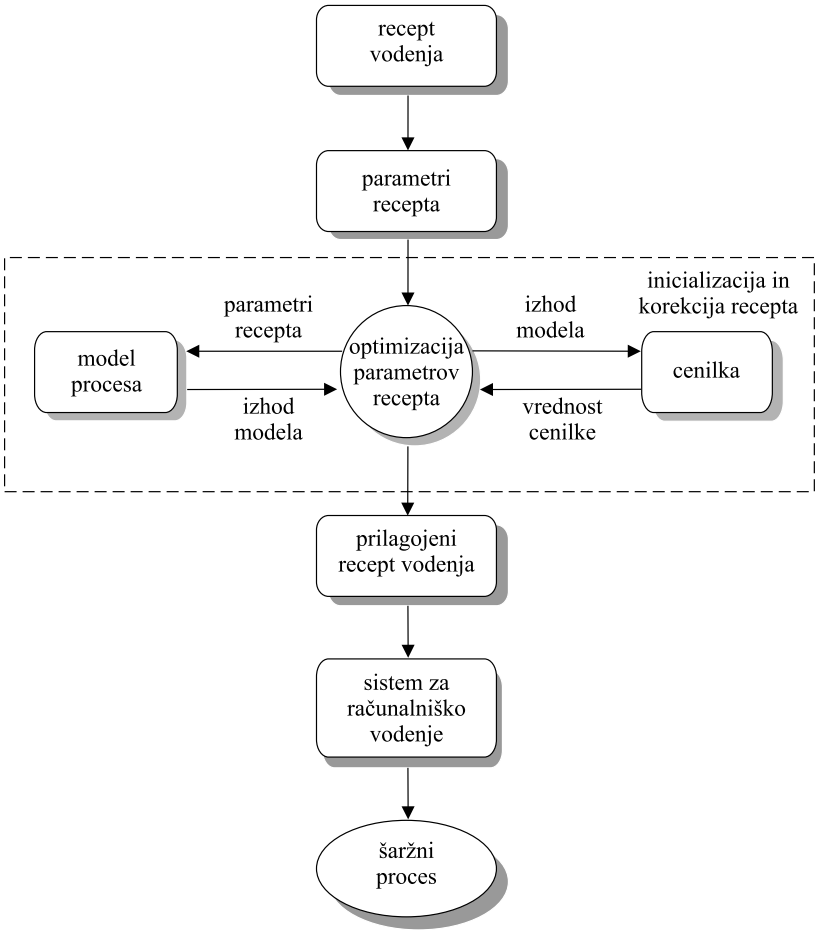


Sl. 7.131. Stresanje kvalitete ob uvedbi funkcije optimiranja.

Pri postopku prilagodljivih receptov je poleg (pravočasnega) odkrivanja motenj pomembno predvsem pravilno ukrepanje (vodenje). V splošnem koncept vključuje tri možne načine izboljšanja receptov, t.j. inicializacijo recepta, korekcijo recepta in že omenjeno eksperimentalno optimizacijo, ki poteka direktno na procesu. Pri inicializaciji recepta gre za popravek parametrov recepta na osnovi znanih odstopanj vhodnih spremenljivk pred pričetkom šarže, pri korekciji pa parametre recepta popravljamo sproti, t.j. med samo šaržo, če pride do odstopanj obratovalnih pogojev.

Prilagoditve parametrov recepta temeljijo na optimizaciji s pomočjo modela procesa (razen pri eksperimentalni optimizaciji, ki poteka z eksperimentiranjem direktno na procesu - glej podpoglavje “Direktna optimizacija procesa”). Z modelom so podani

odnosi med vhodnimi spremenljivkami procesa, parametri recepta (le-ti predstavljajo krmilne spremenljivke vodenja) ter izhodnimi spremenljikami procesa. Optimizacija pa se vrši na osnovi cenilke, s katero zapišemo zahteve vodenja. Pri optimizaciji za posamezno šaržo izračunamo s pomočjo modela vrednost cenilke za trenutne obratovalne pogoje in jo optimiramo s spreminjanjem parametrov recepta. Optimalni parametri recepta se nato posredujejo v računalniški sistem vodenja z osnovnim vodenjem, ki skrbi za izvedbo šarže po receptu (Sl. 7.132).



Sl. 7.132. .Postopek optimiranja parametrov recepta

Postopek prilagodljivih receptov si poglejmo za primer šaržnega procesa hidrolize. Hidroliza je eden od najpomembnejših podprocesov v proizvodnji TiO_2 pigmenta, kjer se iz $TiOSO_4$ ob dodajanju kali in vode ter segrevanju raztopine ustrezen čas na temperaturi vrelišča obori TiO_2 gel:

$$TiOSO_4 + H_2O \leftrightarrow TiO_2 + H_2SO_4. \qquad (7.172)$$

Potek šarže predpisuje recept, s katerim so določene tudi nekatere pomembne spremenljivke vodenja, kot so dodatek kali, količina in dinamika dodajanja vode v času reakcije, trajanje obarjanja, itd. Večina teh parametrov ostaja nespremenjenih, čeprav koncentracija vhodne raztopine, ki je rezultat proizvodnje predhodnih podprocesov, niha od šarže do šarže. Posledica so nihanja v porazdelitvi velikosti oborjenih delcev in posledično spreminjanje kvalitete končnega pigmenta, ki včasih presega dopustne meje.

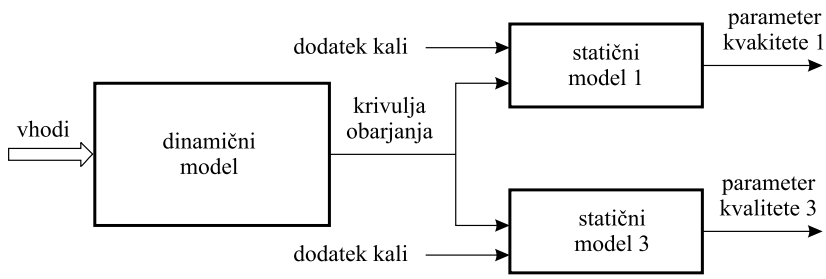
Zaradi številnih in medsebojno zelo kompleksnih povezav procesa operater praktično ne more vršiti popravkov recepta, to nalogo pa lahko izvaja računalniški sistem ob ustrezni algoritemski podpori.

Za proces hidrolize smo načrtali postopek inicializacije recepta, kjer nihanja vhodnih koncentracij kompenziramo z dodajanjem kali. Osnovno izhodišče pri tem je bilo, da z ustreznim doziranjem dodatka kali stabiliziramo potek kemijske reakcije (krivulje obarjanja, glej Sl. 7.134) ter na ta način dosežemo tudi enakomernejšo kvaliteto produkta.

Model procesa

V ta namen smo razvili model procesa. Glede na zahtevnost procesa hidrolize, ki ga je zaradi kompleknosti težko natančno opisati, ter različnih znanj, ki so bila na voljo (znanja s področja kemijske kinetike, eksperimentalni laboratorijski podatki, podatki iz rednega obratovanja, izkušnje operaterjev), smo pristopili k razvoju semi-empiričnega matematičnega modela. Model podaja odvisnost med vhodnimi in obratovalnimi pogoji ter parametri recepta na eni strani ter izhodi procesa oz. rezultati obratovanja na drugi strani.

Potek krivulje obarjanja v odvisnosti od vhodnih in obratovalnih pogojev smo predstavili z dinamičnim modelom, ki temelji na zakonih kemijske kinetike, preslikavo med krivuljo obarjanja in kazalci kvalitete pa smo predstavili z empiričnim (regresijskim) modelom, ki je prikazan na Sl. 7.133. Potek modeliranja je podrobneje predstavljen v (Šel, 1997).



Sl. 7.133. Struktura semi-empiričnega dinamičnega modela procesa hidrolize

Dinamični model je predstavljen z enačbami kemijske kinetike, ki za reakcijo tipa

$A + B \rightarrow C$ določajo, da je hitrost reakcije *r(t)* odvisna od koncentracij komponent *A* in *B* in jo podamo z izrazom

$$r(t) = k(T)c_A^a c_B^b, \tag{7.173}$$

kjer sta *a* in *b* empirično določeni konstanti, *k(T)* pa je specifična hitrost reakcije, ki je podana z Arrheniusovo enačbo $k = Ae^{-E/RT}$ (*E* je aktivacijska energija, *T* je temperatura, *R* je kemijska konstanta, *A* je frekvenčni faktor). Z upoštevanjem navedenih zakonitosti lahko dinamični matematični model hidrolize po poenostavitvah (reakcije so večinoma prvega reda, zato so eksponenti modela, razen pri dodatku kali u_{kali}, enaki 1; v postopku estimacije parametrov so bile nekatere optimirane vrednosti parametrov približno enake 0) predstavimo z naslednjimi enačbami:

$$\begin{aligned}
\frac{dc_{TiOSO4}}{dt} &= -k_1 c_{TiOSO4} c_{H2O} \\
\frac{dc_{H2O}}{dt} &= -k_1 c_{TiOSO4} c_{H2O} + k_4 u_{H2O} \\
\frac{dc_{TiO2}}{dt} &= k_1 c_{TiOSO4} c_{H20} - k_2 c_{TiO2} u_{kali}^{k3} + r_{gel} \\
\frac{dc_{H2SO4}}{dt} &= -\frac{dc_{TiOSO4}}{dt} \\
\frac{dc_{TiO2gel}}{dt} &= k_2 c_{TiO2} u_{kali}^{k3} - r_{gel}
\end{aligned} \tag{7.174}$$

kjer so *c* koncentracije posameznih komponent, *u* so krmilne spremenljivke procesa (dodatek vode in dodatek kali), r_{gel} je aktivnost gela (konstanta), *k* so parametri modela, pri čemer sta k_1 in k_2 predstavljena z Arrheniusovo enačbo in skupaj vsebujeta 4 parametre. Izhod procesa je koncentracija $c_{TiO2gel}$, ki predstavlja tudi izkoristek hidrolize.

Statični model pa je določen s preslikavo posameznih delov krivulje obarjanja v kazalce kvalitete in je npr. za kazalec kvalitete D_{50} (premer skupka kristalov - flokul) določen po naslednji empirični relaciji:

$$D_{50} = p_1 K_{23} + p_2 K_{23}^2 + p_3 K_{67} + p_4 K_{89} + p_5 u_{kali} + p_6 K_{maks} + p_7 K_{maks}^2 + p_8 K_{78} K_{maks}^2 + p_9 \tag{7.175}$$

kjer so *K* nakloni posameznih delov krivulje obarjanja (npr. K_{23} je naklon krivulje med 20 in 30% izkoristka), K_{maks} je maksimalni naklon krivulje obarjanja, *p* pa so parametri modela.

Estimacija parametrov v izrazih (7.174) in (7.175) je potekala na osnovi eksperimentalnih podatkov iz industrijskih šarž, ki so bile izvedene v širšem območju obratovanja procesa z načrtnim spreminjanjem posameznih vhodov v proces (dodatek kali, dodatek vode, čas dodatka vode). Parametri k so bili ocenjeni s pomočjo optimizacije, parametri p pa z metodo najmanjših kvadratov.

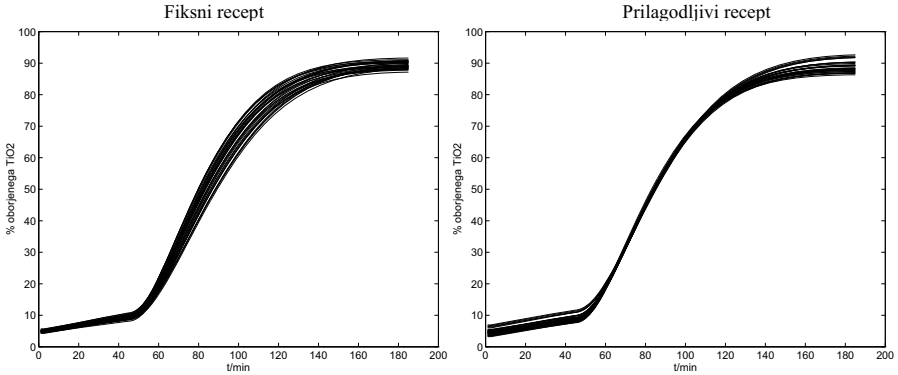
Cenilka

Cenilka J za optimizacijo parametrov recepta povezuje kazalca kvalitete, t.j. D_{50} in 1_{μ} (delež flokul, ki imajo premer pod 1 μm), ter končni izkoristek šarže Izk in je podana v obliki:

$$J = c_1(D_{50cilj} - D_{50})^2 + c_2(1_{\mu cilj} - 1_{\mu})^2 + c_3(Izk_{cilj} - Izk)^2 . \qquad (7.176)$$

Pri optimizaciji z minimizacijo J želimo zmanjšati odstopanja od ustreznih vnaprej podanih ciljnih vrednosti. Z izbiro parametrov c lahko damo ustrezen večji oz. manjši poudarek posameznemu cilju vodenja.

Sl. 7.134 podaja kot primer rezultate simulacij, ki ponazarjajo krivulje obarjanja za skupino šarž, pri kateri so se obratovalni parametri spreminjali podobno, kot se razmere obratovanja spreminjajo v procesu, vodenje pa je potekalo primerjalno pri konstantnih in prilagodljivih parametrih recepta. Prilagodljiv recept je vključeval spreminjanje dodatka kali, ki je potekalo po zgoraj opisanem postopku s pomočjo modela procesa in optimizacije. Iz slike je razvidno, da s prilagajanjem dodatka kali glede na obratovalne pogoje posamezne šarže dosežemo ponovljivejši potek šarž (ožji razpon krivulj obarjanja pri prilagodljivem receptu). V končni fazi pa se to posledično odraža tudi v manjšem stresanju končne kvalitete.



Sl. 7.134. Simulacija vodenja s fiksnim in prilagodljivim receptom vodenja s primerjavo potekov krivulj obarjanja.

Δ

∇

Primer 7.22: *Razporejanje šarž pri obratovanju paralelnih šaržnih reaktorjev*

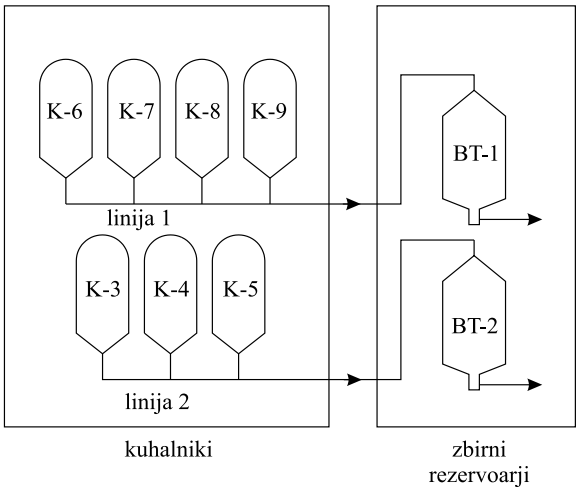
Razporejanje opravil lahko opredelimo zelo splošno kot časovno razporeditev virov, s pomočjo katerih želimo izvršiti določen nabor nalog ali opravil tako, da je zadoščeno omejitvam glede časa in kapacitet virov in so upoštevane odvisnosti med opravili. Primeri razporejanja so npr. razporejanje delovnih nalog v proizvodnem procesu, določitev vrstnega reda letal pri pristajanju, poslovanje s strankami pri bančnih okencih, izvajanje programov v računalniškem centru, ipd.

Razporejanje opravil sodi med zahtevnejše optimizacijske naloge, saj pri večini problemov razporejanja število izračunov za optimalno reševanje raste eksponentno z velikostjo problema, t.j. zlasti s številom opravil, ki jih je potrebno razvrstiti. Glavni problem pri reševanju je zato vprašanje optimalnosti rešitve in kako tako rešitev poiskati čim hitreje in učinkoviteje. Poleg tega so posamezni primeri razporejanja predvsem zaradi posebnosti, ki nastopajo v omejitvah, v veliki meri specifični, kar otežuje razvoj splošnejših metod reševanja. Pri problemih razporejanja je zato uporabljenih mnogo različnih pristopov, kot so specilizirane tehnike razporejanja in splošne tehnike kombinatorične optimizacije (npr. dinamično programiranje in metoda razvejitev in omejitev) (Conway in sod., 1967), ter povsem splošne metode kot so naključno vzorčenje, verjetnostni algoritmi (npr. simulirano ohlajanje) (Žerovnik, 1991), metode evolucijskega računanja (Filipič, 1992) in nevronske mreže.

V delu bomo kot primer reševanja predstavili razporejanje šarž pri procesu kuhanja celuloze (Hvala, 1992). Proces poteka paralelno v sedmih šaržnih reaktorjih - kuhalnikih (Sl. 7.135) in obsega tipične faze šaržnega procesa, t.j. polnjenje kuhalnika (faza1), segrevanje vsebine kuhalnika (faza 2), fazo reakcije (faza 3) in praznjenje kuhalnika (faza 4). Kuhanje poteka ločeno na vsakem kuhalniku, pri tem pa nekatere omejitve skupnih proizvodnih kapacitet vnašajo določeno stopnjo medsebojne odvisnosti v obratovanje kuhalnikov. Pomembne so zlasti naslednje:

- ena sama polnilna naprava za polnjenje kuhalnikov;
- omejena dobava pare v fazi segrevanja;
- skupni zbirni rezervoar za kuhalnike na posamezni liniji, ki sprejme omejeno količino suspenzije celuloze po kuhanju.

Posledica navedenih omejitev je, da naprave, ki so skupne vsem kuhalnikom, zakasnijo napredovanje kuhalnika v nasledjo šaržo oz. fazo šarže. Mrtvi čas oz. čakanje kuhalnikov pa je mogoče koristneje porabiti za podaljševanje tistih faz šarže, ki prispevajo k bolj ekonomičnemu obratovanju. Z razporejanjem lahko mrtvi čas vnaprej predvidimo in ga predhodno izkoristimo za podaljšanje faz segrevanja in kuhanja, kar pripomore k izboljšanju kvalitete in zmanjšanju porabe energije.



Sl. 7.135. Razporeditev kuhalnikov in zbirnih rezervoarjev pri procesu kuhanja celuloze

Razporejanje šarž vključuje dve nalogi, ki sta medsebojno povezani:

- razvrščanje kuhalnikov, ki določi najugodnejše *zaporedje* prihodnjega polnjenja kuhalnikov, in
- izračun *časov* za izvedbo posameznih šarž.

V nadaljevanju bo podrobneje opisana izdelana metoda reševanja, ki temelji na optimizaciji.

Postopek za iskanje optimalne razvrstitve kuhalnikov temelji na pregledovanju množice možnih zaporedij. Cilj postopka je čim bolj zmanjšati število možnih zaporedij in sicer na ta način, da pri generiranju zaporedij izhajamo iz zaporedja kuhalnikov, ki predstavlja dopustno rešitev. Dopustno rešitev dobimo s simulacijo obratovanja pri fiksnem času trajanja šarž, množico za pregledovanje pa dobimo s permutacijo zaporedja kuhalnikov dopustne rešitve. Pri permutaciji se omejimo na premikanje kuhalnikov samo za eno mesto v razvrstitvi, kar je smiselno glede na specifične značilnosti problema. Permutacijo tudi omejimo na primere, ko se sosednja kuhalnika v zaporedju praznita v različni zbirni rezervoar. Primer: za osnovno zaporedje kuhalnikov Z = **3** 6 7 **4** 8 je množica možnih rešitev ξ = {6 **3** 7 **4** 8, **3** 6 **4** 7 8, **3** 6 7 8 **4**, 6 **3** **4** 7 8, 6 **3** 7 8 **4**}.

Časovni poteki šarž so pri danem zaporedju lahko različni in od izbrane cenilke za ocenjevanje je odvisno, katere rešitve so bolj zaželene. Časovno razporeditev določimo s pomočjo linearnega programiranja (*LP*). To je pogosto uporabljena statična optimizacijska metoda, ki je primerna tudi za reševanje problemov večjih dimenzij. Z *LP* iščemo vrednosti spremenljivk $x_1, x_2, ..., x_n$, kjer je cenilka linearna funkcija neznanih

spremenljivk, omejitve pa so podane kot linearne enačbe in neenačbe ter omejitve vrednosti iskanih spremenljivk. Standarden zapis za *LP* je naslednji:

$$
\begin{aligned}
&\text{minimiziraj} && z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
&\text{glede na} && a_{11}x_1 + a_{12}x_2 + \ldots a_{1n}x_n = b_1 \\
& && a_{21}x_1 + a_{22}x_2 + \ldots a_{2n}x_n = b_2 \\
& && \vdots \\
& && a_{m1}x_1 + a_{m2}x_2 + \ldots a_{mn}x_n = b_m \\
& && x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0.
\end{aligned}
\tag{7.177}
$$

Za proces kuhanja celuloze so neznane spremenljivke časi začetkov S_i in zaključkov C_i šarž. Podvržene so različnim omejitvam, ki jih definiramo v linearni obliki, kot je predstavljeno v nadaljevanju. V izrazih (7.178)- (7.181) so s t označene konstante, ki predstavljajo številčno vrednost omejitev. Izračunamo jih za vsako šaržo v zaporedju posebej, določene pa so glede na trenutno stanje šarže.

Omejitve

Prednostne omejitve - Na vsakem kuhalniku se mora šarža zaključiti, preden se začne izvajanje naslednje šarže na istem kuhalniku:

$$S_i \geq C_i \quad i = 1, \ldots, n\,. \tag{7.178}$$

Strežne omejitve - Časi začetkov šarž so urejeni v intervalih zaradi omejitve hkratnega polnjenja več kot enega kuhalnika; t_p je čas, ki je potreben za polnjenje enega kuhalnika:

$$S_i \geq S_{i-1} + t_p \quad i = 2, \ldots, n\,. \tag{7.179}$$

Zahteve časa izvajanja - Za zaključek *i*-te šarže je potreben določen minimalni čas t_{si}, v katerem se izvršijo potrebne šaržne operacije (vse ali samo še preostale, če je šarža že v izvajanju):

$$C_i \geq t_0 + t_{si}\,. \tag{7.180}$$

t_0 označuje trenutni čas. Če se i-ta šarža v zaporedju še ni pričela izvajati, čas t_0 nadomestimo z S_j, ki označuje predvideni čas začetka šarže, ki se bo zaključila ob času C_i:

Omejitve kapacitet - Omejitve kapacitet se nanašajo na zbirne rezervoarje. Nivo v rezervoarju se mora zaradi nadaljnjega kontinuirnega poteka proizvodnje gibati med predpisano minimalno in maksimalno vrednostjo. Zato mora biti čas začetka praznjenja kuhalnika, ki se približno ujema s časom zaključka šarže, omejen na naslednji način:

$$t_i^l \le C_i \le t_i^u \quad i = 1,...,n \,. \tag{7.181}$$

kjer t_i^l označuje spodnjo časovno mejo za začetek praznjenja, da nivo po praznjenju kuhalnika ne naraste prek maksimalne vrednosti, t_i^u pa označuje zgornjo časovno mejo, t.j. kdaj se mora začeti praznjenje kuhalnika, da se nivo pred praznjenjem ne zniža pod dopustno mejo. Te časovne meje se izračunajo glede na trenutni nivo v zbirnih rezervoarjih, predvideni pretok praznjenja rezervoarjev in predvideno zaporedje praznjenja kuhalnikov na liniji.

Cenilka

Pri optimizaciji z *LP* smo uporabili cenilko, ki minimizira čase čakanja kuhalnikov pred začetkom nove šarže (S_i-C_i) in absolutna odstopanja časov zaključkov šarž (E_i^+ in E_i^-) od želenih vrednosti C_{ri}. Želene čase za zaključek šarž C_{ri} predhodno izračunamo glede na želeni enakomerni potek nivoja v zbirnih rezervoarjih ali pa glede na enakomerni potek polnjenja kuhalnikov. Problem linearnega programiranja je torej

$$\begin{aligned} &\text{minimiziraj} && \alpha\sum_i (S_i - C_i) + \beta\sum_i (E_i^+ + E_i^-) \\ &\text{glede na} && C_{ri} = C_i - E_i^+ + E_i^- \\ & && E_i^+ \ge 0, E_i^- \ge 0, i = 1,...,n \\ & && \textit{izrazi (7.178) – (7.181)} \end{aligned} \tag{7.182}$$

kjer sta α in β uteži posameznih delov cenilke.

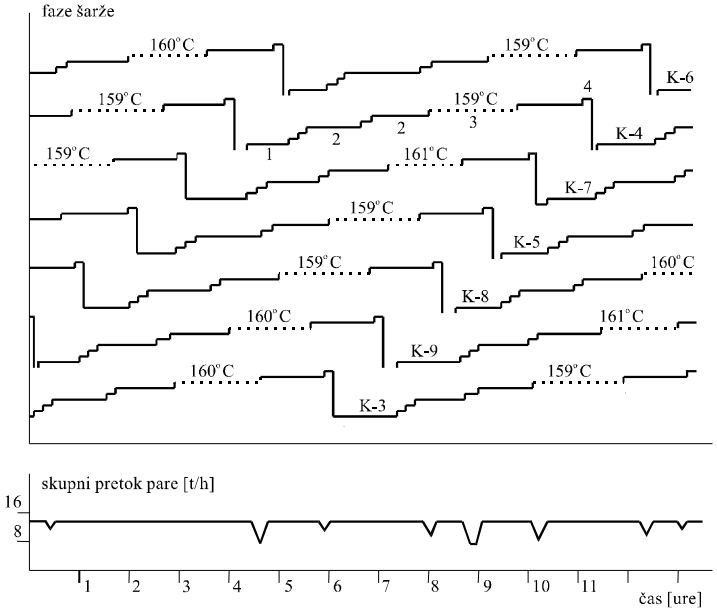
Z *LP* določimo čase začetkov in zaključkov šarž za vsako izmed zaporedij v množici možnih rešitev. Izbran je tisti razpored ali urnik šarž, ki daje čim bolj optimalno rešitev.

Postopek je primeren za sprotno vodenje v realnem času. Po tem postopku se razporejanje šarž pri rednem obratovanju ponavlja periodično v enakih intervalih (npr. vsakih 10 minut) in vsakokrat ob začetku nove šarže na enem od kuhalnikov. Razporejanje se običajno vrši za 10-14 prihodnjih šarž. Zaradi prilagodljive cenilke omogoča poudarjanje različnih ciljev v skladu z zahtevami vodenja.

Na Sl. 7.136 je v obliki stopničastega diagrama prikazan simulacijski primer poteka šaržnih operacij, kjer je razporejanje šarž potekalo po opisanem postopku. V diagramu je dolžina posamezne stopnice premosorazmerna času trajanja ustrezne faze šarže; najpomembnejše faze so označene s številkami od 1 do 4; nad fazo kuhanja pa je vpisana temperatura kuhanja.

Iz diagrama je razvidna enakomerna razporeditev šaržnih operacij, predvsem pa kratko čakanje kuhalnikov (prekinjena črta) pred fazo polnjenja (odebeljena polna črta - faza 1). Prav tako je razvidno, da je s postavitvijo urnika mogoče podaljšati fazo kuhanja s kuhanjem pri nižji temperaturi (maksimalna temperatura je 162°C), ki pomeni tudi

boljšo kvaliteto končnega produkta. Razpoložljivi čas za podaljšanje kuhanja je določen z razporejanjem šarž, kjer čakanje kuhalnika, ki bi po zaključku šarže predvidoma nastopilo zaradi zasedenosti polnilne naprave, pravočasno ugotovimo in zato vnaprej koristneje porabimo za podaljšanje šarže. Kot prikazuje diagram skupnega pretoka pare, pa se z razporejanjem šarž uredi tudi enakomerna poraba pare, ki jo kuhalniki potrebujejo v fazi segrevanja (faza 2).



Sl. 7.136. Časovna razporeditev šarž in posameznih faz šarže po kuhalnikih

Prikazana metoda razporejanja šarž pri procesu kuhanja celuloze je splošno zasnovana in jo je mogoče uporabiti povsod tam, kjer samo določitev zaporedja ni dovolj za rešitev problema.

Δ

7.4 Metode umetne inteligence v vodenju sistemov

Večina metod, ki smo jih predstavili kot osnovo za načrtovanje pomembnejših funkcij vodenja v prejšnjem podpoglavju, bi lahko šteli med tradicionalne metode. Le-te imajo svoje korenine večinoma v tradicionalnih strokah (elektrotehnika, strojništvo, kemijska tehnologija), in jih je nato teoretično razvila in utemeljila teorija vodenja. V zadnjih dveh desetletjih pa so se na področju vodenja uveljavile metode umetne inteligence, ki pa imajo nekoliko drugačne korenine.

V tem podpoglavju bomo nekoliko več poudarka dali dvema področjema, ki sta se uveljavili predvsem v zadnjih letih, to je področje umetnih nevronskih mrež in področje mehke logike, na kratko pa bomo omenili tudi vlogo ekspertnih sistemov, strojnega učenja in genetskih algoritmov pri načrtovanju in realizaciji sistema za vodenje.

Metode umetne inteligence v vodenju sistemov v nekaterih primerih omogočajo doseči načrtovalcu vodenja veliko več kot ustaljeni pristopi. Glavna vprašanja, ki se porajajo pri metodah umetne inteligence v vodenju oziroma pri inteligentnih sistemih vodenja so naslednja:

- Kaj je inteligentnega pri inteligentnih sistemih vodenja?
- Kaj je tista stvar, ki jo inteligentni sistem vodenja omogoča, konvencionalne metode vodenja pa ne?
- Kdaj je primerno razmišljati o uporabi inteligentnih sistemov vodenja?

Da bi lahko odgovorili na zastavljena vprašanja si najprej poglejmo kratek pregled razvoja inteligentnih metod vodenja.

Pojem inteligentnega vodenja je bil uveden že v petdesetih letih. Danes srečujemo ta pojem v raznih pomenskih zvezah in z različnimi vsebinskimi poudarki. Razvoj optimalnega, adaptivnega in robustnega vodenja je bistveno pripomogel k razširitvi koncepta avtomatizacije in povečal možnosti uporabe avtomatskega vodenja v praksi. Napredek v tehnologiji mikroprocesorjev pa je pripeljal do možnosti izvedbe vodenja na vseh nivojih aplikacij. Kljub visokemu nivoju avtomatizacije pa njeno načrtovanje temelji v glavnem na determinističnem opisu sistemov. Konvencionalna teorija vodenja temelji na pojmih kot so: matematični model sistema, zapis z relacijami in Boolova logika. Pri načrtovanju inteligentnega vodenja pa skušamo učinkovito načrtovati vodenje tudi brez omenjenih treh elementov.

Ideja o razvoju avtonomnih naprav ni nova, saj so se z njo ljudje poigravali že pred pojavom digitalnih računalnikov. Jedro raziskovanj danes pa temelji na iskanju najboljšega načina vgradnje znanja v napravo, tako da bo lahko učinkovito "mislila". Na začetku je veljalo prepričanje, da lahko znanje predstavimo samo z graditvijo baze lastnosti o objektih, s katerimi naprava deluje. Šele v devetnajstem stoletju je bilo "znanje" definirano kot mreža trditev, ki se razvijejo na podlagi izjavne in predikatne logike.

V začetku dvajsetega stoletja so raziskovalci preučevali človeški živčni sistem in možgane kot osnovni center vodenja v človeškem telesu. V tridesetih letih so tovrstno raziskovalno delo pričeli imenovati z izrazom kibernetika. Šele v petdesetih letih so raziskovalci kibernetiko pričeli združevati z razvojem računalnikov in na ta način omogočili strojem reševanje geometrijskih problemov, igranje šaha, prepoznavanja vzorcev in govora. Tako se je rodilo tisto, kar danes pogosto imenujemo "umetna inteligenca". Smernice sodobnih raziskovanj inteligentih sistemov vodenja pa so usmerjene predvsem na združitev umetne inteligence in naprednih algoritmov vodenja.

Z razvojem tehnologije so se polega kompleksnosti proizvodnih sistemov povečale tudi naše zahteve, kar povzroča nove in nove izzive razvoju sistemov vodenja. Da bi lahko izpolnila te zahteve, mora imeti industrija boljše in avtonomnejše sisteme vodenja na vseh nivojih (od socio-ekonomskega, preko proizvodnega do transportnega). Tak sistem vodenja naj bi imel optimalen učinek tudi pri večjih odstopanjih v sistemu samem ali v njegovem okolju, ki bi se lahko pojavljala tudi v daljšem časovnem obdobju. Poleg tega naj bi tak sistem kompenziral izpade zaradi okvar brez posredovanja operaterja.

V tem smislu naj bi inteligentni sistem vodenja bil zmožen (Shoreshi, 1995):

- dojemanja stanja sistema kot celote in zaznavanja sprememb notranjih stanj sistema in njegove okolice;
- uporabe kvantitativnih in kvalitativnih modelov obnašanja sistema glede na vhodne signale dobljene preko vhodnih tipal;
- primernega ukrepanja za prilagajanje učinka sistema ali njegovega delovanja za kompenzacijo izpada posameznih komponent zaradi okvare.

Konvencionalni sistemi vodenja delujejo pravilno, samo kadar jim v večji ali manjši meri pomaga človek. Problem nastopi v okoljih, kjer človek zaradi nedostopnega, neprijaznega ali nevarnega okolja ne more delati. Primeri takih okolij, kjer morajo imeti sistemi vodenja veliko stopnjo avtonomije, so na primer: vesolje, globine oceanov in notranjost jedrskih reaktorjev.

Ključno vprašanje, ki se poraja je, ali lahko realiziramo tako inteligentno vodenje, ki bo odpravilo pomanjkljivosti konvencionalnih metod in se obnašalo inteligentno v skladu s prej naštetimi lastnostmi. Čeprav je ideja o inteligentnih napravah relativno stara, pa je šele tehnologija današnjega časa tista, ki je omogočila postavitev temeljev za razvoj inteligentnih sistemov vodenja. Umetne nevronske mreže kot elementi, ki se učijo in vršijo akcijo, mehka logika kot oblika kvalitativnega dojemanja in sistemi vodenja na osnovi pravil (ekspertni sistemi) kot prilagodljivi elementi, so primeri novejših teoretičnih dosežkov, ki omogočajo formulacijo in izvedbo inteligentnih sistemov vodenja. Seveda pa je s strojnega vidika prav ceneni mikroprocesor tisti element, ki omogoča izvedbo omenjenih algoritmov inteligentnega vodenja.

Inteligenca in možne uporabe inteligentnih sistemov vodenja

"Inteligenca" je lastna človekovim sposobnostim in kot taka nekoliko neprimerna za opisovanje sposobnosti sistema vodenja. Bolj primerno poimenovanje za inteligentne sisteme vodenja je avtonomno vodenje. S takim pogledom lahko definiramo naslednje pojme (Shoreshi, 1995):

- inteligenca pomeni določeno stopnjo avtonomnosti;
- avtonomnost pomeni samostojnost in neodvisnost;
- inteligentni sistem vodenja je tisti sistem vodenja, ki ima največjo stopnjo avtonomnosti v smislu sposobnosti učenja, rekonfiguracije, dojemanja, planiranja in sprejemanja odločitev ter sposobnost izločanja le najbolj pomembnih informacij iz

nestrukturiranih in ošumljenih podatkov iz katerega koli kompleksnega dinamičnega sistema in/ali okolice.

Moderni tehnološki sistemi in proizvodne metode vpeljujejo številne zanimive probleme, ki jih ne moremo enostavno rešiti s konvencionalnimi pristopi k vodenju. Primeri takih problemov so:

- pogoste spremembe v postopku izdelave izdelka;
- pogoste spremembe zaradi kombinacij polizdelkov in posebnosti posameznih izdelkov;
- uvajanje naprednejših in zelo kompliciranih proizvodnih metod;
- dvig hitrosti proizvodnje, manj zadrževalnega časa v poteku izdelave;
- lahke mehanske konstrukcije obdelovalnih strojev (resonančni efekti);
- uvajanje celostnih sistemov vodenja, ki vsebujejo poleg osnovnih funkcij vodenja tudi taktične, upravljalske, operacijske, nadzorne in kontrolne funkcije.

Posledica tega je, da so problemi vodenja težje rešljivi zaradi:

- sprememb delovnih točk in okoliščin;
- izrazito nelinearnega in časovno spremenljivega delovanja;
- dviga zahtevanih hitrosti dinamičnih odzivov procesov;
- povečanja kompleksnosti modelov procesov (interakcije, visokofrekvenčni vplivi);
- mešanja kvalitativnega in kvantitativnega znanja o procesu.

Iz tega sledi, da potrebujemo za učinkovito reševanje opisanih problemov popolnoma drugačne in nove pristope, ki bodo dopolnjevali konvencionalne. Pristopi umetne inteligence dajejo možnost za reševanje naštetih problemov na več področjih vodenja sistemov. Ena izmed možnih razdelitev metod je glede na področje njihove uporabe.

Področja uporabe metod in pristopov umetne inteligence pri vodenju sistemov lahko razdelimo na dve večji skupini.

Prva skupina se nanaša na proces načrtovanja in izgradnje sistemov za vodenje. V tem okviru so seveda zelo pomembni postopki priprave različnih alternativ rešitev v različnih fazah življenjskega cikla sistemov za vodenje ter sprejemanje ustreznih odločitev. Tu so nam npr. lahko v veliko pomoč ekspertni sistemi, ki vsebujejo znanja izkušenih strokovnjakov.

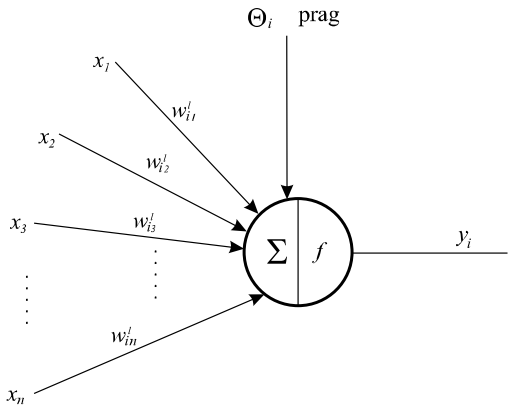
Druga skupina pa se nanaša na "vgrajeno inteligenco" v sam sistem vodenja kot izdelek. Tu gre seveda v principu za to, da posamezne funkcije sistema za vodenje naredimo bolj avtonomne in torej manj odvisne od človekovega posredovanja. Metode umetne inteligence se pojavljajo pri realizaciji praktično vseh pomembnejših funkcij sistemov za vodenje, ki smo jih navedli v poglavju 4. Najbolj znana področja uporabe pa so: obdelava podatkov in ugotavljanje statusa procesa, opreme in proizvoda (npr. odkrivanje napak), določanje želenega delovanja (razvrščanje, optimiranje), priprava ukrepov in odločanje (regulacija) itd.

Pri načrtovanju inteligentnih sistemov vodenja se srečujemo, tako kot pri konvencionalnih metodah načrtovanja vodenja, s celim spektrom metod in pristopov. Poleg tega lahko v literaturi naletimo na zelo različne interpretacije posameznih pristopov oziroma različne vidike, ki posamezne pristope predstavijo kot metodo umetne inteligence ali v kaki drugi luči. Zaradi tega bomo v nadaljevanju predstavili le osnovne ideje najbolj znanih metod, za katere lahko z dokaj gotovosti rečemo, da spadajo na področje umetne inteligence.

7.4.1 Umetne nevronske mreže

Uporaba umetnih nevronskih mrež se je razširila na mnoga področja. Naj omenimo le nekatera kot so npr.: robotika, telekomunikacije, elektronika, bančništvo in mnoga druga. Na teh področjih so nameni uporabe umetnih nevronskih mrež različni, npr.: razpoznavanje vzorcev in dokumentov, procesiranje in analiza signalov, kompresija podatkov ter ne nazadnje tudi za identifikacijo procesov in avtomatsko vodenje.

Osnovna značilnost umetne nevronske mreže (angl. artificial neural network) je, da je sestavljena iz precejšnjega števila preprostih procesnih elementov oziroma nevronov, ki so povezani v mreže. Ti nevroni naj bi bili matematična abstrakcija biološkega nevrona. Procesni elementi so lahko različni pri različnih tipih umetnih nevronskih mrež. Na Sl. 7.137 vidimo shemo procesne enote, ki je skupna mnogim tipom umetnih nevronskim mrež in je osnovni gradnik pri avtomatskem vodenju procesov najpogosteje uporabljane večnivojske mreže – večnivojski perceptron (angl. multilayer perceptron).



Sl. 7.137. Procesni element ali umetni nevron

Poleg večnivojske umetne nevronske mreže poznamo tudi druge vrste umetnih nevronskih mrež. Med najbolj znane sodijo: Hopfieldova, Kohonenova, Gaussova, itd.

Shema procesne enote na Sl. 7.137 je grafični prikaz enačbe

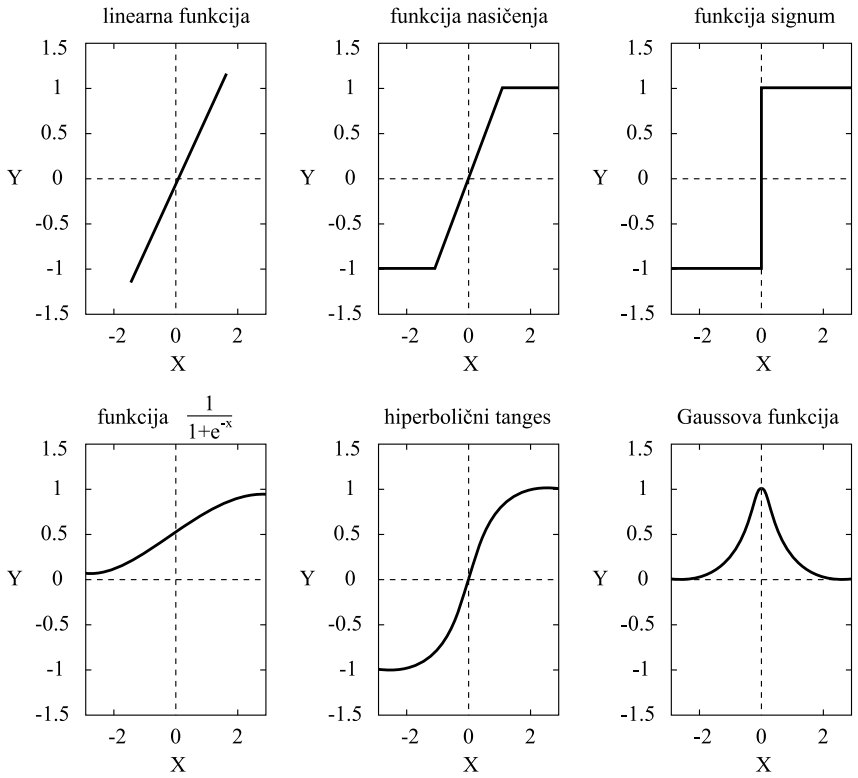
$$y_i(t) = f\left(\sum_{j=1}^{n} w_{ij}^{l}(t)\, x_j(t) + \Theta_i(t)\right)$$

kjer pomenijo:

y_i – izhod i-te procesne enote,
w_{ij} – vrednost uteži j-tega vhoda do i-te procesne enote v l-tem nivoju,
x_j – j-ti vhod,
Θ_i – prag i-te procesne enote,
n-število vhodov procesne enote.

Procesna enota je, kot lahko vidimo na Sl. 7.137, sestavljena iz dveh delov:

- izhodne funkcije procesnega elementa f (desni del procesnega elementa na Sl. 7.137);
- funkcije aktivacije (levi del procesnega elementa na Sl. 7.137).



Sl. 7.138. Pogosteje uporabljane izhodne funkcije v umetnih nevronskih mrežah

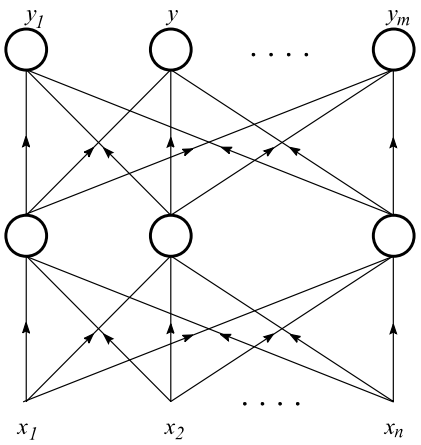
Izhodna funkcija je pri večnivojski nevronski mreži vedno enaka, izberemo pa jo sami največkrat med funkcijami sigmoidne oblike. Sigmoidna funkcija je zvezna, monotono

naraščajoča ter navzgor in navzdol omejena.

Sl. 7.138 prikazuje primere najpogosteje uporabljenih izhodnih funkcij.

Večnivojska nevronska mreža je, kot že samo ime pove, sestavljena iz več nivojev in je razširitev enonivojske mreže. Vsak nivo je sestavljen iz več procesnih enot, ki med seboj niso povezane. Vsi izhodi procesnih enot so povezani z vsemi vhodi procesnih enot naslednjega nivoja. Zadnji nivo imenujemo izhodni nivo.

Izhodi procesne enote tega nivoja so tudi izhodi umetne nevronske mreže. Vsaki povezavi med dvema procesnima enotama ali med vhodom in procesno enoto priredimo utež. Topologijo večnivojske nevronske mreže prikazuje Sl. 7.139.



Sl. 7.139. Topologija dvonivojske nevronske mreže

Matematično gledano opravi umetna nevronska mreža preslikavo iz realnega prostora dimenzije n v realni prostor dimenzije m. Enonivojska nevronska mreža je sposobna opravljati le linearne preslikave, večnivojska pa tudi nelinearne. Da dosežemo želeno preslikavo, morajo biti uteži pravilno določene. Uteži izračunamo po algoritmu, ki ga določa pravilo učenja. Poznamo tri skupine pravil učenja. V prvo skupino uvrščamo vsa pravila, ki za vsak vhodni vzorec zahtevajo poznavanje pravilnega izhoda. Tako učenje imenujemo nadzorovano učenje (angl. supervised learning). Drugo skupino sestavljajo pravila, ki ne zahtevajo poznavanja pravilnega izhoda umetnih nevronskih mrež, ampak lahko damo le oceno kvalitete dejanskega izhoda. Tako učenje imenujemo ocenjevano učenje (angl. graded reinforcement learning). Pri prvih dveh skupinah učenja gre za optimizacijo uteži, da bi dosegli čim manjšo razliko med dejanskim in zaželenim (oz. pravilnim) izhodom nevronske mreže. V tretjo skupino pravil učenja uvrščamo samoorganizacijska pravila, kjer sploh ni potrebno poznavanje pravilnega izhoda za dane vhode. Gledano s stališča preslikav to pomeni, da preslikavo določi nevronska mreža sama. To skupino zelo redko uporabljamo v sistemih vodenja.

V primeru, da uporabimo umetne nevronske mreže v vlogi regulatorja, ne poznamo

vnaprej njenega izhoda (regulirnega signala). Poznamo le želeni izhod procesa. Torej ne moremo direktno uporabiti nadzorovanega učenja. Tudi samoorganizacijska pravila niso primerna, saj želimo določeno preslikavo vhoda (napake) v izhod regulatorja. Ta preslikava je sicer neznana (ne poznamo želenega regulirnega signala), vendar je določena s tem, da želimo doseči čim manjšo napako. Na prvi pogled je najprimernejše ocenjevano učenje, saj lahko podamo neko oceno o uspešnosti regulatorja glede na razliko med dejanskim in želenim izhodom procesa. Žal pa je v splošnem ocenjevano učenje dolgotrajnejše in manj učinkovito od nadzorovanega.

Umetne nevronske mreže se uporabljajo za razne oblike vodenja na osnovi modela procesa, vendar jih še pogosteje srečujemo pri identifikaciji nelinearnih sistemov.

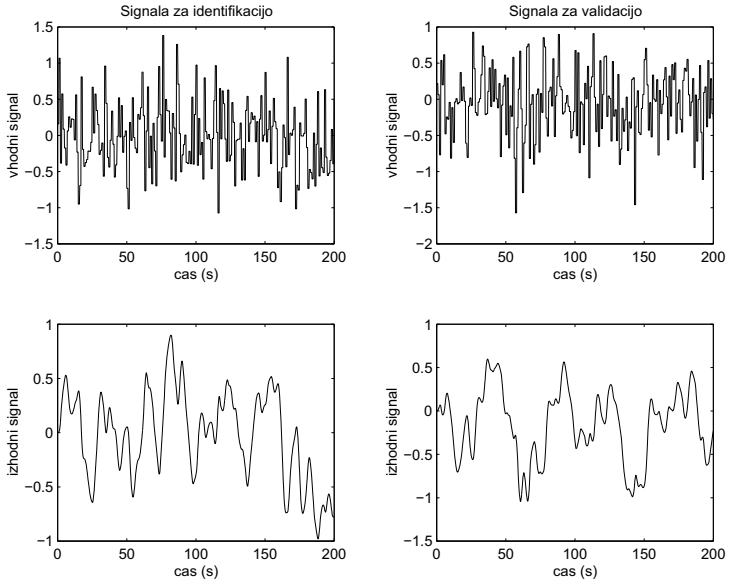
Pri identifikaciji nelinearnih sistemov s postopkom učenja nastavimo uteži nevronske mreže tako, da izhod nevronske mreže čim bolj sledi dejanskemu izhodu procesa. Omenili smo že, da opravi umetna nevronska mreža preslikavo funkcije iz enega v drug realni prostor. Pri tem gre za preslikavo statične funkcije. Da pa bi umetno nevronsko mrežo uporabili za identifikacijo dinamičnih sistemov, nas zanimajo preslikave dinamičnih funkcij. Dinamiko vpeljemo v umetne nevronske mreže tako, da iz njenega izhoda pripeljemo na vhod različno zakasnjene signale. Podatki o modelu so strukturni (zakasnjeni vhodni in izhodni signali, število nivojev nevronske mreže, število nevronov v posameznih nivojih) in parametrični (izhodne funkcije posameznih nevronov in vrednosti uteži). Prednost uporabe umetne nevronske mreže za identifikacijo nelinearnih sistemov je v tem, da lahko z njo aproksimiramo poljubno nelinearno funkcijo. Način izbora strukturnih parametrov je prepuščen izkušnjam človeka, čeprav so v zadnjem času razvili nekaj postopkov, ki delo olajšajo. Tudi metode učenja oziroma optimizacije so se izboljšale, in niso več tako časovno potratne, kot je metoda povratnega učenja (Matko, 1995).

∇

Primer 7.23: *Identifikacija procesa treh shranjevalnikov z umetnimi nevronskimi mrežami*

Čeprav se umetne nevronske mreže praviloma uporabljajo za identifikacijo nelinearnih procesov, bomo iz ilustrativnih razlogov prikazali primer identifikacije z umetno nevronsko mrežo na linearnem procesu treh shranjevalnikov (Sl. 7.10).

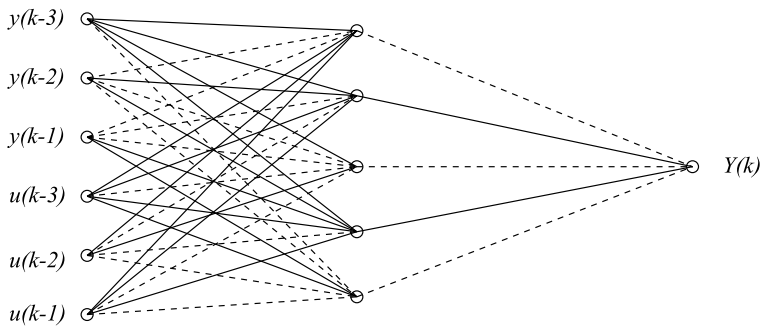
Najprej izberemo dva frekvenčno primerno bogata vhodna signala, in sicer enega za identifikacijo in drugega za vrednotenje modela procesa. Nato izvedemo eksperiment na procesu ter dobimo ustrezni par izhodnih signalov. Sl. 7.140 prikazuje signala za identifikacijo in vrednotenje modela procesa.



Sl. 7.140. Signali za identifikacijo in vrednotenje modela procesa

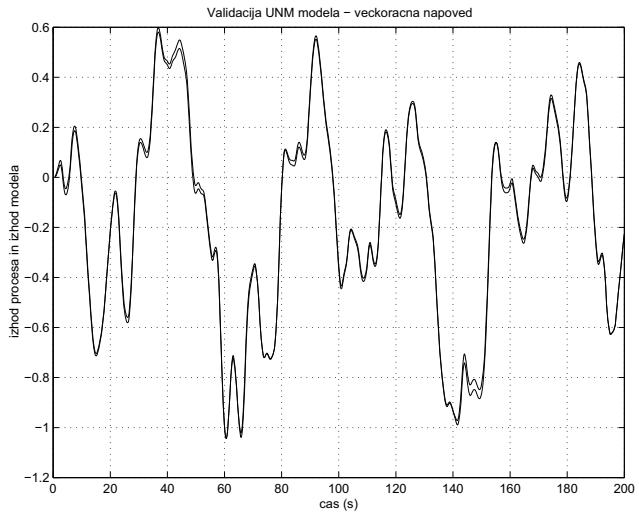
Nato vhodni in izhodni signal za identifikacijo uporabimo v postopku razvoja nevronske mreže.

Strukturo nevronske mreže moramo izbrati zelo skrbno, da bi lahko učinkovito določili parametre. Strukturo prikazuje Sl. 7.141. Kot lahko vidimo, smo za vhod v nevronsko mrežo izbrali zakasnjene vhodne in izhodne signale (za en, dva in tri vzorce). Uteži mreže smo določili (mrežo smo naučili) z Levenberg-Marquardtovo modifikacijo Gauss-Newtonove optimizacijske metode. Rezultate vrednotenja tako dobljenega modela prikazuje Sl. 7.142.



Sl. 7.141. Struktura uporabljene nevronske mreže

Vidimo, da se izhod procesa in izhod modela (nevronske mreže) dovolj dobro ujemata.



Sl. 7.142. Vrednotenje modela procesa

Δ

7.4.2 Mehka logika in mehko vodenje

Mehka logika je v zadnjih desetih letih pritegnila močno pozornost, čeprav so bili njeni teoretični temelji podani že v šestdesetih letih. Mehka logika pomeni razširitev klasične logike in je bila razvita zaradi kvalitativnega opredeljevanja informacije. V mehki logiki z veljavnostjo neke izjave ne definiramo samo popolne veljavnosti ali pa popolne neveljavnosti, kot je to v primeru klasične logike, temveč tudi vmesne stopnje veljavnosti. Torej stopnje med popolno veljavnostjo in popolno neveljavnostjo. Na ta način uvedemo mehkost v dojemanje informacije. Z vidika sistemske teorije je bistvo mehke logike v zapisu preslikave med vhodnim in izhodnim prostorom. To preslikavo zapišemo z množico **IF-THEN** pravil. S pomočjo zapisa v obliki pravil definiramo relacijo med vhodnimi spremenljivkami na **IF** strani in izhodnimi spremenljivkami na **THEN** strani pravil.

Mehka logika se uporablja predvsem za načrtovanje tako direktnih kot nadzornih sistemov vodenja, nepogrešljiva pa je tudi pri identifikaciji nelinearnih sistemov.

7.4.2.1 Elementi mehke logike

V nadaljevanju bodo predstavljeni najpomembnejši elementi mehke logike, ki so pomembni tudi pri mehkem vodenju.

Mehke množice

Tako kot obstaja močna povezava med klasično logiko in teorijo množic, obstaja tudi močna povezava med mehko logiko in mehkimi množicami. Osnovno idejo koncepta

mehke logike predstavlja pojem *mehke množice*, zato bomo v nadaljevanju govorili o osnovnih lastnostih mehkih množic.

Klasične oz. ostre množice se odlikujejo po tem, da lahko za vsak objekt določimo, ali je element množice ali pa ne. Pripadnost objekta neki množici lahko zavzame le dve vrednosti, t.j. 0 ali 1. Podmnožica univerzuma S, ki jo označimo z U je definirana kot preslikava med elementi podmnožice S v elemente množice $\{0,1\}$

$$U: \ S \rightarrow \{0,1\} \tag{7.183}$$

Preslikavo lahko predstavimo v smislu urejenih parov. Prvi element urejenega para je element množice S in drugi element je element množice $\{0,1\}$.

Kritika koncepta klasičnih množic, ki naj bi bil za praktično uporabo preidealen, ker so številne ugotovitve v praksi dokaj ohlapne, je pospešil razvoj koncepta mehkih množic. Večkrat uporabljeni primer je primer starosti ljudi. Če želimo razvrstiti ljudi glede na njihovo starost v ostre množice, bi morali določiti ostro mejo, ko človek ni več mlad. Delitev je nesmiselna, saj pripelje do absurda, ko lahko ugotovimo, da se človek v enem dnevu postara. Mehko množico lahko opredelimo z naslednjo definicijo:

Če je X domena vhodne spremenljivke in so njeni elementi označeni z x potem lahko mehko množico A znotraj X definiramo kot množico urejenih parov

$$A=\{ x, \mu_A (x) \mid x \in X\} \tag{7.184}$$

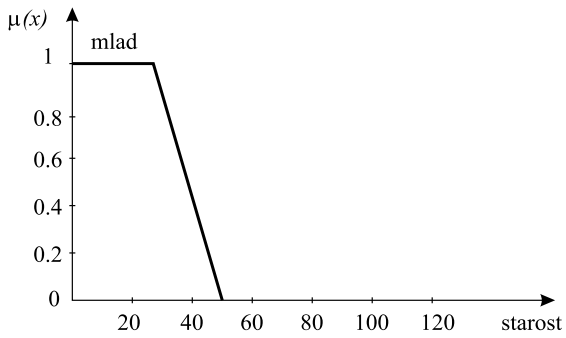
$\mu_A (x)$ ***imenujemo pripadnost elementa x mehki množici A.***

Mehka logika pravilnost vsake izjave definira kot neko relativno stopnjo. Z drugimi besedami lahko rečemo, da mehka logika na enostavna vprašanja dovoljuje odgovore, ki niso ne popolna zanikanja in ne popolna pritrjevanja. Na ta način pravzaprav komunicirajo ljudje pri vsakdanjih pogovorih. Lahko rečemo, da je mehka logika posplošenje klasične logike.

Pripadnostne funkcije

Pripadnostna funkcija je funkcija, ki definira vsaki točki vhodnega prostora pripadajočo pripadnost v obliki realnega števila med spodnjo mejo pripadnosti in zgornjo mejo pripadnosti. Pripadnostne funkcije so navadno navzdol in navzgor omejene z 0 in 1. Pripadnostne funkcije so lahko najrazličnejših oblik od trapezoidne, trikotne, gaussove, uporabniško definirane, itd. Pripadnostna funkcija mora zadostovati pogoju, ki predpisuje, da ležijo pripadnosti na intervalu med 0 in 1, je pa lahko poljubne oblike in jo definiramo tako, da je preprosta, učinkovita glede na problem in računsko nezahtevna.

Uporaba določene oblike pripadnostne funkcije je največkrat odvisna od narave problema. Tipičen primer pripadnostne funkcije za množico mlad je prikazan na Sl. 7.143.



Sl. 7.143. Pripadnostna funkcija za množico mlad

V tem primeru predstavlja domeno vhodne spremenljivke potencialna starost med 0 in 120 let. Mehka logika definira zvezno pripadnostno funkcijo, ki omogoča zvezni prehod med vrednostjo mlad in star.

Tudi nad mehkimi množicami so definirane operacije, ki jih poznamo iz klasične teorije množic. Te operacije preidejo v mejnem primeru, to je v primeru, ko omejimo pripadnost množicam le na 0 in 1, v klasične operacije nad množicami.

Mehki operatorji

V mehki logiki lahko izjavo *A AND B* zapišemo s funkcijo **min**(A,B). Na enak način lahko izjavo *A OR B* zapišemo s funkcijo **max**(A,B). Končno lahko operacijo *NOT A* v mehki logiki zapišemo kot 1-A. Ob uvedbi zgornjih mehkih operatorjev lahko ugotovimo, da pravilnostne tabele ostanejo enake kot v primeru klasičnih operatorjev.

Splošno definiramo mehke logične operatorje na naslednji način: mehki presek ali konjunkcija (AND), mehka unija ali disjunkcija (OR) in mehki komplement (NOT). Zgoraj definirane mehke operatorje imenujemo tudi *klasični mehki operatorji* (AND=min, OR=max, NOT=aditivni komplement). Presek dveh mehkih množic A in B je definiran s funkcijo T na naslednji način

$$\mu_{A\cap B}(x) = T(\mu_A(x), \mu_B(x)) = \mu_A(x) \otimes \mu_B(x) \qquad (7.185)$$

kjer je $\otimes$ binarni operator funkcije T. Splošni operator preseka imenujemo *triangularna* ali *trikotna norma* ali *T-norma*. Unija dveh mehkih množic A in B je definirana s funkcijo S na naslednji način

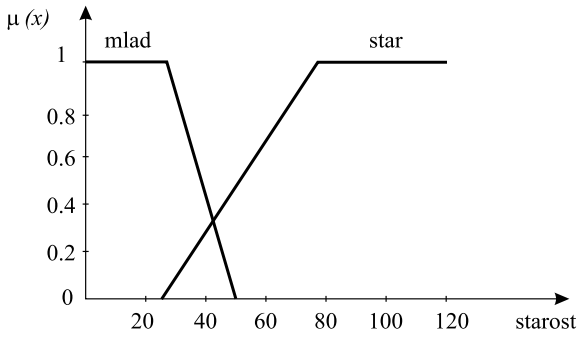
$$\mu_{A\cup B}(x) = S(\mu_A(x), \mu_B(x)) = \mu_A(x) \oplus \mu_B(x) \qquad (7.186)$$

kjer je $\oplus$ binarni operator funkcije S. Splošni operator unije imenujemo *triangularna* ali *trikotna konorma* ali *S-norma*.

Lingvistične spremenljivke

Mehke množice uvajajo pojem *lingvistične spremenljivke,* ki lahko zavzema določeno

lingvistično vrednost. V prejšnjem primeru lahko definiramo lingvistično spremenljivko **starost**, ki ima dve vrednosti, *mlad* in *star*. Lingvistična spremenljivka je definirana na določeni domeni, pripadajo ji določene lingvistične vrednosti in njihova interpretacija v obliki mehkih množic oz. pripadnostnih funkcij, kar prikazuje Sl. 7.144.



*Sl. 7.144. Lingvistična spremenljivka **starost***

Lingvistične spremenljivke imajo popolnoma enako funkcijo kot konvencionalne spremenljivke, lahko jim pripišemo neko vrednost, lahko poizvemo po njeni vrednosti, njeno vrednost lahko primerjamo z ostalimi vrednostmi.

IF-THEN pravila

Mehke množice, lingvistične spremenljivke in mehki operatorji tvorijo osnovne elemente mehke logike. Za izmenjavo informacije je potrebno oblikovati izjave v smislu stavkov. Pogojni stavki v obliki **IF-THEN** pravil dajo smisel mehki logiki. Osnovno pravilo mehkega sklepanja lahko zapišemo

***IF** x **IS** A **THEN** y **IS** B*

kjer sta *A* in *B* lingvistični vrednosti definirani v obliki mehkih množic na domeni spremenljivk X in Y. Pogojni del ***IF** x **IS** A* imenujemo *premisa*, ***THEN*** del *y **IS** B* pa *konsekvenca.*

Primer pravila je lahko naslednji

***IF** paradižnik **IS** rdeč **THEN** paradižnik **IS** zrel*

Mehka pravila lahko interpretiramo na naslednji način: najprej se izvede premisa, kjer se realizira *mehčanje* (fuzzification) vhodne spremenljivke, nato pa se ob upoštevanju mehkih operatorjev v drugem koraku tvori posledica pravila, ki ji rečemo *implikacija*.

Premisa je lahko sestavljena iz več delov, kot na primer

***IF** nebo **IS** sivo **AND** veter **IS** močan **THEN** . . .*

V tem primeru se premise izračunavajo istočasno in rezultirajo v eni sami vrednosti rezultirajoče premise. Pri tem uporabljamo mehke logične operatorje, ki so elementi premise.

Tudi posledični del je lahko sestavljen iz več delov, kot na primer:

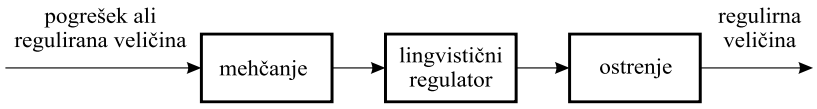
***IF** temperatura_vode **IS** nizka **THEN** ventil_topli **IS** odprt **AND** ventil_hladni **IS** zaprt*

kjer premisa enako vpliva na vsakega od delov v konsekvenčnem delu. Konsekvenčni del definira izhodno mehko množico. Preslikava med pogojnim in posledičnim delom je realizirana s pomočjo funkcije *implikacije*. Najpogosteje uporabljena metoda implikacije za izračun izhodne mehke množice je metoda minimuma (**min**) ali pa metoda produkta (**prod**).

7.4.2.2 Mehka regulacija

Mehka logika je dosegla največji razvoj na področju vodenja. Najprej so se pojavili mehki regulatorji, ki izhajajo iz konvencionalnih linearnih PI, PD in PID regulatorjev. Glede na obliko izhoda jih razdelimo na mehke regulatorje Mamdanijevega tipa in mehke regulatorje Sugenovega tipa. Pri regulatorjih Mamdanijevega tipa je izhod regulatorja v obliki lingvistične spremenljivke, v primeru regulatorjev Sugenovega tipa pa je izhod regulatorja ostra vrednost, kar dodatno poenostavi sintezo. Kasneje pa so se pojavili posebni mehki regulatorji, kot posledica novih rešitev v teoriji mehke logike. Razvili so tudi samonastavljive ali samoorganizirajoče regulatorje, ki jih lahko uvrstimo na področje adaptivnega vodenja.

Strukturo osnovnega mehkega regulatorja prikazuje Sl. 7.145.



Sl. 7.145 Strukturna shema mehkega regulatorja.

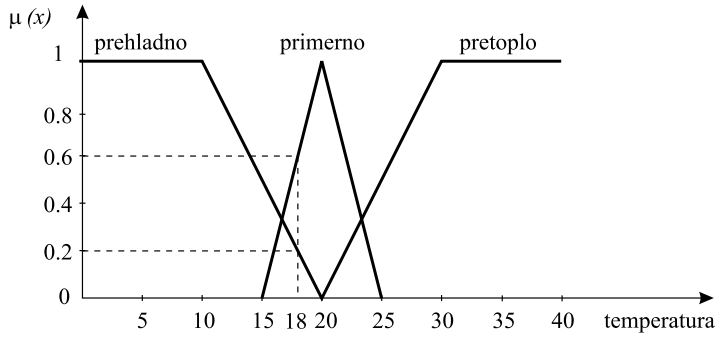
Mehki regulator je sestavljen iz naslednjih blokov: mehčanje ostrih spremenljivk; inferenčni mehanizem, ki na osnovi mehke logike tvori preslikavo med vhodnimi ter izhodnimi mehkimi spremenljivkami; in ostrenje mehkih spremenljivk. Ostre spremenljivke tvorijo vhod in izhod iz regulatorja, torej njegovo povezavo z zunanjim svetom. V nadaljevanju bodo podrobneje predstavljeni bistveni sestavni deli mehkega regulatorja na primeru regulacije temperature v prostoru.

Mehki regulatorji tipa P

Vhod v regulator predstavlja merjena temperatura prostora. Prvi korak v celotnem sklepanju je določanje pripadnosti posamezne vhodne spremenljivke določeni mehki množici. Vhodna spremenljivka je ostre vrednosti in je na intervalu domene

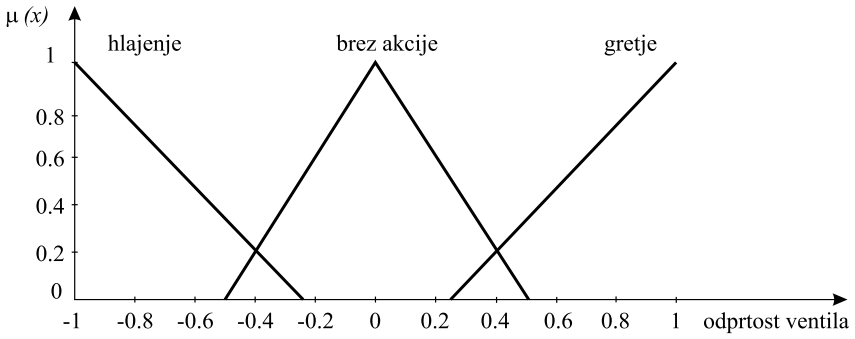
spremenljivke [0,40°C], izhodna vrednost po mehčanju pa je mehka vrednost in sicer vrednost pripadnosti $\mu \in [0,1]$. V procesu mehčanja priredimo ostri vrednosti (npr. 18 °C) lingvistično spremenljivko **temperatura** z vrednostmi *prehladno, primerno* in *pretoplo*. Lingvistična spremenljivka **temperatura** je predstavljena na Sl. 7.146.

Ustrezne pripadnostne funkcije izberemo v postopku načrtovanja regulatorja in so lahko trikotne, trapezne, gaussove, uporabniško definirane, itd.. V našem konkretnem primeru zavzema lingvistična spremenljivka **temperatura** vrednost *prehladno* s pripadnostjo 0.2, vrednost *primerno* s pripadnostjo 0.6 in vrednost *pretoplo* s pripadnostjo 0.



*Sl. 7.146. Lingvistična spremenljivka **temperatura** in njene pripadnostne funkcije*

Lingvistična spremenljivka **temperatura** predstavlja vhod v inferenčni mehanizem, njegov izhod pa je lingvistična spremenljivka **dovod_toplote**, ki lahko zavzame vrednosti *hlajenje, brez_akcije* in *gretje.* Pripadnostne funkcije lingvistične spremenljivke **dovod_toplote** prikazuje Sl. 7.147. Domena spremenljivke **odprtost_ventila** je od -1 do +1, pri čemer pomeni -1 maksimalno odprt ventil za hlajenje in +1 maksimalno odprt ventil za gretje, pri vrednosti 0 sta oba ventila zaprta.



*Sl. 7.147. Lingvistična spremenljivka **dovod_toplote***

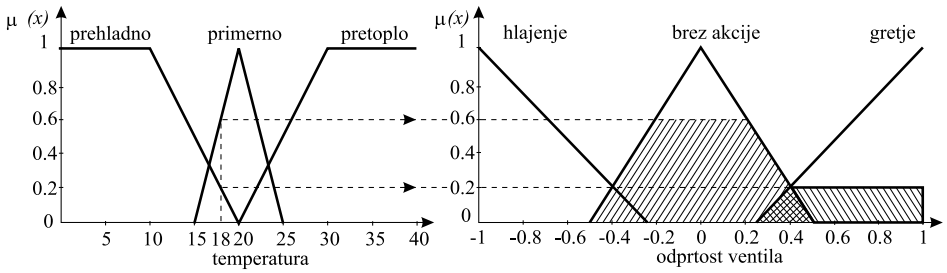
Enostaven mehki regulator za gornji primer regulacije temperature sobe ima naslednji inferenčni mehanizem:

IF **temperatura** ***IS*** *prehladno* ***THEN*** **dovod_toplote** ***IS*** *gretje*

IF **temperatura** ***IS*** *primerno* ***THEN*** **dovod_toplote** ***IS*** *brez_akcije*

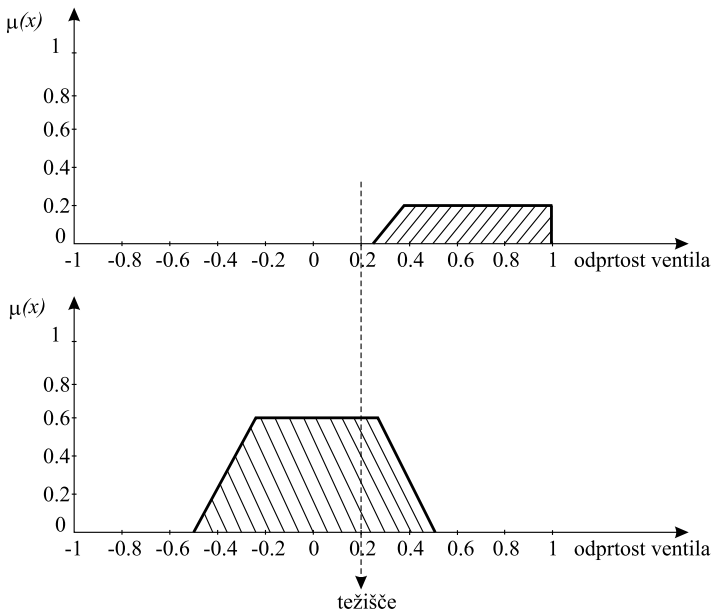
IF **temperatura** ***IS*** *pretoplo* ***THEN*** **dovod_toplote** ***IS*** *hlajenje*

V tem primeru določimo vrednost lingvistične spremenljivke **dovod_toplote** na način kot to prikazuje Sl. 7.148.



Sl. 7.148. Inferenčni mehanizem določevanja lingvistične spremenljivke ***dovod_toplote***

Ker je pripadnost spremenljivke **temperatura** vrednosti *prehladno* enaka 0.2, prispeva k spremenljivki **dovod_toplote** le del vrednosti *greti* s pripadnostjo 0.2 (šrafirana površina). Prispevki vseh pravil tvorijo mehko lingvistično spremenljivko **dovod_toplote**. Zadnji del mehkega regulatorja je ostrenje. V tem delu je potrebno prirediti lingvistični spremenljivki **dovod_toplote** dejansko vrednost položaja ventila (oziroma ventilov za gretje in hlajenje), to je ostro vrednost izhodne spremenljivke regulatorja.



Sl. 7.149. Ostrenje lingvistične spremenljivke ***dovod_toplote***

Najpogostejša metoda ostrenja je težiščna metoda, kjer ostri vrednosti ustreza kar težišče vseh odrezanih pripadnostnih funkcij lingvistične spremenljivke na izhodu regulatorja, kar prikazuje Sl. 7.149.

Mehki regulatorji tipa PI

Obravnavani regulator predstavlja najenostavnejšo varianto temperaturnega regulatorja in predstavlja pravzaprav nelinearno statično karakteristiko ali nelinearni P (proporcionalni) regulator, saj ne upošteva dinamike regulirane temperature. Če upoštevamo spremembo temperature, dobimo nelinearni regulator tipa PD (proporcionalno diferencirni). V ta namen je potrebno vpeljati novo spremenljivko na vhodu regulatorja, to je verbalno spremenljivko **Δ_temperature**. Oba omenjena tipa regulatorjev ne omogočata odprave pogreška v ustaljenem stanju. Pogrešek v ustaljenem stanju lahko odpravimo z regulatorjem tipa I (integrirni) oziroma PI (proporcionalno-integrirni). I oziroma PI regulator dobimo tako, da integriramo izhoda P oziroma PD regulatorja, to pomeni, da izhod regulatorja ne pomeni položaja ventila, temveč spremembo položaja. V mehkem regulatorju moramo zato lingvistično spremenljivko **dovod_toplote** zamenjati s spremenljivko **Δ_dovoda_toplote**. Tudi pravila se v tem primeru ustrezno spremenijo, saj imamo v vzročnem delu dve lingvistični spremenljivi, v posledičnem delu pa eno lingvistično spremenljivko

IF* temperatura *IS* *T* *AND* *Δ*_temperature *IS* *ΔT* *THEN* *Δ*_dovod_toplote *IS* *ΔQ

Obravnavani algoritem mehkega regulatorja ima veliko prostostnih stopenj, ki jih je

potrebno določiti pri njegovi sintezi. Prostostne stopnje mehkega regulatorja so naslednje: vhodne in izhodne vrednosti, vrednosti lingvističnih spremenljivk, oblika in vrednosti pripadnostnih funkcij, pravila in metoda ostrenja. Pri načrtovanju mehkega regulatorja si pomagamo predvsem s simulacijo, s pomočjo katere lahko preizkusimo delovanje regulatorja.

Uporaba mehkih regulatorjev je smiselna v primeru procesov, za katere ne obstajajo ustrezni modeli ali pa je modeliranje zahtevno in drago in pa v primeru nelinearnih procesov. Zelo so primerni, ko skušamo izkušnje operaterja, ki vodi določen proces ročno, prenesti v algoritmizirano obliko, ker je logika mehkega regulatorja prilagojena načinu človekovega razmišljanja.

∇

Primer 7.24: *Mehko vodenje nivoja v rezervoarju*

Sistem sestoji iz črpalke, ki poganja tekočino v rezervoar, regulacijskega ventila, s pomočjo katerega reguliramo dovod tekočine v rezervoar (območje ventila je med 0 in 1), krogelnega rezervoarja z radijem (R=1m) in izpustnega ventila. Prikazan je na Sl. 7.150.

Matematično modeliranje temelji na enačbi masnega ravnotežja:

$$\Phi_{vh} - \Phi_{izh} = \frac{dV}{dt} = S(h)\frac{dh}{dt} \tag{7.187}$$

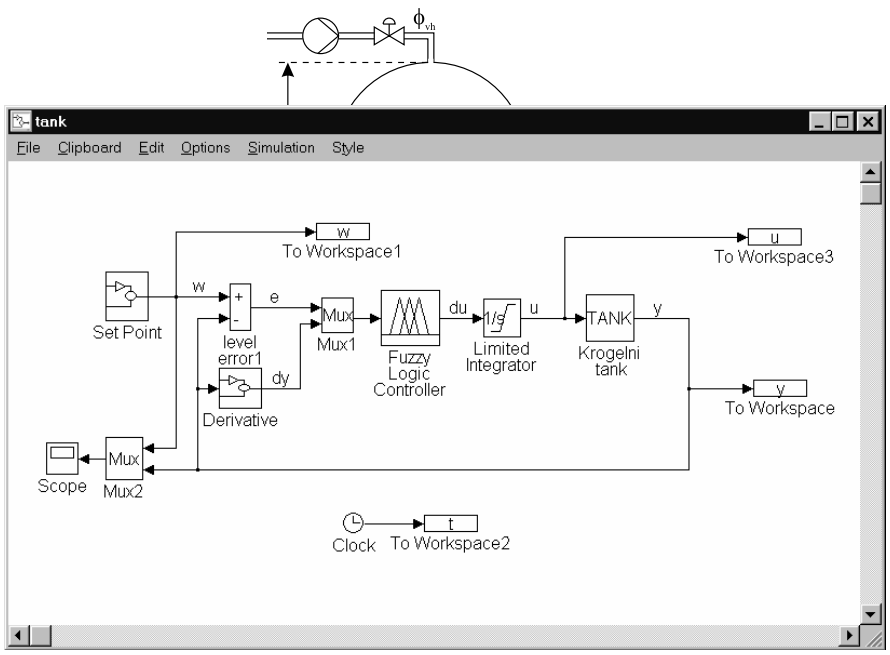
kjer je Φ_{vh} vhodni volumski tok, Φ_{izh} izhodni volumski tok, $S(h)$ površina tekočine kot funkcija višine tekočine v rezervoarju in h višina tekočine v rezervoarju. Površino kot funkcijo višine nivoja tekočine lahko zapišemo z naslednjo enačbo:

$$S(h) = \pi\,(2Rh-h^2) \tag{7.188}$$

Izhodni volumski tok Φ_{izh} je odvisen od višine nivoja tekočine in ga zapišemo z enačbo:

$$\Phi_{izh} = c_p\sqrt{2\cdot g\cdot h} \tag{7.189}$$

Vhodni volumski tok Φ_{vh} pa je odvisen od položaja ventila, ker predpostavimo konstanten pretok na izhodu iz črpalke.



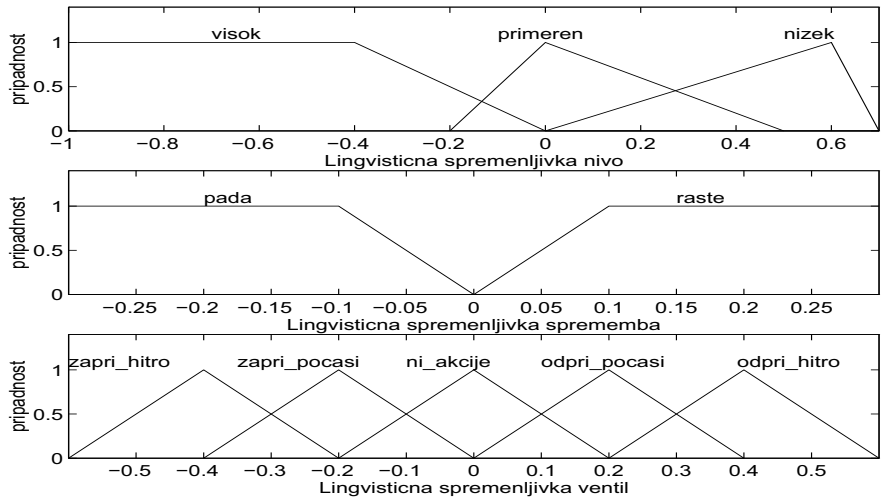
Sistem izkazuje močno nelinearno obnašanje. Nelinearnost je posledica korenske odvisnosti izhodnega pretoka Φ_{izh} in kvadratične odvisnosti površine tanka $S(h)$ glede na višino nivoja tekočine v tanku. Nelinearni model tanka je podan v simulacijski shemi z blokom **TANK** (Sl. 7.151).

Shema, ki je prikazana na Sl. 7.151 prikazuje mehki regulator, ki ima za vhod regulacijski pogrešek e in spremembo izhodnega signala Δy. Izhod regulatorja je sprememba ventila Δu. Mehki regulator načrtamo s pomočjo paketa ANFIS v okolju Matlab-Simulink.

Regulator naj ima dva vhoda in en izhod. Prva vhodna spremenljivka je pogrešek e in je definirana na območju med -1 in 0.75. Imenujemo jo ***nivo*** in je razdeljena na 3 pripadnostne funkcije, ki jih poimenujemo z vidika nivoja tekočine v rezervoarju. Druga vhodna spremenljivka je odvod izhodnega signala Δy (v primeru konstantne reference je Δy enak Δe) in je definirana na območju med -0.28 in 0.28. Razdeljena pa je na 2 pripadnostni funkciji in jo imenujemo ***sprememba***. Izhodna spremenljivka Δu je definirana na intervalu med -0.6 in 0.6 in je razdeljena na 5 pripadnostnih funkcij. Imenujemo jo ***ventil***. Pripadnostne funkcije medsebojno povezujejo štiri pravila. Prvi dve pravili imata po en vhod in en izhod. Drugi dve pravili pa po dva vhoda, ki sta med seboj povezana z ***AND*** operatorjem in en izhod.

Sl. 7.151. Shema mehkega sistema vodenja

Pripadnostne funkcije vseh treh linguističnih spremenljivk so prikazane na Sl. 7.152.



Sl. 7.152. Prikaz lingvističnih spremenljivk

Pravila s katerimi opišemo mehki regulator so pa naslednja:

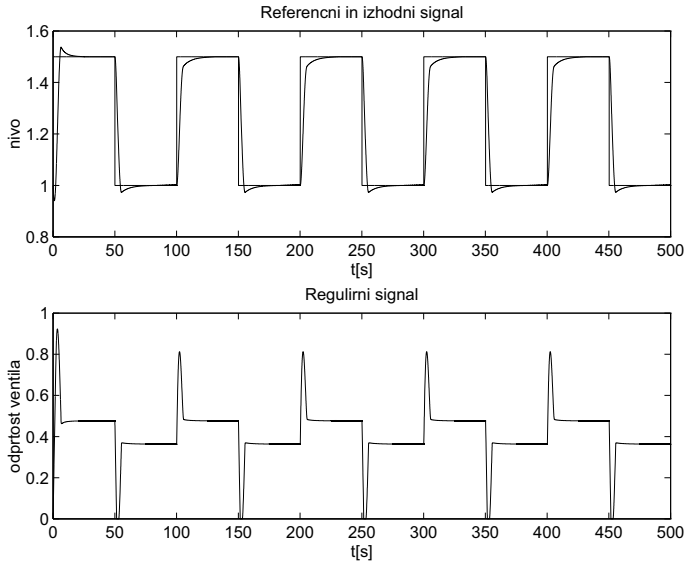
IF nivo IS *nizek* ***THEN ventil IS*** *odpri_hitro*

IF nivo IS *visok* ***THEN ventil IS*** *zapri_hitro*

IF nivo IS *primeren* ***AND sprememba IS*** *pada* ***THEN ventil IS*** *odpri_pocasi*

IF nivo IS *primeren* ***AND sprememba IS*** *raste* ***THEN ventil IS*** *zapri_pocasi*

Rezultati simulacije so prikazani na Sl. 7.153.



Sl. 7.153. Mehko vodenje nivoja tekočine v okroglem rezervoarju

Na sliki sta prikazana referenčni in regulirani signal (potek nivoja) in pa regulirni signal (odprtost ventila). Ugotovimo lahko, da se dinamiki polnjenja in praznjenja bistveno razlikujeta. Glede na nelinearnost, ki se pojavlja v procesu je mehko vodenje zelo smiselno, saj lahko s pomočjo mehkega regulatorja ta problem laže obvladamo.

Δ

7.4.3 Ekspertni sistemi, strojno učenje in genetski algoritmi

V prejšnjem delu tega podpoglavja smo si nekoliko podrobneje ogledali problematiko nevronskih mrež in pa mehke logike s poudarkom na mehki regulaciji. Gre za dva pristopa, ki sta se na področju vodenja nedvomno že zelo uveljavila. V nadaljevanju pa si na kratko poglejmo še tri pristope, ki so prav tako zanimivi, vendar pa njihova uporaba še ni tako zelo razširjena.

7.4.3.1 Ekspertni sistemi

Operatersko vodenje kompleksnih dinamičnih sistemov kot sta na primer letalo ali žerjav zahteva veščino, ki jo operater pridobi z izkušnjami. Kadar to veščino prenesemo v napravo v obliki nekih pravil ponavadi govorimo o ekspertnih sistemih. Ekspertni sistemi temeljijo na simbolični umetni inteligenci, ki se loči od metod, ki temeljijo neposredno na vrednostih signalov kot so na primer umetne nevronske mreže.

Ekspertni sistemi se pri vodenju sistemov uporabljajo na več načinov, vendar v vsakem primeru predstavljajo nadomestilo za veščine človeka. Neposredno za vodenje lahko uporabimo ekspertni sistem za direktno ali indirektno obliko vodenja. Direktna oblika

vodenja pomeni, da ekspertni sistem opravlja vlogo regulatorja v neki zaprtozančni konfiguraciji. Bolj pogosta je indirektna oblika, kjer ekspertni sistemi prevzamejo nadzorno nalogo nad sistemi vodenja na nižjih nivojih. Ta nadzorna naloga lahko pomeni spreminjanje parametrov podrejenim sistemom vodenja, vklapljanje in izklapljanje določenih podsistemov glede na izbrane kriterije, spremljanje obnašanja za zagotavljanje varnosti, odkrivanje napak in druge dejavnosti. Ekspertni sistem lahko človeškega operaterja pri funkciji vodenja popolnoma nadomešča, ali pa mu služi le kot svetovalno orodje, ki ga operater uporablja po želji.

Druga oblika uporabe ekspertnega sistema pri vodenju sistemov pa je njegova uporaba v postopku načrtovanja sistema vodenja. Postopki načrtovanja vodenja sicer temeljijo na številnih, teoretično dobro podprtih numeričnih postopkih, vendar razvoj sistema vodenja dinamičnih procesov ni algoritmiziran postopek in je močno odvisen od znanja pa tudi kreativnosti in intuicije načrtovalca. Ta lahko pri načrtovanju upošteva raznovrstne omejitve, izvedbene zahteve, zahteve operaterjev in si pri načrtovanju pomaga z raznovrstnimi oblikami informacije. V ekspertni sistem lahko shranimo veščine ali del veščin načrtovalcev sistemov vodenja in ga uporabimo kot pomoč (ne nadomestilo!) drugim načrtovalcem. S tem namenom je bilo razvitih kar nekaj ekspertnih sistemov za naloge kot so: načrtovanje določenih kompleksnejših vrst regulatorjev, uglaševanje raznih vrst regulatorjev itd.

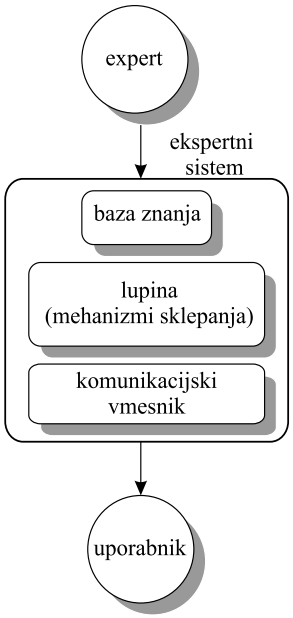
Večina simbolične umetne inteligence, ki se uporablja pri vodenju sistemov temelji na pravilih. Osnovna oblika pravila je

IF *<pogoj>* ***THEN*** *<dejanje>*

Pogoj lahko izrazimo z Boolovo, binarno logiko ali pa na primer z mehko logiko, ki smo jo že spoznali. Ekspertni sistem v grobem delimo na tri glavne module (Sl. 7.154): bazo znanja, mehanizem sklepanja in uporabniški vmesnik.

Glavni problem pri načrtovanju ekspertnih sistemov je načrtovanje baze znanja oziroma prenos znanja veščine operaterja na napravo ali, konkretno pri vodenju, na sistem vodenja (Bratko, Urbančič, 1996).

Možen pristop k prenosu znanja operaterja je z intervjuji operaterja, pri katerih pričakujemo, da bo operater opisal svojo veščino oziroma delovanje v različnih situacijah. Ta opis potem ustrezno formaliziramo, da postane primeren za vpis v avtomatski sistem vodenja. Problematično pri tem pristopu je, da so operaterjeve veščine v precejšnji meri podzavestne, in jih zato operater ni zmožen opisati popolnoma, ampak le približno. S takim načinom pridobivanja znanja za avtomatski sistem vodenja lahko dobimo le osnovne smernice, ki jih uporabimo pri načrtovanju baze znanja. Direktni operaterjevi opisi veščin torej ne zadostujejo pri načrtovanju sistema vodenja.



Sl. 7.154. Zgradba ekspertnega sistema

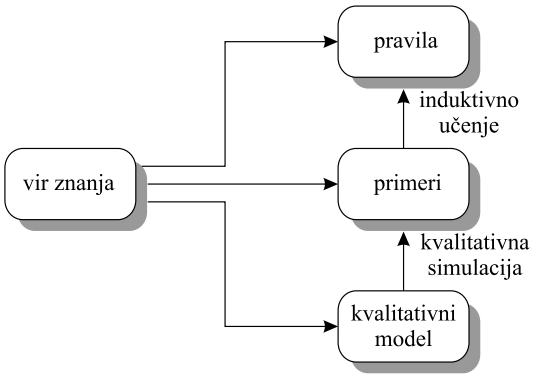
Alternativni pristop, ki izhaja iz pomanjkljivosti opisanega, temelji na zajemanju znanja iz operaterjevega prikaza veščin. Ta prikaz veščin je ponavadi dostopen v obliki zapisov dejanj zbranih v času delovanja procesa. Postopek izločanja pravil iz zapisov operaterjevih dejanj se imenuje strojno učenje (Bratko, Urbančič, 1996).

7.4.3.2 Strojno učenje

Pri zapisu zgodovine delovanja procesa imamo podatke ponavadi vzorčene in jih lahko uredimo v pare (stanje sistema, dejanje) razvrščene v časovnem zaporedju. Pri strojnem učenju vidimo zaporedje teh parov kot množico primerov in zanemarimo njihovo odvisnost od časa. To zanemaritev lahko upravičimo z dejstvom, da je regulacijska akcija odvisna le od stanja časovno nespremenljivega procesa. Po drugi strani pa so dejanja operaterja odvisna tudi od preteklih stanj procesa, in ne le od trenutnega. Dejstvo je, da operater sledi določene cilje v določenem časovnem intervalu in nato zamenja svoje cilje. Zaradi tega so dejanja odvisna od stanja procesa in od ciljev oziroma zahtev, čeprav te niso del stanja procesa, ampak so samo v operaterjevih možganih. Ne glede na opisano pa večina pristopov strojnega učenja uporablja le množice primerov kot smo jih opisali na začetku.

Veliko algoritmov strojnega učenja je pravzaprav iskanje funkcije, ki poveže stanje procesa z ustreznim dejanjem. Za iskanje te funkcije je na voljo katerikoli pristop za identifikacijo (tudi tisti, ki smo jih že omenili pri umetnih nevronskih mrežah). Pri ekspertnih sistemih so metode strojnega učenja omejene na simbolični zapis in zato se

metode učenja-identifikacije, ki smo jih opisali pri umetnih nevronskih mrežah ne uporabljajo. Obstaja dober razlog za tako omejitev. Pri ekspertnih sistemih želimo ohraniti razumevanje operaterjeve veščine, kar je lahko pomembno dejstvo pri načrtovanju inteligentnega sistema vodenja. Najbolj znane metode strojnega učenja za ekspertne sisteme so:



Sl. 7.155. Ilustracija prenosa znanja

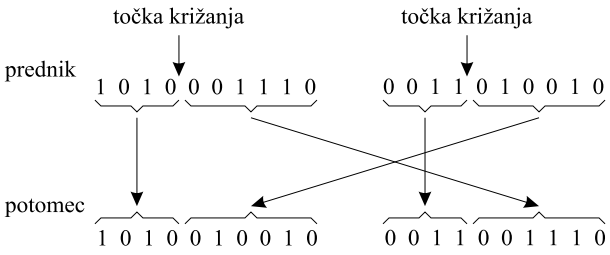
- učenje zvez iz primerov;
- učenje odločitvenih dreves;
- induktivno logično programiranje (angl. Inductive Logic Programming).

Sl. 7.155 ilustrira prenos znanja iz njegovega vira.

Pristop k učenju (iskanju parametrov in strukture), ki ni omejen le na sisteme s simboličnim zapisom, pa so genetski algoritmi.

7.4.3.3 Genetski algoritmi

Genetski algoritmi izhajajo, kot pove že ime, iz genetike, ki se je razvila pri proučevanju živih bitij. Lastnosti bitij so zapisane v genih, ki so nosilci informacij. Če želimo spremeniti lastnost organizma, na osnovi določenih pravil spremenimo gene. Na teh spoznanjih temelji teorija genetskih algoritmov, ki je pravzaprav prevedba opisanega dogajanja v računalniško okolje. Na ta način dobljeni algoritmi so primerni za reševanje najrazličnejših optimizacijskih problemov. V računalniku so geni predstavljeni s številkami, katerih določene kombinacije pomenijo vrednosti parametrov optimiranega osebka. Ponavadi se uporablja dvojiško kodiranje parametrov, kot je to prikazano na Sl. 7.156. Genetski algoritem nima o kodiranju parametrov nobene informacije (npr. koliko parametrov optimiramo in koliko bitov ima vsak). Med svojim delovanjem obravnava množico takšnih zaporedij, od katerih vsako predstavlja eno rešitev. Osebek je v primeru problematike s področja vodenja npr. regulator, za katerega iščemo najboljše nastavitve parametrov.



Sl. 7.156. Primer križanja vrednosti gena z dvojiško zapisano informacijo

Postopek optimizacije je sestavljen iz več korakov. Najprej določimo prostor preiskovanja, nato predpišemo funkcijo uspešnosti, ki jo bo uporabljal genetski algoritem in izberemo začetne vrednosti parametrov ter nazadnje poženemo optimizacijo. Genetski algoritem je sestavljen iz postopkov ovrednotenja, razmnoževanja, križanja in mutiranja ter se izvaja, dokler ni dosežen ustavitveni pogoj, ki je lahko določen na več načinov. V večini primerov se izvajanje zanke konča po vnaprej določenem številu korakov ali ko se rezultat, glede na funkcijo uspešnosti, ne izboljšuje več. Ker je genetski algoritem delno naključne narave, je koristno optimizacijo pognati večkrat. Po vsakem teku namreč dobimo nekoliko drugačno rešitev, na koncu pa se odločimo za tisto, ki nam najbolj ustreza.

Če opazujemo živa bitja, ki so se razvila z evolucijo, vidimo, da je postopek zelo učinkovit, vendar se pri prenosu na računalnik pojavijo določene težave. Predvsem so operacije, ki se dogajajo z geni v naravi, precej bolj zapletene, kot te v računalniških algoritmih, poleg tega pa tudi niso še povsem raziskane. Drugi problem je časovna zahtevnost. Genetski algoritmi obdelujejo množico rešitev, pri tem pa morajo preveriti uspešnost vsake izmed njih. Ker današnji računalniki še vedno delujejo po von Neumanovi zasnovi, ovrednotijo rešitve eno za drugo, kar pri velikem številu elementov v enem koraku traja precej časa. Po drugi strani se v naravi določena vrsta lahko razvija milijone let, mi pa običajno želimo rešitev takoj oziroma najkasneje v nekaj urah. Zavedati se moramo, da genetski algoritmi niso univerzalna optimizacijska metoda, uspešno pa zapolnjujejo nekatere vrzeli, ki so bile z doslej znanimi metodami zelo težko rešljive.

Če podrobneje pogledamo opisane metode in pristope, je hitro jasno, da uporaba ene ali več metod umetne inteligence, še ne zagotavlja inteligentnosti sistema vodenja. S primerno izbiro metode in načrtovanjem pa imamo s temi pristopi na voljo več orodij, s katerimi lahko načrtujemo sisteme vodenja, ki imajo višjo stopnjo avtonomnosti.

Ko damo metode umetne inteligence pod drobnogled (na primer umetne nevronske mreže ali mehko vodenje) lahko opazimo, da imamo opravka z nelinearnimi sistemi, s primerno prijaznimi vmesniki za načrtovalce in izvajalce. Če gledamo na ta način, je

tudi učenje (kot element inteligentnosti) le oblika optimizacije sistema.

Zelo pomembno je razumeti, da inteligenca oziroma avtonomnost sistema vodenja ni lastnost, ki se jo pridobi z uporabo določene metode, temveč lastnost, ki jo razumni načrtovalec načrtno vgradi v sistem vodenja.

7.5 Metode in postopki za razvoj programske opreme

V računalniško podprtih sistemih vodenja je seveda velika večina funkcij izvedena s pomočjo programske opreme. Zato so vse bistvene lastnosti sistema za vodenje v veliki meri odvisne od kakovosti programske opreme. S tem, kako učinkovito in kakovostno razvijati programsko opremo se ukvarja programsko inženirstvo, ki je preraslo v samostojno stroko znotraj računalništva.

Pričujoče podpoglavje je namenjeno programskemu inženirstvu in pa domenskemu inženirstvu, ki predstavlja eno od sodobnih vej programskega inženirstva, temelječo na ponovni uporabljivosti že razvitih modulov oziroma komponent.

Poudarek podpoglavja je na razlagi principov, na katerih temelji kakovosten razvoj programske opreme, na prikazu sodobnih pristopov k realizaciji začetnih faz v življenjskem ciklu programske opreme ter opredelitvi principa domenskega inženirstva, s težiščem na razgraditvi domene zveznih in šaržnih procesov.

7.5.1 Metode programskega inženirstva

Z metodami programskega inženirstva rešujemo probleme v življenjskem ciklu programske opreme. Ker kakovost programske opreme zavisi predvsem od kakovosti aktivnosti v zgodnjih fazah tega cikla, obstaja največ metod prav za analizo, specificiranje, načrtovanje in preizkušanje, manj pa za integracijo in vzdrževanje. Vse metode, ne glede na to ali so procesno ali objektno usmerjene, upoštevajo načela programskega inženirstva, ki so bistvena za doseganje kakovostnega izdelka, to je programske opreme.

7.5.1.1 Programsko inženirstvo

Programsko inženirstvo je v primerjavi z drugimi strokami, kot so npr. gradbeništvo, strojništvo, elektrotehnika, relativno mlado. Prvič se je izraz pojavil koncem šestdesetih let, ko so na konferenci o pogramski opremi razpravljali o prijemih za odpravo posledice zastoja razvoja na tem področju. Definicija, ki so jo takrat postavili, še danes zadovoljivo opisuje programsko inženirstvo kot disciplino (Presman, 1987). Definicija pravi, da je ***programsko inženirstvo uvedba in uporaba jasnih inženirskih načel za graditev ekonomične programske opreme, ki je zanesljiva in učinkovito deluje v realnem svetu***.

Osnovni predmet, s katerim se ukvarja programsko inženirstvo, je programska oprema.

Programska oprema je glede na proces nastajanja, namen uporabe in zahtevane karakteristike zelo raznovrstna, (npr. programska oprema operacijskih sistemov, programska oprema za poslovno-informacijske sisteme, programska oprema za sisteme realnega časa in vgrajene sisteme, programska oprema za večpredstavnost, itd.). Področje vodenja procesov je, tudi kar se tiče programske opreme, zelo zahtevno, saj je za izvedbo algoritmov vodenja v najzahtevnejših primerih potrebno izdelati programsko opremo, ki zadovoljuje zahtevam iz celotnega spektra. Če je bila nekoč najtežja zahteva, da morajo industrijske procese voditi sistemi, ki so izvedeni s programsko opremo realnega časa, pa se danes vse bolj kaže zahteva tudi po programski opremi, ki je sposobna vzdrževati podatke v procesnih podatkovnih bazah in komunicirati z uporabniki preko alfanumeričnih izpisov, statičnih in dinamičnih slik ter zvoka.

Programska oprema je torej zanesljiva, učinkovita in ekonomična, če je izdelana v skladu z načeli programskega inženirstva. Med najvažnejša načela spadata načeli abstrakcije in enkapsulacije.

Abstrakcija je obravnavanje samo tistih informacij, ki so bistvene za določeni nivo dekompozicije ter odložitev podrobnih odločitev v čim bolj pozne faze razčlenjevanja. Abstrakcijo bi lahko definirali kot osnovno orodje za obvladovanje kompleksnosti. V programskem inženirstvu uporabljamo različne vrste abstrakcij. Od bolj enostavnih abstrakcij omenimo podatkovne strukture in funkcionalno dekompozicijo, od kompleksnejših pa objektno in procesno abstrakcijo.

Enkapsulacija pomeni pravilen način združevanja informacije v ustreznih entitetah. Dosežemo jo s pomočjo principov modularnosti, lokalizacije in skrivanja informacij. *Modularnost* je smiselno razčlenjevanje zahtev, načrtovanja in implementacije v elemente ustrezne velikosti in kompleksnosti, ob upoštevanju njihove čim večje notranje kohezije in čim nižje medsebojne sklopljenosti. *Lokalizacija* je način grupiranja elementov v module, predvsem ob upoštevanju sklopljenosti in povezanosti, ki bosta opisani v nadaljevanju. *Skrivanje informacij* je v tesni povezavi z abstrakcijo. Cilj skrivanja informacij je onemogočiti dostop do informacije, ki ni nujna na določenem nivoju abstrakcije.

Z upoštevanjem opisanih načel v proces razvoja programske opreme vpeljemo red in logiko ali z drugimi besedami, strukturo. Strukturiranje je lahko funkcionalno, objektno, procesno, podatkovno ali glede na relevantne entitete iz problemske domene. Paradigme strukturiranja, stili in vzorci, ki sestavljajo model načrtovanja programske opreme, predstavljajo *arhitekturo* programske opreme.

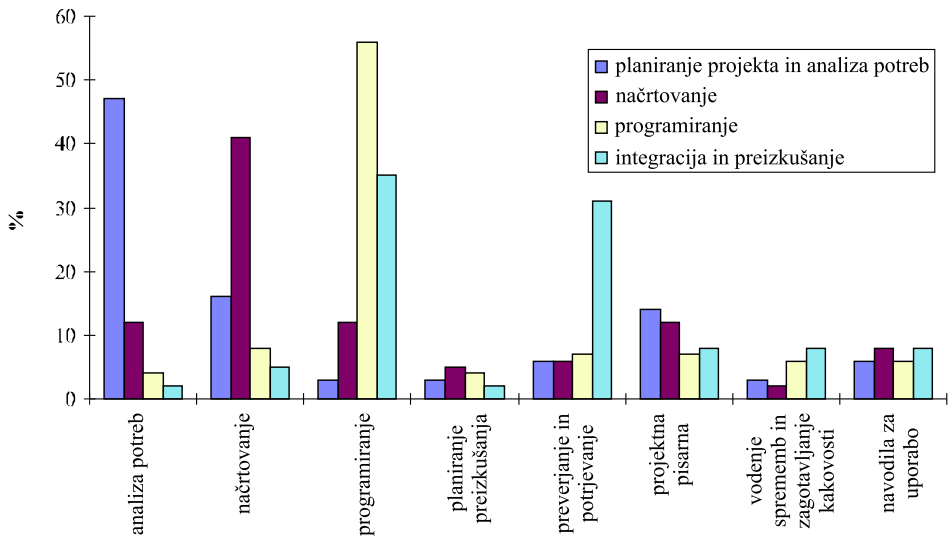
7.5.1.2 Kaj zajema programsko inženirstvo

Programsko inženirstvo v splošnem zajema obravnavanje vseh aktivnosti, ki naj bi se izvajale v procesih razvoja in uporabe programske opreme in vseh proizvodov, ki nastajajo v teh procesih. Najpomembnejši aspekt obravnave je vsekakor življenjski cikel, v katerem so poleg faz in aktivnosti obravnavani tudi metrika in kakovost programske opreme ter upoštevanje subjektivnih človeških faktorjev. Na tem mestu

bomo na kratko predstavili le značilnosti življenjskega cikla in ozadje ter merila za presojo zrelosti procesa izgradnje programske opreme.

Življenjski cikel programske opreme

Življenski cikel programske opreme se v principu ne razlikuje od splošnega življenjskega cikla, ki smo ga predstavili v drugem poglavju in je seveda prav tako sestavljen iz faz in aktivnosti. Porazdelitev posameznih akivnosti znotraj faz temelji na atributih projekta, kakovosti komunikacije med naročnikom in izvajalcem, zahtevnosti problema in zahtevnosti rešitve. Konkretna porazdelitev je različna za vsak projekt in je odvisna od kvalitativnih ocen vseh štirih skupin atributov.



Sl. 7.157. Prepletenost faz in aktivnosti v življenjskem ciklu programske opreme

Na osnovi teh ocen je bil postavljen COCOMO model (Boehm, 1981), s pomočjo katerega se da vnaprej planirati potek projektov programske opreme tako s časovnega kot stroškovnega vidika. V model je lahko zajeta tudi faza vzdrževanja oziroma izkoriščanja programske opreme, za katero velja, da je toliko manj potratna z viri, kolikor bolj kakovostno so bile izvedene zgodne razvojne faze. Na Sl. 7.157 je prikazana tipična porazdelitev aktivnosti znotraj faz z vidika obremenitve projektne skupine, iz katere je razvidno, katere aktivnosti se morajo sočasno izvajati in s kakšno intenzivnostjo. Aktivnosti, ki se morajo izvajati skozi ves projekt razvoja in izvedbe programske opreme so: analiza potreb, načrtovanje, programiranje, planiranje preizkušanja, preverjanje in potrjevanje rešitev, vodenje projektne pisarne, vodenje sprememb in zagotavljanje kakovosti ter izdelava navodil za uporabo.

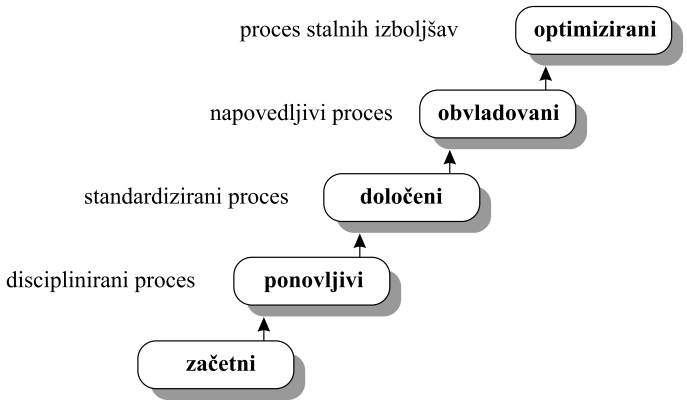
Glede na sosledje posameznih faz in na vrsto aktivnosti, ki se v teh fazah izvajajo, tudi

pri programskem inženirstvu ločimo podobne modele življenjskega cikla, kot pri sistemskem inženirstvu.

Kakovost procesa razvoja in izvedbe programske opreme

Ne glede na to, kateri pristop izberemo pri projektih programske opreme, je bistveni pogoj za uspeh dovolj visoka kakovost procesa nastajanja programskega izdelka, oziroma širše računalniškega sistema, in kakovost samega izdelka. Poudarek na kakovostnem procesu je v zadnjem času vse večji, kajti nesporno je, da brez kakovostnega procesa ni kakovostnega izdelka. Proces razvoja in izvedbe programske opreme praviloma vključuje izvajalsko in naročniško organizacijo. Zavedanje o pomenu kakovosti in poznavanje informacijske tehnologije je lahko v obeh organizacijah zelo različno. Bistveno za uspeh je torej, da zna vsaka organizacija sama za sebe določiti, kje se nahaja glede na zrelost takoimenovanega procesa razvoja in izvedbe sistemov programske opreme. Pri tem si je moč pomagati z rezultati številnih raziskav v svetu. Najbolj poznan, in tudi priznan, je model zrelosti programske opreme (Paulk in sod., 1993).

Zrelostni model je zasnovan na konceptu stalnih izboljšav. Celoten proces razvoja in izvedbe sistemov programske opreme je opredeljen po kakovosti na pet nivojev. Za vsak nivo so podani cilji, ki ob izpolnitvi stabilizirajo pomembne komponente razvoja. Ko je razvojni proces na določenem nivoju stabiliziran, se lahko organizacija premakne proti naslednjemu nivoju, pri čemer doseženi cilji predstavljajo pogoje za prehod na višji nivo. Ker model ni pomemben le za oceno ali samooceno trenutnega stanja ponudnika ali uporabnika programske opreme, ampak je tudi izrednega pomena pri cenovnem usklajevanju stroškov projektov, si ga na kratko poglejmo (Sl. 7.158).



Sl. 7.158. Model zrelosti procesa izgradnje programske opreme

Model določa pet stopenj zrelosti in sicer: začetno, ponovljivo, določeno, obvladovano in optimizirano. Začetna stopnja opredeljuje proces, ki je naključen, in kjer njegova kakovost zavisi predvsem od znanja, izkušenj in pridnosti posameznikov.

Ponovljiva stopnja je dosežena, ko se v projekte programske opreme vključi upravljanje, s katerim se doseže izpolnjevanje cenovnih in časovnih rokov ter funkcionalnosti izdelkov. Na ta način se lahko znanje, doseženo na prejšnjih projektih, koristno uporabi na sledečih.

Ko se v ponovljivi proces vključi še izdelava dokumentacije, ki je v skladu s standardi, prav tako kot sam proces razvoja in izvedbe, postane proces določen. Dokumentacija je lahko izdelana po internih ali zunanjih standardih. Standardiziranost procesa zahteva določitev postopkov ocenjevanja procesa in izdelkov, izobraževanje in koordinacijo med izvajalci in naročniki.

Da se ugotovi, koliko standardiziranost prispeva k dvigu kakovosti, je potrebno preiti iz ocen na meritve karakteristik procesa in izdelkov. Ko so postopki meritev in uporabe rezultatov določeni in uporabljani, je dosežena obvladljiva stopnja.

Zadnja in najvišja stopnja je optimizirana stopnja. V tej stopnji se izvaja neprestano izboljševanje že doseženega, kar pomeni, da je potrebno obvladovati spremembe, tako v procesu, izdelkih in tehnologijah, ki oboje omogočajo.

Zrelostni model izhaja iz izhodišč, na katerih temeljijo tudi standardi za obvladovanje kakovosti procesov ISO 9000. Številni avtorji dokazujejo, da med njima ni bistvenih razlik, in da se zahteve enega lahko prenesejo v zahteve drugega. Izdelane so analize podrobnih preslikav zahtev za posamezne nivoje zrelosti v točke standarda ISO 9001 (Paulk, 1995). Ker pa se programska oprema le loči od sistemov in dejavnosti, za katere je namenjen ISO 9001, je bilo izdano posebno vodilo z oznako ISO 9000-3 (ISO, 1991). V njem so opisani načini in zahteve za uvajanje standardov ISO v proces razvoja, izdelave in vzdrževanja programske opreme. Standard ne predpisuje nobene metode ali metodologije programskega inženirstva, zahteva le, da je proces v izvajalskih organizacijah višji od ponovljive zrelostne stopje. Model zrelosti in ISO 9000-3 tako vsak na svoj način določata pogoje in načine procesa razvoja, izvedbe in vdrževanja programske opreme, pri čemer je prvi preglednejši za izvajalne organizacije, drugi pa za uporabnike.

Kakovost programske opreme kot izdelka

Glavni cilj programskega inženirstva je izdelava kakovostne programske opreme. To kakovost merimo s pomočjo naslednjih atributov:

- pravilnost, to je skladnost programske opreme s potrebami (statična lastnost);
- zanesljivost, to je zadovoljivo delovanje programske opreme skozi čas (dinamična lastnost);
- varnost: programska oprema je varna, če med delovanjem nikoli ne povzroči situacije, ki bi lahko ogrozila ljudi ali naprave;
- robustnost, to je sposobnost programske opreme, da pravilno funkcionira tudi ob nenormalnih razmerah;

- možnost oz. enostavnost preizkušanja pravilnosti, zanesljivosti, varnosti in robustnosti programske opreme;
- razumljivost, to je stopnja, do katere je možno razumeti programsko opremo oz. njeno dokumentacijo;
- pnovna uporabljivost, to je možnost uporabe iste programske opreme pri različnih aplikacijah;
- možnost dokazovanja pravilnosti programske opreme;
- možnost kontroliranega vzdrževanja oz. spreminjanja zaradi razširjanja ali popravljanja napak;
- prenosljivost v drugo strojno ali programsko okolje;
- učinkovitost, oz. učinkovita izraba računalniških virov (pomnilnik, čas).

7.5.1.3 Zgodnje razvojne faze programske opreme

Zgodnje razvojne faze življenjskega cikla programske opreme obsegajo:

- analizo potreb sistemov ali procesov, ki bodo avtomatizirani ali informatizirani s programsko opremo;
- specificiranje zahtev programske opreme in
- načrtovanje programske opreme.

Analiza potreb sistemov ali procesov, ali kratko sistemska analiza, je prva in najpomembnejša faza življenjskega cikla, kot aktivnost pa se izvaja tudi v ostalih fazah. Glavni namen sistemske analize je obravnavanje naročnikovih potreb in zahtev, njihovo pravilno razumevanje in dokumentiranje. V sistemski analizi izredno pomembno vlogo igra komunikacija med naročniki in izvajalci. Informacije, ki jih v analizi dobi izvajalec od naročnika, so praviloma nepopolne, nejasne in celo nasprotujoče. Osnovni cilj analize je, da se iz neurejenih informacij izdela usklajen model sistema, bodisi kot dokument ali prototip. To je možno le s številnimi ponavljanji, v katerih naročnik in izvajalec preoblikujeta in usklajujeta tak model oziroma prototip. Pred leti opravljene raziskave so pokazale, da se je 56% vseh napak pojavilo v fazi analize, pri čemer je bilo v to fazo vloženo skoraj polovico celotnega dela. Največji stroški odpravljanja napak so zabeleženi prav za odpravljanje napak, storjenih v fazi sistemske analize. Ocenjuje se, da se je stanje na tem področju v zadnjem času precej izboljšalo, predvsem zaradi vse večje uporabe konceptov in metod sistemskega ter programskega inženirstva. Ne glede na to pa še vedno velja, da je napake, ki so storjene v začetnih razvojnih fazah, kasneje najtežje odkriti, in da je cena njihovega odpravljanja izredno visoka.

Proces nadaljnje obdelave izsledkov sistemske analize imenujemo specificiranje zahtev. Specificiramo strukturo računalniškega sistema, v katerega bo vgrajena programska oprema, sestavne dele, njihova statična in dinamična razmerja ter funkcije. Tudi pri specificiranju igra pomembno vlogo komunikacija, tokrat med med sistemskim analitikom in načrtovalcem. Meja med specificiranjem in začetnim načrtovanjem je

navadno neizrazita in zavisi tako od zahtevnosti problema in rešitve kot tudi od zrelosti procesa izgradnje programske opreme.

Načrtovanje programske opreme je proces skozi katerega se zahteve, ki so bile v fazi sistemske analize vsklajene s potrebami naročnika, preslikajo v sprva grobo, potem pa, skozi proces postopnega razčlenjevanja, vedno bolj podrobno predstavitev programske opreme. Sama preslikava predstavlja kreativni proces, pri katerem so ključnega pomena znanje in izkušnje. Problem je potrebno razčleniti zaradi znanega pravila, da je vsota kompleksnosti *n* podproblemov manjša od kompleksnosti vsote istih *n* podproblemov.

Za uspešno načrtovanje kompleksne programske opreme za sisteme vodenja procesov moramo znati obvladovati tudi najzahtevnejše probleme načrtovanja, ki so predvsem *medsebojno izključevanje* pri dostopu do skupnih virov ter medprocesna *komunikacija* in *sinhronizacija.* Te probleme rešujemo s pomočjo raznih mehanizmov, ki so del procesne ali objektne abstrakcije, če pa niso, jih je potrebno realizirati s pomočjo obstoječih nižjenivojskih abstrakcij (konstruktov). Od mehanizmov za medprocesno komunikacijo in sinhronizacijo lahko omenimo *semaforje* in *monitorje* pri sistemih s skupnim pomnilnikom ter razne mehanizme, ki temeljijo na *pošiljanju sporočil* pri distribuiranih sistemih.

V zgodnjih razvojnih fazah življenjskega cikla programske opreme se za dokumentiranje spoznanj, potreb in zahtev največ poslužujemo modelov, ki so vsebovani v metodah programskega inženirstva. Te metode so v skladu z veljavnimi standardi in priporočili (npr. IEEE Software Engineering Standards), hkrati pa tudi podprte z računalniškimi orodji (orodja programskega in sistemskega inženirstva - CASE).

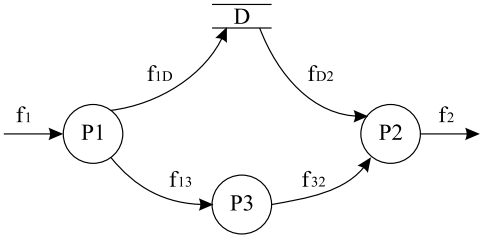
Metode programskega inženirstva za zgodnje razvojne faze

V kratkem pregledu prikazujemo samo neprototipne metode in predvsem tiste, ki so primerne za uporabo v zgodnjih razvojnih fazah sistemov procesnega vodenja (to so metode za sisteme realnega časa), kajti velika razlika med poslovno-informacijskimi in sistemi realnega časa se odraža tudi v postopkih analize in načrtovanja teh sistemov. Procesi in funkcije, ki jih avtomatiziramo v poslovno-informacijskih sistemih, so mnogo jasnejše in enostavnejše, bolj pa so zahtevne strukture podatkov, s katerimi te funkcije operirajo. Zato se pri razvoju informacijskih sistemov uveljavljajo podatkovno usmerjeni pristopi, kot npr. informacijsko inženirstvo, ki se je uveljavilo v začetku 80-tih let in temelji na stabilnih podatkovnih modelih. Pri snovanju in načrtovanju sistemov realnega časa pa je ravno obratno. Podatkovne strukture so običajno preprostejše, zahtevni in časovno kritični pa so procesi in funkije. Pri razvoju sistemov realnega časa se zato uporabljajo procesno usmerjeni pristopi (metode in metodologije), v zadnjem času pa vedno bolj objektno usmerjeni pristopi. Vsak od teh ima določene prednosti in tudi pomankljivosti. Tabela 7.11 prikazuje primerjavo med skupinami omenjenih metod.

Tabela 7.11. Primerjava različnih metod za zgodnje razvojne faze sistemov procesnega vodenja

Metode	Prednosti	Slabosti
PROCESNO USMERJENE NA OSNOVI PODATKOVNIH TOKOV– podatkovno vzbujanje (De Marco, Hatley-Pirbhai, ...)	• zelo razumljive • enostavna notacija v modelih	• vedno manj uporabljane • neponovljive • vodijo k veliko kode
PROCESNO USMERJENE NA OSNOVI PODATKOVNIH TOKOV - dogodkovno vzbujanje (Ward-Mellor, Yourdon's Modern Structured Analysis, ...)	• ponovljive • razumljive • podprte z orodji CASE	• zelo občutljive na spremembe • zahtevna notacija • zahtevna preslikava v izvedbo
OBJEKTNO USMERJENE (Shlaer-Mellor, Coad-Yourdon, OMT, ...)	• ponovljive • razumljiva notacija • neposredna preslikava v izvedbo • malo občutljive na spremembe • podprte z orodji CASE	• zahtevne za razumevanje

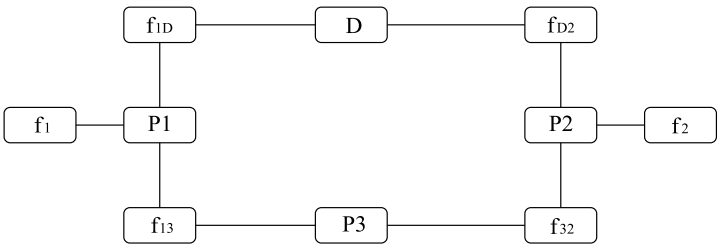
Bistvena razlika med procesno in objektno usmerjenimi metodami je v načinu dekompozicije in v osnovnih entitetah modelov. Oba načina dekompozicije sta na kratko opisana v sledečih podpoglavjih, razliko med entitetami modelov pa na kratko prikazuje preprost in poenostavljen zgled.



Sl. 7.159. Model procesne dekompozicije

Na Sl. 7.159 je z diagramom tokov podatkov, ki je opisan v nadaljevanju, prikazan procesni model za nek nivo dekompozicije, kjer vidimo tri procese (P1 do P3) in podatkovno shrambo D, povezane med seboj s tokovi podatkov (f). Lastnosti procesov so nadalje definirane s pisnimi specifikacijami, lastnosti podatkovnih tokov in shrambe pa s strukturo (kar pa na sliki ni prikazano).

Preslikavo v statični informacijski objektni model prikazuje Sl. 7.160.



Sl. 7.160.Model objektne dekompozicije

Na tej sliki nastopajo vse entitete s prejšnjega modela (procesi, podatkovna shramba, podatkovni tokovi), le da so sedaj predstavljene z objekti. Vsak objekt je nadalje opisan s svojimi atributi in obnašanjem, kar pa je skrito v njihovi notranjosti. Navzven so vidne le povezave med objekti, ki predstavljajo relacije med njimi. (Še enkrat poudarjamo, da je zgled močno poenostavljen in služi le za ilustracijo razlike, nikakor pa ne kot postopek preslikave med dvema konceptualno različnima modeloma.)

Sistemska analiza se od načrtovanja programske opreme precej razlikuje glede na vsebino, zahtevano znanje in rezultate. Sistemska analiza zahteva poznavanje tehnologije sistemov in metodološka znanja, medtem ko načrtovanje zahteva znanja računalništva. V procesu analize in načrtovanja sistemov sodelujejo strokovnjaki, ki posedujejo vsa ta znanja: uporabniki z različnih tehnično-tehnoloških področij, sistemski analitiki in sistemski načrtovalci. Stično točko v medsebojni komunikaciji predstavljajo metode in metodologije, ki vsebujejo različne modele. Ti morajo biti poznani in razumljivi vsem prej naštetim strokovnjakom. Modeli, ki so uporabni za analizo in specificiranje procesnih sistemov morajo:

- minizirati čas in napor pri razvoju sistemov;
- na jasen in skladen način predstaviti različne sisteme;
- biti primerni za predstavitev različno obsežnih sistemov,
- omogočati predstavitev podatkov, procesov in krmiljenja;
- omogočati natančno predstavitev časovnih dogajanj v sistemih.

Za razliko od matematičnih modelov, ki smo jih obravnavali v začetku tega poglavja so ti modeli lahko grafični, tabelarični ali besedilni. Predstavitveno močnejši so grafični. Mednje uvrščamo različne vrste diagramov, od katerih ima vsak svoj nabor simbolov in pravil. Besedilni modeli so po posebnih pravilih urejena besedila in so lahko samostojna ali pa kot dodatek grafičnim.

Procesno usmerjene metode

Pri procesno usmerjenih metodah gledamo na sistem kot na množico med seboj povezanih procesov. Vsak proces izvršuje določeno število funkcij, ki so lahko zelo kompleksne ali pa tudi zelo enostavne. Zelo kompleksne procese je potrebno dekomponirati na enostavnejše. Glede na način dekompozicije ločimo funkcionalno in

dekompozicijo glede na dogodke. Slednja je sodobnejša in primernejša in kot taka tudi uporabna pri razvoju sistemov programske opreme za vodenje procesov.

V zgodnjih razvojnih fazah uporabljamo vrsto modelov, med katerimi so najpogostejši diagrami tokov podatkov, diagrami prehajanja stanj in diagrami razmerij med entitetami. Izgled in uporaba le teh sta prikazana v nadaljevanju s pomočjo praktičnega zgleda, zato na tem mestu podajamo le njihove osnovne lastnosti.

Diagram tokov podatkov je grafični model, ki prikazuje tokove podatkov med procesi, torej procesno dimenzijo sistema. Osnovni elementi diagrama so: podatkovni (zvezni in diskretni) in kontrolni tokovi (ti so lahko dogodkovni tokovi ali promptni tokovi), podatkovni in kontrolni procesi, podatkovne in kontrolne shrambe. Procesi v kontekstu diagramov tokov podatkov pomenijo element diagrama, izražen z grafičnim simbolom, ki preoblikuje podatke vhodnih tokov v podatke izhodnih tokov.

Za prikaz časovnega obnašanja sistema, to je časovne sistemske dimenzije, se najpogosteje uporablja *diagram prehajanja stanj*, ki je tudi grafični model. Diagram prikazuje stanja sistema, pogoje za prehod stanj in akcije, ki se izvedejo ob prehodih. To so tudi osnovni elementi diagrama.

Grafični model, ki se uporablja za prikaz podatkovne sistemske dimenzije, je *diagram razmerij med entitetami*. Diagram prikazuje shranjene podatke o sistemu in njihova medsebojna razmerja. Osnovni elementi diagrama so: objektni tip, pridruženi objektni tip in ramerje.

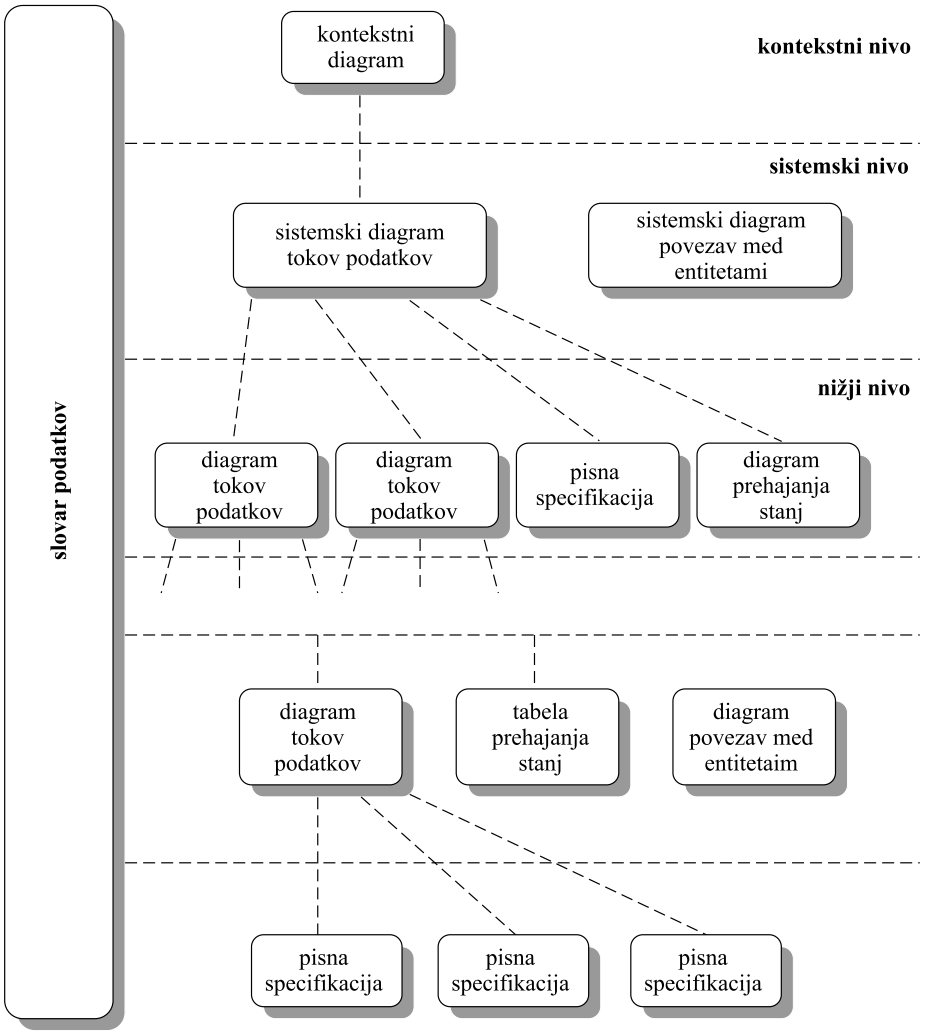
Tabela prehajanja stanj je tabelarični model, ki prikazuje stanja sistema v odvisnosti od pogojev za prehod stanj. Za razliko od diagrama prehajanja stanj, so v tabeli specificirana vsa stanja in vsi pogoji za prehod. Zaradi tega je manj pregledna kot diagram, vendar bolj natančna.

Pisne specifikacije so besedilni modeli, ki so uporabljeni za specificiranje osnovnih procesov. Specifikacije so lahko pisane kot: običajno besedilo, strukturiran naravni jezik, psevdokoda, zapis v obliki predpogojev/popogojev, odločitvena drevesa in grafi.

Slovar podatkov je besedilni model, ki predstavlja urejeno zbirko vseh podatkov, ki nastopajo v vseh ostalih modelih. Podatki so v slovarju podrobno opisani in po potrebi še nadalje razgrajeni.

Celoten model sistemske analize mora sestavljati množica med seboj vsklajenih modelov. V celotno strukturo so delni modeli vgrajeni po nivojih. Splošen primer nivojske strukture je prikazan na Sl. 7.161.

Najvišji nivo abstrakcije predstavlja kontekstni nivo, kjer se nahaja vsaj kontekstni diagram, lahko pa tudi še kašnen drug besedilni model. Kontekstni diagram je diagram tokov podatkov, ki vsebuje le en proces. Namen diagrama je prikazati, kaj je vsebina obravnavanega sistema, kaj je okolica sistema in prek, katerih podatkov sistem komunicira z okolico.



Sl. 7.161. Splošna nivojska struktura modelov

Naslednji, po hierarhiji nižji nivo, je sistemski nivo s sistemskim diagramom tokov podatkov. Na njem so prikazani glavni podprocesi obravnavanega sistema z grafičnimi simboli, tudi imenovanimi procesi.

Vsak tak proces, tako kontrolni kot podatkovni, se nadalje razgradi v diagrame, ali v primeru, da je preprost, v pisne specifikacije. Podatkovni procesi so dekomponirani z novimi diagrami tokov podatkov, kontrolni procesi pa z diagrami ali tabelami prehajanja stanj.

Povezave med posameznimi nivoji v hierarhični strukturi morajo biti skladne, kar z drugimi besedami pomeni, da morajo biti izstopajoči tokovi iz višjega nivoja po

strukturi enaki vstopajočim tokovom nižjega nivoja. Poleg mednivojske skladnosti je potrebna tudi skladnost med vsemi vrstami diagramov, to je med procesnimi (diagrami tokov podatkov), časovnimi (diagrami prehajanja stanj) in podatkovnimi (diagrami povezav med entitetami) diagrami ter skladnost med diagrami, procesnimi specifikacijami in slovarjem.

Objektno usmerjene metode

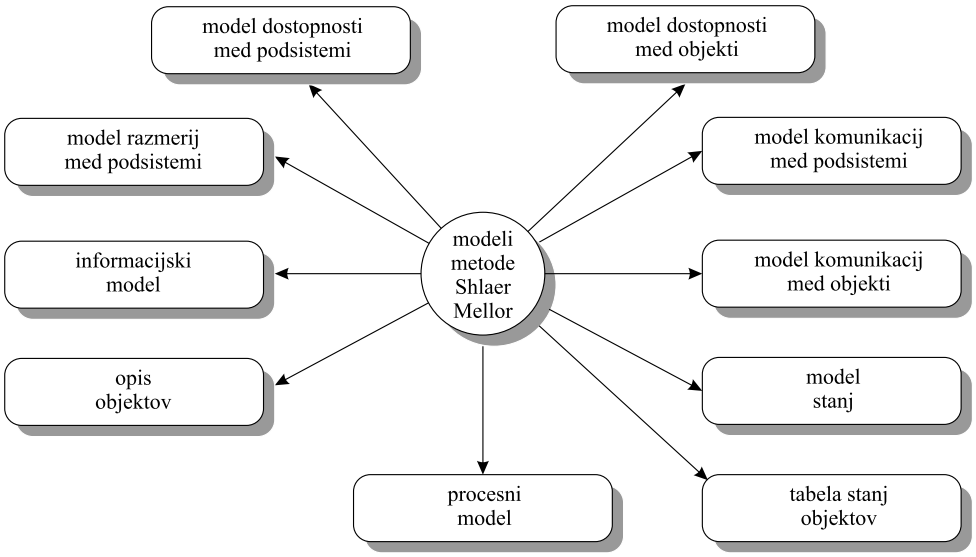
Objektno usmerjene metode sledijo konceptom objektno usmerjenega pristopa, ki ostajajo skozi vse razvojne faze enaki. Najpomebnejši so: identiteta, klasifikacija, dedovanje in polimorfizem. Večina od njih je implicitno nakazana v nadaljevanju, kjer so na kratko predstavljene splošne značilnosti pristopa.

Začetki objektno usmerjenega pristopa segajo v začetek sedemdesetih let, ko so razvili prve objektno usmerjene programske jezike. Najprej se je uveljavil v izvedbenih fazah, v osemdesetih letih pa se je razširil tudi v sistemsko načrtovanje in dalje v analizo. Objektno usmerjen pristop gleda na sistem kot na množico objektov, ki med seboj komunicirajo. Temelji na naslednjih osnovnih izhodiščih:

- vsaka stvar iz realnega sveta se da predstaviti z objektom. Vsak objekt ima svojo identiteto;
- vsak objekt je določen z množico lastnosti - atributov (internih spremenljivk objekta) in z množico obnašanj - metod (postopkov za osveževanje vrednosti ali dostopov do atributov);
- relacija (povezava) je abstrakcija množice relacij (odnosov), ki obstajajo med stvarmi v realnem svetu. Da lahko govorimo o relacijah v smislu objektno usmerjenega pristopa (relacije med objekti), je potrebno stvari iz realnega sveta najprej abstrahirati z objekti. Relacije nam tako povedo, v kakšnem odnosu je objekt z drugimi objekti in kakšno vlogo ima v celotnem sistemu;
- množica vrednosti atributov določa stanje objekta. Vrednosti atributov spreminjamo s pošiljanjem sporočil, ki aktivirajo ustrezne metode. Pošiljanje sporočil je edini legalni način za spreminjanje stanja določenega objekta;
- objekti, ki imajo podobno strukturo in obnašanje, se združujejo v razrede. Razred predstavlja osnovni vzorec obnašanja in strukture za množico podobnih objektov. Vsak objekt je primerek (pojavitev) nekega razreda;
- razred lahko definiramo kot specializacijo enega ali več razredov. Razred, ki je definiran kot specializacija nekega drugega razreda, imenujemo podrazred. Podrazred podeduje atribute in metode svojega nadrazreda.

Za objektno usmerjene metode veljajo v splošnem iste ugotovitve kot za procesno usmerjene. Bistvena razlika je v objektni abstrakciji in dekompoziciji. Ta se od procesne razlikuje v tem, da se na kontekstnem nivoju nahaja podatkovni pogled na sistem, ki je izveden z diagramom, na katerem so predstavljeni objekti obravnavanega sistema in relacije med njimi. Po izgledu in pravilih je objektno relacijski diagram precej podoben diagramu razmerij med entitetami. Vsak objekt se nato dekomponira glede na njegovo

obnašanje, ki je predstavljeno z diagramom prehajanja stanj ali tabelo prehajanja stanj. V vsakem stanju se izvajajo določene aktivnosti (v objektnem pristopu imenovane metode), ki preoblikujejo podatke in ustvarjajo pogoje za prehode stanj. Model aktivnosti je prikazan z diagramom, ki se navzven praktično v ničemer ne razlikuje od diagrama tokov podatkov. Ker objekti med seboj komunicirajo s sporočili, je pomemben model, ki prikazuje poti sporočil. Če sledimo sporočilom, ki dejansko spreminjajo stanja posameznih objektov, lahko sledimo procesom, ki jih analiziramo ali specificiramo. Objektno usmerjene metode poleg osnovnih modelov, ki so bili ravnokar omenjeni, uporabljajo še posebne, metodam lastne modele. Primer takega nabora modelov za objektno usmerjeno metodo Shlaer-Mellor (Shlaer in Mellor, 1985) je prikazan na Sl. 7.162.



Sl. 7.162. Modeli metode Shlaer-Mellor

∇

Primer 7.24: *Uporabe metode Ward-Mellor za sistemsko analizo realnega tehničnega procesa*

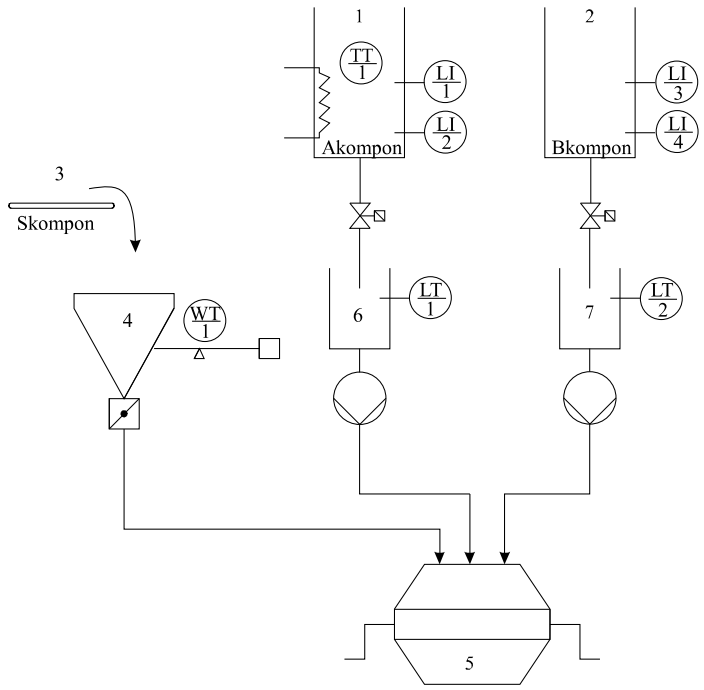
Opredelitev problema

Za primer smo vzeli poenostavljen primer realnega tehnološkega procesa mešanja komponent, ki ga v podobnih oblikah srečujemo v vrsti industrijskih podjetij. Gre za tipični sekvenčni proces, ki ga izvaja računalnik, nadzira pa operater. Tak primer omogoča prikaz uporabe ene izmed metod programskega inženirstva za sistemsko analizo skozi ves proces dekompozicije. Seveda se smiselnost uporabe takih metod veča z velikostjo obravnavanega problema. Izbrali smo metodo Ward-Mellor, ki je primerna za obravnavanje sistemov realnega časa. Za obravnavani primer smo na kratko opisali tehnološki postopek in njegovo vodenje ter na osnovi tega izdelali vse pomembne

modele in jih tudi opremili s komentarji.

Opis tehnološkega postopka

V procesu mešanja se v mešalcu mešajo ena sipka in dve tekoči komponenti. Poenostavljena tehnološka shema procesa je prikazana na Sl. 7.163.



Sl. 7.163. Poenostavljena tehnološka shema mešanja komponent

Tekoča komponenta A (Akompon) se nahaja v vmesnem rezervoarju 1, tekoča komponetna B (Bkompon) pa v vmesnem rezervoarju 2. Oba rezervoarja sta opremljena z indikatorji minimalnega dopustnega, delovnega in maksimalnega dopustnega nivoja. Rezervoar za Akompon je ogrevan z električnim grelcem in je opremljen tudi z merilnikom temperature.

S pomočjo dozirnega traku 3 se s sipko komponento (Skompon) polni tehtnica 4 na predhodno podano količino. Tehtnica se prazni v mešalec 5 s prostim padom. Z direktnim dotekanjem se polnita dve dozirni posodi 6 in 7, ki sta opremljeni z merilnikom nivoja. Iz dozirnih posod črpata črpalki komponenti v mešalec.

Po končanem mešanju se mešalec izprazni.

Opis načina vodenja

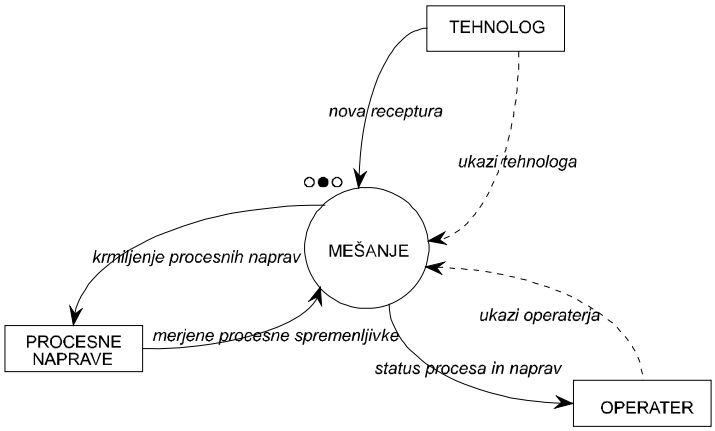
1. Tehnolog vnese recepturo, ki vsebuje podatke o količini Akompona in Bkompona, količini (teži) Skompona in času mešanja. Receptura se lahko spremeni kadarkoli, razen med doziranjem tekočih komponent v dozirni posodi in sipke komponente na tehtnico. Če se receptura spremeni med mešanjen, se mora mešanje izvesti v času iz stare recepure.
2. Če med doziranjem pade količina tekočih komponent v vmesnih rezervoarjih pod minimalno vrednost, se mora doziranje ustaviti, dokler se z ročnim polnenjem nivo v vmesnih rezervoarjih ne dvigne do vrednosti, ki omogoča zdozirati po recepturi zahtevano količino tekočih komponent v vmesne posode, to je do delovnega nivoja.
3. Akompon mora imeti stalno temperaturo, za kar skrbi električni grelec. Viskoznost Akompona se izračunava avtomatsko iz temperature in se posreduje operaterju.
4. Operater preko računalniškega sistema izvede zagon in ustavitev procesa ter začetek doziranja. Ostali postopki tečejo avtomatsko.
5. Operater preko računalniškega sistema sprejema vse statuse izvršnih členov in alarme.

Postopek reševanja

Preden pričnemo z zgledom analize je potrebno za pravilno razumevanje razložiti dvojen pomen besede *proces*. Prvič proces pomeni tehnološki proces, ki ga analiziramo. Drugič pa je proces element diagrama tokov podatkov, ki v splošnem pretvarja vhodne podatke v izhodne. V prvem primeru torej pomeni abstrakcijo zaporedja dogodkov iz realnega sveta, v drugem pa izražavo te abstrakcije s simbolom. Za popolno razumevanje zgleda bi morali pojasniti še vrsto pravil, vendar zaradi pomanjkanja prostora priporočamo bralcu, da si jih ogleda v (Shlaer in Mellor, 1986). V zgledu sprotno pojasnjujemo le tista pravila metode, ki so nujna za razumevanje.

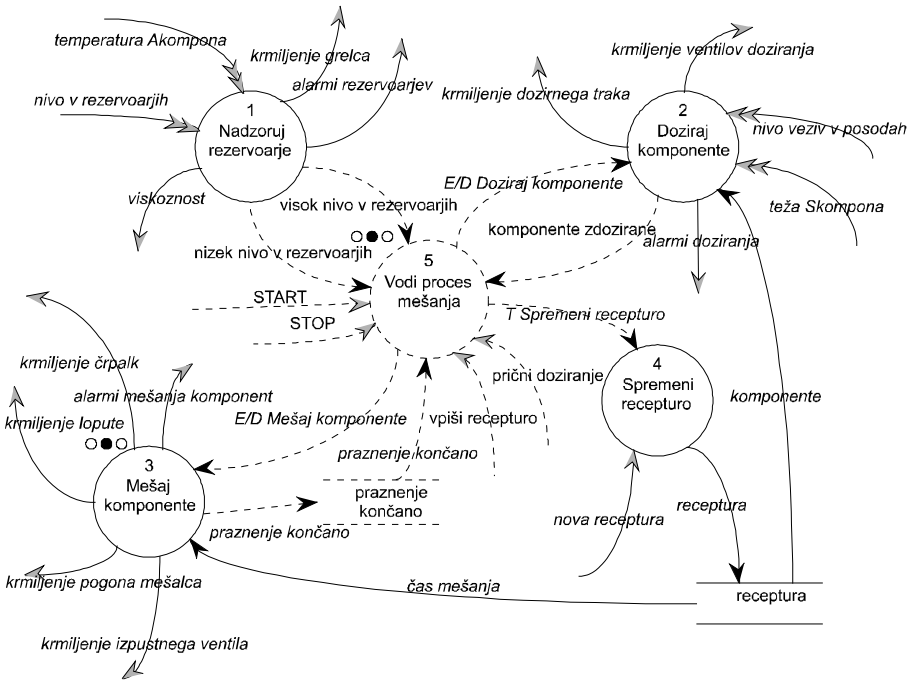
Algoritmi, ki naj jih izvajajo osnovni procesi in so opisani s pisnimi specifikacijami, niso opisani v podrobnosti. Način opisa, kot je predstavljen v tem zgledu, ustreza v realnih projektih prvemu prehodu skozi specificiranje, kjer se postavlja struktura dekompozicije in osnovne funkcije posameznih algoritmov.

Postopek analize je izveden s sistemsko dekompozicijo, ki poteka od najvišjega nivoja abstrakcije do opisov podrobnosti in vseskozi sledi s številkami označeni nivojski strukturi. Postopek pričenjamo z določitvijo kontekstnega diagrama (Sl. 7.164). Kontekstni diagram razmejuje sistem, to je proces mešanja, ki ga analiziramo, od okolice. Okolico v tem primeru predstavljajo procesne naprave, na katerih se izvaja proces in ekipa, ki ta proces vodi. Obravnavani proces komunicira z okolico preko kontekstnih tokov. Tokovi, ponazorjeni s polnimi črtami prikazujejo podatkovne tokove, črtkane črte pa kontrolne tokove. Vsi kontekstni tokovi so neelementarni, kar pomeni, da so sestavljeni iz komponent. Sestava kontekstnih tokov je prikazana v nadaljevanju. Oznaka treh pik s srednjo počrnjeno pri simbolu za kontekstni proces pomeni, da je ta simbol še nadalje dekomponiran z diagramom.



Sl. 7.164. Kontekstni diagram procesa mešanja komponent

Diagram, s katerim je dekomponiran kontekstni proces, imenujemo sistemski diagram (Sl. 7.165).



Sl. 7.165. Sistemski diagram procesa mešanja komponent

Na njem je prikazana logična razdelitev tehnološkega procesa na posamezne postopke in podatkovne ter kontrolne povezave med njimi. Na sliki najdemo štiri simbole za podatkovne procese in enega za kontrolni proces. Podatkovni proces *Nadzoruj rezervoarje* je stalen proces in se izvaja neodvisno od ostalih procesov. Ostali trije

podatkovni procesi se izvajajo le takrat, kada jih omogoča kontrolni proces *Vodi proces mešanja*. Ta proces jih omogoča ali onemogoča s promptnim tokom *E/D* (Enable/Disable). Kontrolna logika je skrita v notranjosti kontrolnega procesa in je pojasnjena s pripadajočim diagramom prehajanja stanj. Na sistemskem diagramu opazimo tako zvezne podatkovne tokove, prikazane z dvojnimi puščicami, kot tudi diskretne (prikazane z enojnimi puščicami). Prvi ponazarjajo stalno razpoložljiv podatek, drugi pa podatek ob določenem času. Počrnjene puščice pomenijo, da imajo ti podatkovni tokovi vir in ponor na simbolih na diagramu, medtem ko nepočrnjene pomenijo, da je vir ali ponor na diagramu višjega ali nižjega nivoja.

Na sistemskem diagramu vidimo, da je podatkovni proces *Nadzoruj rezervoarje* enostaven proces, katerega ni potrebno nadalje dekomponirati z novim diagramom. Zadostuje že pisna specifikacija, to je opis z besedilom, ki pa mora tudi slediti določenim zakonitostim. V našem zgledu je uporabljena specifikacija s psevdokodo (Sl. 7.166).

```
Proces Nadzoruj rezervoarje
REPEAT FOREVER
BEGIN
  IF 'nivo v rezervoarjih' =< minimalni nivo
          THEN 'nizek nivo v rezervoarjih'
  IF 'nivo v rezervoarjih' => delovni nivo
          THEN 'visok nivo v rezervoarjih'
  IF nadozorovane procesne veličine ali naprave izven meja pravilnega delovanja
          THEN 'alarmi rezervoarjev'
  IF 'temperatura Akompona' <>referenčna temperatura +/- prag
          THEN 'krmiljenje grelca' = vključeno
          ELSE  'krmiljenje grelca' = izključeno
  'viskoznost' = f ( 'temperatura Akompona')
END
```

Sl. 7.166. Pisna specifikacija procesa Nadzoruj rezervoarje

Podatki, ki preko podatkovnih ali kontrolnih tokov vstopajo ali izstopajo v proces, so pisani v narekovajih.

Tudi podatkovni proces *Doziraj komponente* je enostaven proces, zato je tudi ta pojasnjen s pisno specifikacijo (Sl. 7.167). Za razliko od prejšnjega, je ta aktiven le, kadar ga omogoči kontrolni proces s promtnim tokom *E/D Doziraj komponente*. Vsi promptni tokovi tipa E/D so dejansko sestavljeni iz dveh komponent, omogoči (E) in onemogoči (D) podatkovni proces, kjer prva proces aktivira, druga pa deaktivira.

```
Proces Doziraj komponente
WHEN    'E/D Doziraj komponente' = E
REPEAT FOREVER
          BEGIN
          IF 'teža Skompona' < 'količina Skompona'
                    THEN 'krmiljenje dozirnega traka' = vključeno
                     ELSE 'krmiljenje dozirnega traka' = izključeno'
          IF 'nivo Akompona'  < 'količina Akompona'
                    THEN 'krmiljenje ventila Akompona' = vključeno
                    ELSE 'krmiljenje ventila Akompona' = izključeno'
          IF 'nivo Bkompona'  < 'količina Bkompona'
                    THEN 'krmiljenje ventila Bkompona' = vključeno
                     ELSE 'krmiljenje ventila Bkompona' = izključeno'
          IF                          ' teža Skompona' = 'količina Skompona'
                    AND  'nivo Akompona'  = 'količina Akompona'
                    AND  'nivo Bkompona'  = 'količina Bkompona'
                     THEN 'komponente zdozirane'
           IF nadzorovane procesne veličine ali naprave izven meja pravilnega
delovanja
                     THEN 'alarmi doziranja'
          END
```

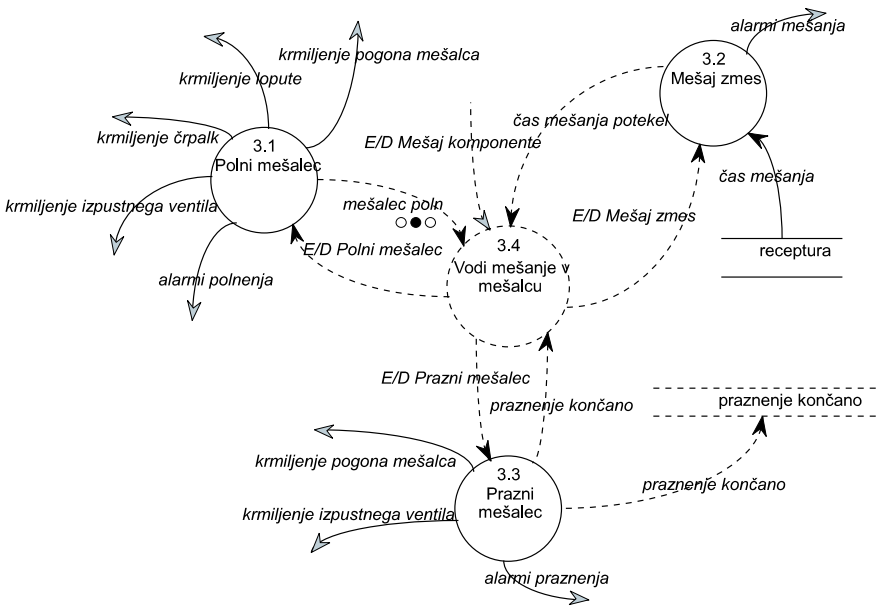
Sl. 7.167. Pisna specifikacija procesa Doziraj komponente

Podatkovni proces *Spremeni recepturo* je enostaven trigerski proces (Sl. 7.168). To pomeni, da ga aktivira kontrolni proces, in ko se celoten algoritem zapisan s pisno specifikacijo izvrši, se samodejno deaktivira.

```
Proces Spremeni recepturo
WHEN   'T Spremeni recepturo'
REPEAT FOREVER
          BEGIN
                    'receptura' = 'nova receptura'
          END
```

Sl. 7.168. Pisna specifikacija procesa Spremeni recepturo

Proces *Mešaj komponente* je dekomponiran z diagramom toka podatkov (Sl. 7.169). Na njem je prikazana še podrobnejša razdelitev postopkov, podatkovnih in kontrolnih tokov ter mehanizma izvajanja postopkov.



Sl. 7.169. Dekompozicija podatkovnega procesa Mešaj komponente

Pisne specifikacije za osnovne procese s tega diagrama prikazujejo slike (Sl. 7.170, Sl. 7.171, Sl. 7.172).

Proces Polni mešalec
WHEN 'E/D Polni mešalec' = E
'krmiljenje izpustnega ventila' = izključeno
'krmiljenje pogona mešalca' = vključeno
'krmiljenje lopute' = vključeno
'krmiljenje črpalk' = vključeno
REPEAT FOREVER
BEGIN
IF 'mešalec poln'
THEN
'krmiljenje pogona mešalca' = izključeno
'krmiljenje lopute' = izključeno
'krmiljenje črpalk' = izključeno
IF nadozorovane procesne veličine ali naprave izven meja pravilnega delovanja
THEN 'alarmi polnenja'
END

Sl. 7.170. Pisna specifikacija procesa Polni mešalec

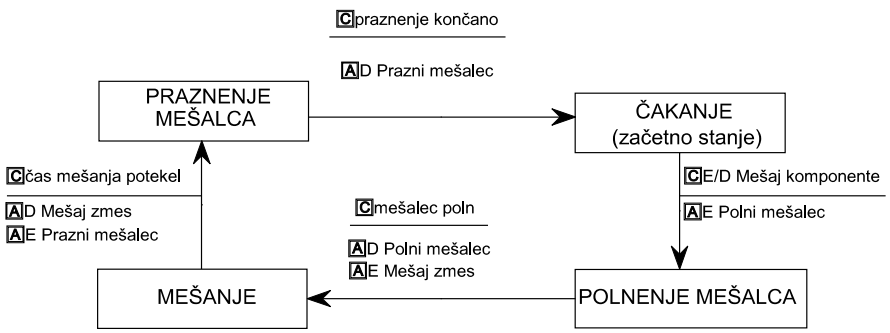
```
Proces Mešaj zmes
WHEN    'E/D Mešaj zmes' = E
tm = 'čas mešanja'
REPEAT FOREVER
            BEGIN
             IF tm potekel
                        THEN 'čas mešanja potekel'
             IF nadzorovane procesne veličine ali naprave izven meja pravilnega
delovanja
                        THEN 'alarmi mešanja'
END
```

Sl. 7.171. Pisna specifikacija procesa Mešaj zmes

```
Proces Prazni mešalec
WHEN    'E/D Prazni mešalec' = E
 'krmiljenje izpustnega ventila' = vključeno
REPEAT FOREVER
            BEGIN
                        IF vsa zmes iztekla iz mešalca
                            THEN 'praznenje končano'
                         'krmiljenje izpustnega ventila' = izključeno
                         'krmiljenje pogona mešalca' = izključeno
                        IF nadzorovane procesne veličine ali naprave izven meja
pravilnega delovanja
                            THEN 'alarmi praznenja'
END
```

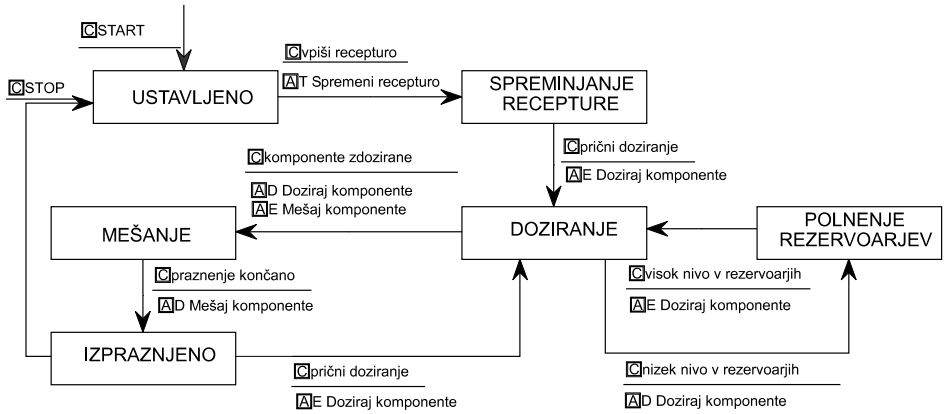
Sl. 7.172. Pisna specifikacija procesa Prazni mešalec

Kontrolni proces *Vodi mešanje v mešalcu* je pojasnjen z diagramom prehajanja stanj (Sl. 7.173). Ker v našem primeru kontrolni proces dejansko določa vrstni red izvajanja operacij šarže, ki pa se izvajajo le zaporedno, jih lahko ponazorimo s stanji. Če temu ne bi bilo tako, bi morali v diagramu prehajnja stanj definirati nova stanja, ki bi bila kombinacije vzporednih stanj oziroma operacij. Na diagramu je prikazano začetno stanje, iz katerega se začenjajo prehodi v druga stanja, ko je omogočen proces *Mešaj komponente*. Hkrati je to tudi končno stanje, v katerega preide proces po zadnjem aktivnem stanju. Na diagramu so s simbolom C prikazani pogoji za prehode, s simbolom A pa akcije, ki se ob prehodih izvedejo.



Sl. 7.173. Dekompozicija kontrolnega procesa Vodi mešanje v mešalcu *z diagramom prehajanja stanj*

Ostane nam samo še kontrolni proces s številko 5 na sistemskem diagramu (Sl. 7.165) imenovan *Vodi proces mešanja*. Ta kontrolni proces določa časovno izvajanje podatkovnih procesov šaržnega dela analiziranega tehnološkega procesa, ki je sestavljen iz stalnega dela (podatkovni proces *Nadzoruj rezervoarje*) in šaržnega dela (ostali podatkovni procesi). Mehanizem omogočanja in onemogočanja postopkov je prikazan na diagramu prehajanja stanj (Sl. 7.174). Tudi v tem primeru se operacije šarže izvajajo le zaporedno, zato smo jih tudi ponazorili s stanji.



Sl. 7.174 Dekompozicija kontrolnega procesa Vodi proces mešanja

Drugačno predstavitev mehanizma omogočanja in onemogočanja postopkov za isti kontrolni proces prikazuje tabela prehajanja stanj (Tabela 7.12).

Tabela 7.12.Tabela prehajanja stanj za kontrolni proces Vodi proces mešanja

	START	STOP	vpiši recepturo	prični dozira nje	nizek nivo v rezervo-arjih	visok nivo v rezervo-arjih	praznjenje končano	komponente zdozirane
USTAVLJENO (1)	**x**	*	**2**	**x**	*	*	**x**	**x**
SPREMINJANJE RECEPTURE (2)	**x**	*	*	**3**	*	*	**x**	**x**
DOZIRANJE (3)	**x**	*	*	*	**4**	*	**x**	**5**
POLNENJE REZERVOARJEV (4)	**x**	*	*	**x**	*	**3**	**x**	**x**
MEŠANJE (5)	**x**	*	*	**x**	*	*	**6**	*
IZPRAZNJENO (6)	**x**	**1**	*	**3**	*	*	*	**x**

Prednosti tabele stanj pred diagramom so v tem, da se v tabeli razpozna ne samo sosledje prehodov pri določenih pogojih, ampak tudi pravilnost pojavitev teh pogojev. V osenčenih poljih tabele se nahajajo:

- številke, ki določajo naslednje stanje,
- znaki "*", ki pomenijo, da pojavitev določenega pogoja v določenem stanju nima nobenega vpliva in
- znaki "x", ki pomenijo, da se določeni pogoji ne smejo pojaviti v določenih stanjih.

Pri izvedbi je potrebno vse troje upoštevati.

Struktura podatkov

krmiljenje procesnih naprav
- *krmiljenje naprav doziranja*
  - *krmiljenje dozirnega traka*
  - *krmiljenje ventilov doziranja*
- *krmiljenje grelca*
- *krmiljenje naprav mešanja*
- *krmiljenje črpalk*
  - *krmiljenje izpustnega ventila*
  - *krmiljenje lopute*
  - *krmiljenje pogona mešalca*

merjene procesne spremenljivke
- *merjene spremenljivke doziranja*
  - *teža Skompona*
  - *nivo veziv v posodah*
- *merjene spremenljivke rezervoarjev*
  - *nivo v rezervoarjih*
- *temperatura Akompona*

nova receptura
- *komponente*
  - *količina Skompona*
  - *količina Akompona*
  - *količina Bkompona*
- *čas mešanja*

status procesa in naprav
- *alarmi doziranja*
- *alarmi mešanja*
- *alarmi rezervoarjev*
- *viskoznost*

ukazi operaterja
- *prični doziranje*
- *START*
- *STOP*

ukazi tehnologa
- *vpiši recepturo*

Sl. 7.175. Strukture kontekstnih podatkovnih tokov

Struktura podatkov je običajno opisana v slovarju podatkov (Sl. 7.175). Poleg strukture v slovarju podatkov sestavljene in elementarne podatke opremimo še s komentarji. Za naš zgled prikazujemo le strukturo kontekstnih podatkovnih tokov, ki pa so razčlenjeni do elementarnih tokov oziroma podatkov. Podatkovni tokovi v poudarjenem tisku so tisti, ki nastopajo na kontekstnem diagramu. S tehniko zamikov je prikazana celotna podatkovna struktura, ki je bila obravnavana v našem zgledu.

Δ

7.5.2 Metode domenskega inženirstva

Odkar je bila prvič omenjena leta 1968, je takoimenovana kriza programske opreme (*software crisis*) vedno znova obravnavana tema, še več, z naraščajočo kompleksnostjo sistemov postaja vedno bolj aktualna. To krizo bi lahko imenovali tudi *kriza integritete* programske opreme, njeni glavni simptomi pa so neskladnost programske opreme s potrebami uporabnika, nezanesljivost, previsoka cena, prekoračitev rokov pri izdelavi, nefleksibilnost, težko in drago vzdrževanje in nizka ponovna uporabljivost. V preteklosti je bilo tudi že veliko razprav o možnih načinih premagovanja te krize. Na seznamu kandidatov so bili strukturirano programiranje, modularno programiranje, objektno usmerjeni koncepti, razni pristopi k razvoju skozi življenjski cikel, itd. V zadnjem času se vsi strinjajo, da je ogromen potencial v ponovni uporabljivosti (kvalitetne) programske opreme in to ponovni uporabljivosti elementov iz celotnega življenjskega cikla.

7.5.2.1 Ponovna uporabljivost

Pri vsakem razvojnem delu je prisotna želja po ponovni uporabi rezultatov (inženirskih znanj in izdelkov) iz preteklega razvojnega dela. Pri večini razvojnih dejavnosti je to nekaj samoumevnega (gradbeništvo, arhitektura, strojništvo, ladjedelništvo, elektronika), pri programskem inženirstvu pa je bilo v kratki zgodovini te discipline več neuspehov kot uspehov na področju ponovne uporabljivosti. Danes se uspešno uvajanje sistematične ponovne uporabljivosti smatra za možno rešitev prej omenjene krize programske opreme. To potrjujejo najbolj uspešni primeri uvajanja ponovne uporabljivosti, pri katerih je bila stopnja ponovne uporabljivosti (razmerje ponovno uporabljene kode proti celotni kodi) tudi do 90 %.

Vrste (nivoji) ponovne uporabljivosti

Ponovna uporabljivost se pogosto začne, ker hoče programer skrajšati čas razvoja in enostavno "klonira" že napisano kodo. To je prvi začetni premik, ki tudi takoj da določene rezultate v smislu skrajšanega časa razvoja. Na žalost se s tem kmalu pojavijo problemi pri vzdrževanju vedno večjega števila variacij kode.

Naslednji korak je identifikacija nekaterih splošno uporabnih komponent, ki jih gledamo kot črne škatle (zanima nas kaj in ne kako delajo). Komponenta je že zgrajena enota, navadno majhne granularnosti, ki služi za povečanje moči konstruktov programskega jezika. Glavne vrste komponent so bile dolgo procedure in funkcije za numerične in statistične aplikacije ter komponente, ki se uporabljajo pri računalniških algoritmih, kot so bufferji, vrste, liste in drevesa. V novejšem času so to razni elementi uporabniških vmesnikov kot so okna, ikone ali drsniki. Komponente shranjujemo v knjižnice, obstajajo pa tudi posebni programi - browserji - za iskanje ustreznih komponent.

S časom se pojavi nova dilema. Ali uporabiti novo ali staro verzijo komponente? Ali je potrebno spremeniti obstoječi sistem? Za uspešno obvladovanje takih odločitev je

potrebno zgraditi k ponovni uporabljivosti usmerjeno (*reuse-oriented*) organizacijo, z vgrajenimi mehanizmi zagotavljanja kvalitete (*quality assurance - QA*) in upravljanja konfiguracije (*configuration management - CM*). Na tej stopnji že lahko govorimo o sistematični ponovni uporabljivosti.

Kmalu se pokaže, da ima ponovna uporabljivost na nivoju kode omejen doseg. Ponovno uporabljivi izdelki iz zgodnjih faz so na višjem nivoju abstrakcije in z večjo granularnostjo, iz česar sledi tudi večja izrazna moč in večja korist ob njihovi ponovni uporabi.

Naslednji korak od načrtovanja za ponovno uporabljivost je analiza za ponovno uporabljivost. Pri tem so potencialno ponovno uporabljivi celi modeli, arhitekture in entitete iz določene domene na najvišjem nivoju abstrakcije.

Kritični faktorji uspeha uvajanja ponovne uporabljivosti

Kljub ogromnemu potencialu sistematične ponovne uporabljivosti je vse preveč neuspešnih primerov njenega uvajanja, saj gre za težaven in tvegan proces. Pri tem se srečujemo z množico problemov, in to tako tehničnih kot netehničnih. Sistematična ponovna uporabljivost zahteva dolgoročno podporo vodstva, saj lahko traja leta, preden se pokažejo pozitivni ekonomski rezultati. To zahteva spremembo paradigme v smislu premika horizonta od tekočih projektov na sistematično načrtovanje za ponovno uporabljivost. Prav tako so potrebne takšne spremembe v organizaciji, ki so izvedljive le ob podpori vodstva. Glavne naloge, ki jih vodstvo mora izpeljati, če naj bo uvajanje uspešno, so izobraževanje, spodbujanje ponovne uporabljivosti, uvajanje sprememb v kulturi organizacije ter obravnavanje ponovne uporabljivosti kot ekonomske kategorije.

Od tehničnih faktorjev je ključna usmeritev na ponovno uporabljivost v zgodnjih fazah življenjskega cikla. V tem smislu je potrebno izbrati ustrezno domeno ter sistematično razvijati ponovno uporabljive elemente na osnovi domenskega inženirstva.

Potrebna je tudi določena zrelost procesa razvoja programske opreme. Organizacija, ki nima uvedenega sistematičnega pristopa k razvoju bo veliko težje uspešno uvedla ponovno uporabljivost, kar še posebej velja za ponovno uporabljivost v zgodnjih fazah življenjskega cikla. Brez sistematičnega procesa je težko zagotoviti, da bo ponovna uporabljivost obravnavana na ustrezen način in v ustreznem trenutku. Poleg tega bodo entitete, ki naj bi jih ponovno uporabljali, verjetno premalo konsistentne za graditev zanesljivih baz za ponovno uporabljivost. Lahko bi rekli, da mora biti organizacija, če hoče uspešno uvesti ponovno uporabljivost, na vsaj drugi stopnji zrelosti procesa izgradnje programske opreme. Iz tega sledi tudi to, da mora biti program ponovne uporabljivosti trdno vpet v ostale aktivnosti za izboljšanje procesa, kar predpostavlja obstoj strateškega načrta, ki vključuje ponovno uporabljivost.

7.5.2.2 Programsko inženirstvo na osnovi modelov

Programsko inženirstvo na osnovi modelov (*Model Based Software Engineering - MBSE*) (Software Engineering Institute, 1998) predstavlja širši okvir za ponovno uporabljivost v okvirju določene domene. Domeno lahko definiramo kot omejeni realni, hipotetični ali abstraktni svet, ki je naseljen z množico konceptualnih entitet, ki se obnašajo skladno s pravili značilnimi za to domeno. Če to definicijo malo poenostavimo, lahko rečemo, da je domena *družina podobnih izdelkov ali sistemov*.

Cilj MBSE je skrajšati čas izdelave sistemov iz domene ter izboljšati njihovo kvaliteto in možnost vzdrževanja:

- s formalnim razumevanjem in sistematičnim ter zanesljivim zajemanjem in abstrahiranjem informacije o lastnostih in strukturi družine izdelkov;
- z razvojem ponovno uporabljivih virov, ki poenostavljajo razvoj novih izdelkov iz družine.

Ti ponovno uporabljivi viri morajo biti načrtovani tako, da bi bili čim bolj fleksibilni, to je neobčutljivi na spremembe v funkcionalnosti, zmogljivosti ali tehnologiji. Večji del tehnologije, ki se uporablja v MBSE, ni nov, temveč gre za fokusiranje na modele, ki utrjujejo razumevanje družine programskih izdelkov, kar omogoča izgradnjo ponovno uporabljivih zbirk. Pri tem se opiramo na dva principa:

1. Ponovno uporabljiva informacija je specifična za določeno domeno in ne univerzalna.
2. Kohezivnost in stabilnost domene. Informacijo se splača zbirati le, če je dovolj kohezivna in stabilna. Stabilnost omogoča porazdelitev cene zajemanja in abstrahiranja informacije na zadostno število uporabnikov te informacije.

Elementi MBSE

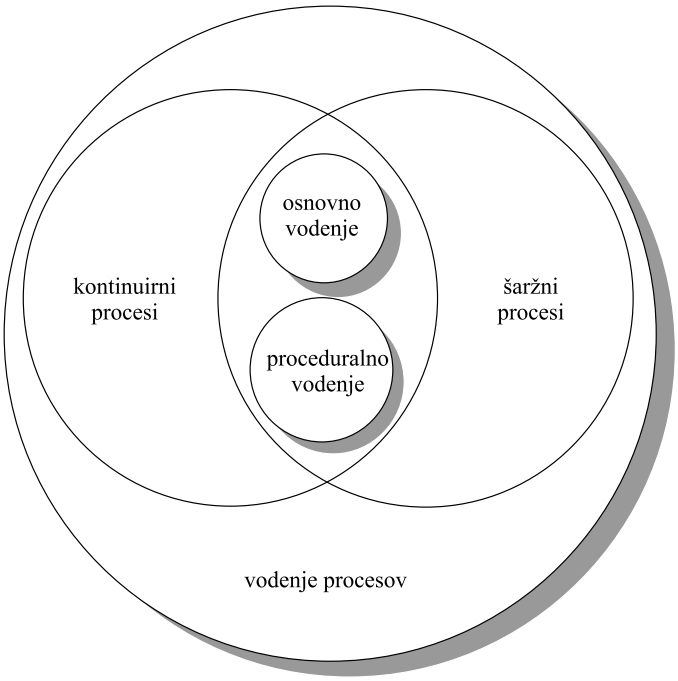
MBSE sestoji iz naslednjih elementov:

- družine izdelkov – domene;
- produkcijskega sistema;
- baze znanja o domeni;
- inženirskega procesa.

Domena

MBSE lahko zajema eno ali več domen in/ali poddomen. Kot primer domene z večjim številom poddomen lahko vzamemo domeno vodenja (nekosovnih) industrijskih procesov, ki se deli na poddomeni vodenja kontinuirnih in šaržnih procesov, kot to prikazuje Sl. 7.176. Vsaka od teh (pod)domen se zopet deli na poddomene, na primer poddomeno osnovnega vodenja (doseganje in vzdrževanje določenega stanja procesne opreme ali procesa) in poddomeno proceduralnega vodenja (izvajanje procesno

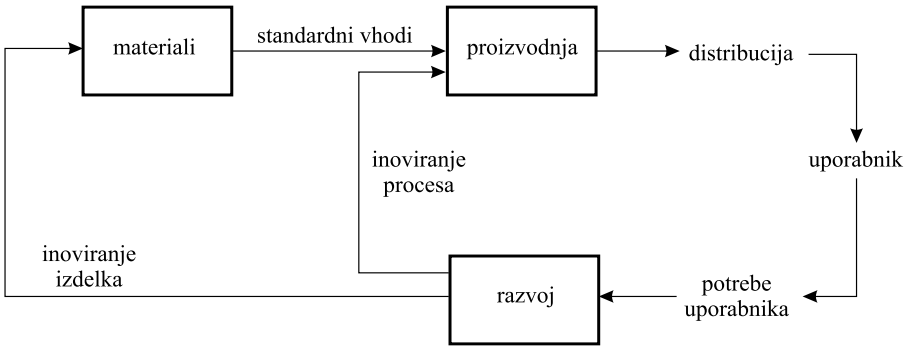
usmerjenih aktivnosti).



Sl. 7.176. Domena vodenja procesov in njene poddomene

Produkcijski sistem

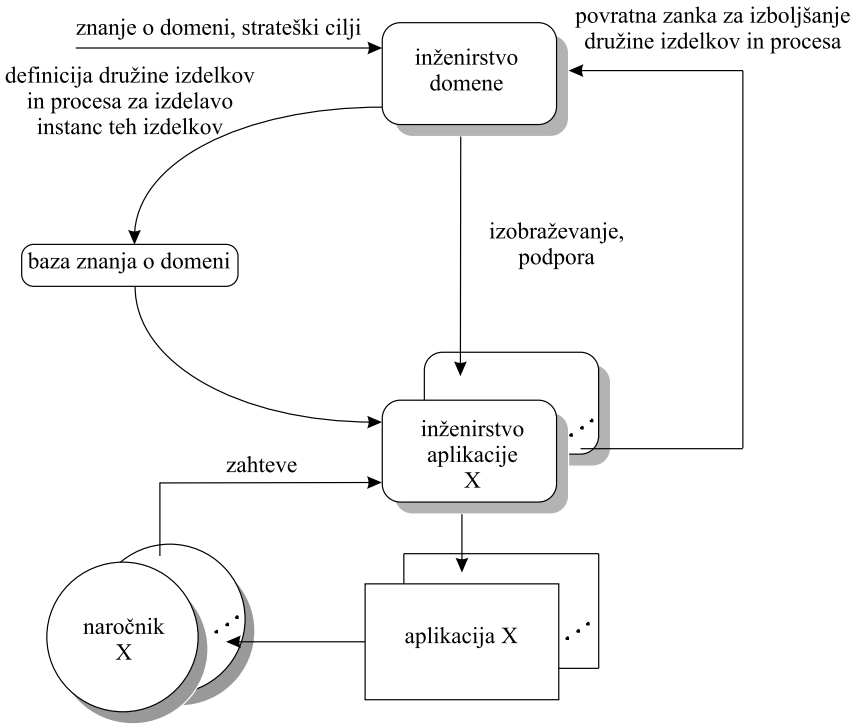
Če hočemo obvladovati spremembe in razvijati tehnologijo, mora proces razvoja programske opreme vključevati domeno in biti organiziran kot produkcijski sistem, ki ga prikazuje Sl. 7.177.



Sl. 7.177. Elementi produkcijskega sistema

“Potrebe uporabnika” vključujejo analizo trga, analizo potreb, definicijo izdelkov in strateško planiranje. Gre predvsem za identifikacijo lastnosti izdelka, ki povečujejo njegovo vrednost za uporabnike.

MBSE predstavlja produkcijski sistem, ki ga prikazuje Sl. 7.178.



Sl. 7.178. Produkcijski sistem MBSE

Na osnovi takega pogleda lahko rečemo, da MBSE predstavlja premik paradigme od obrtniške graditve posameznih unikatnih izdelkov, ki jih izdelujejo vešči nepovezani profesionalci, k graditvi družin podobnih izdelkov v “tovarni” programske opreme, ob uporabi standardiziranega procesa in komponent, ob visoki specializaciji, visoki ponovni uporabljivosti, načrtovanju za ponovno uporabljivost in motivaciji za ponovno uporabljivost.

Baza znanja o domeni

Pri MBSE gre v bistvu za polnjenje baze znanja o domeni in uporabo tega znanja. Bazo znanja o domeni sestavljajo naslednji elementi:

- konceptualni model domene, ki opisuje podobnosti in razlike med posameznimi izdelki v domeni (lastnosti, scenariji uporabe);

- arhitektura, ki opisuje možne elemente programske opreme in njihove medsebojne relacije ter porazdelitev funkcionalnosti med elementi;
- ponovno uporabljive komponente, ki izhajajo iz podobnosti med izdelki v domeni, izdelujejo pa se za vse faze življenjskega cikla in ne samo za kodiranje;
- definicija procesa, ki predstavlja niz korakov pri graditvi izdelka, rezultira pa iz razlik med sistemi v domeni;
- podpora procesu v smislu avtomatizacije ponovne uporabljivosti (za vse faze življenjskega cikla);
- standardi za razvoj, testiranje in integracijo.

Inženirski proces

Kot je vidno iz Sl. 7.178, pri produkcijskem sistemu MBSE obstajata dve inženirski skupini (aktivnosti):

- domensko inženirstvo, ki polni bazo znanja o domeni in
- aplikacijsko inženirstvo, ki iz in s pomočjo elementov baze znanja o domeni razvija izdelke iz domene.

Skupini sta si med sabo komplementarni. Tako na primer v fazi analize zahtev, skupina domenskega inženirstva definira abstrakcije za družino izdelkov, medtem ko aplikacijski inženirji identificirajo zahteve za izdelek.

Interakcija med skupinami služi za preverjanje kvalitete obeh. Baza znanja se sproti dopolnjuje ali popravlja, seveda ob izvajanju aktivnosti zagotavljanja kvalitete (QA) in upravljanja s spremembami (CM).

Različne vrste nalog zahtevajo različna znanja in veščine pripadnikov obeh skupin. Aplikacijski inženirji so dobri analitiki in se v glavnem ukvarjajo s pragmatičnimi spremembami elementov iz baze znanja pri njihovi konkretni uporabi, seveda ob izvajanju aktivnosti QA/CM. Izdelujejo torej konkretne sisteme in validirajo rezultate.

Domenski inženirji morajo imeti močno izražene sposobnosti sinteze. V modele vgrajujejo različne lastnosti izdelkov, načrtovalske odločitve ter omejitve in implementacijske parametre. Ukvarjajo se z razkrivanjem informacije o sistemih, pri čemer unificirajo posamezne rešitve in probleme ter izvajajo sintezo abstraktnih specifikacij in načrtov.

Bistvena razlika med skupinama je v načinu njihovega ocenjevanja in nagrajevanja. Aplikacijske inženirje nagrajujemo glede na razumevanje in zadovoljevanje uporabnikovih potreb, to je za pravočasno izdelavo kvalitetnih izdelkov. Domenske inženirje pa nagrajujemo za tehnične inovacije, ki dajejo organizaciji prednost pred tekmeci ter za fleksibilne elemente baze znanja, ki jih aplikacijski inženirji znajo uporabljati.

7.5.2.3 Domensko inženirstvo

Domensko inženirstvo pokriva vse aktivnosti za polnjenje baze znanja o domeni. Te aktivnosti so:

1. Analiza domene, v okviru katere identificiramo (na osnovi podobnosti med sistemi v domeni) eno ali več domen ter definiramo razlike med sistemi v domeni.
2. Načrtovanje domene, v okviru katere zgradimo prilagodljivi načrt domene, oziroma generično arhitekturo.
3. Implementacija domene, v okviru katere definiramo mehanizme za preslikavo zahtev aplikacijskih projektov v sisteme zgrajene iz ponovno uporabljivih komponent.

Analiza domene

Analiza domene je proces identifikacije, zbiranja, urejanja in abstrahiranja relevantnih informacij v domeni, na osnovi študija obstoječih sistemov in zgodovine njihovega razvoja, na osnovi znanja ekspertov v domeni in pripadajoče teorije ter s pomočjo tehnik informacijskega modeliranja.

Prvi korak v analizi domene je kontekstna analiza, med katero domeno natančno omejimo in določimo njene relacije z ostalimi domenami.

Drugi korak v analizi domene je modeliranje domene. V tem koraku obravnavamo podobnosti in razlike med sistemi v domeni, uredimo razumevanje relacij med entitetami v domeni ter to razumevanje abstrahiramo na razvijalcem programske opreme uporaben način. S sistematičnim prikazovanjem funkcij, objektov (entitet), podatkov in relacij, modeliranje domene določa kaj so aplikacije, kaj delajo in kako delajo. Rezultat modeliranja domene je razumevanje:

- lastnosti programske opreme v domeni;
- standardne terminologije ekspertov v domeni (slovar domene);
- dokumentacije o informaciji (entitetah) v domeni;
- notacije izražanja zahtev programske opreme

Načrtovanje domene

Načrtovanje domene je proces izdelave modela načrtovanja iz konceptualnega modela analize domene. Ob tem se uporablja znanja o ponovni uporabljivosti zahtev in načrtovanja ter o *generičnih arhitekturah*. Model načrtovanja predstavlja generično arhitekturo, ki je dovolj splošna, da velja za večino sistemov v domeni, in da podpira načrtovanje za ponovno uporabljivost. Generična arhitektura je predstavljena z arhitekturnim stilom, ki je najprimernejši za družino izdelkov. Ta arhitektura predstavlja izvedbeno platformo za vse aplikacije v družini in definira navodila za izdelavo generičnega načrta. Gre torej za domensko specifično arhitekturo programske opreme, katere osnova so znane ali predvidene podobnosti in razlike med sistemi v domeni. To je

komponenta načrtovanja za ponovno uporabljivost, ki zagotavlja osnovo za ostale ponovno uporabljive komponente načrtovanja in kodiranja v implementaciji domene ter za podrobno načrtovanje izdelkov v aplikacijah.

Cilj arhitekturnega modeliranja je identificiranje sočasnih procesov (aktivnosti) in aplikacijsko usmerjenih modulov ter alokacija lastnosti, funkcij in podatkov tem procesom in modulom. Arhitekturni model definira osnovne gradnike aplikacije in model koordinacije med njimi, ki je potrebna pri izdelavi aplikacij.

Najprimernejša strategija dekompozicije generične arhitekture je dekompozicija na relevantne entitete iz problemske domene.

Model koordinacije definira način aktiviranja in komunikacije za elemente programske opreme - tipi interakcij med entitetami in mehanizmi teh interakcij.

Implementacija domene

Implementacija domene je proces identifikacije ponovno uporabljivih komponent na osnovi modela domene in generične arhitekture. Domenski inženirji v tej fazi izdelujejo knjižnice komponent iz novih ali obstoječih ponovno uporabljivih zbirk. Ob tem uporabljajo znanja o domeni, pridobljena med analizo domene in generične arhitekture, razvite med načrtovanjem domene. Poleg teh ponovno uporabljivih komponent so izhodi iz implementacije domene še aplikacijski generatorji in domensko usmerjeni jeziki. Za uspešno izgradnjo, upravljanje in vzdrževanje knjižnice ponovno uporabljivega znanja o domeni je zelo pomembno stalno izvajanje aktivnosti QA/CM.

Relacija med domenskim in aplikacijskim inženirstvom

Domensko in aplikacijsko inženirstvo sta komplementarna, interaktivna sočasna procesa, ki tvorita produkcijski sistem programske opreme, usmerjen proti ponovni uporabljivosti in delujoč na osnovi modelov programske opreme. Proces aplikacijskega inženirstva razvija programsko opremo iz elementov baze znanja o domeni, ki jo polni proces domenskega inženirstva.

Fokus aplikacijskega inženirstva je posamezen sistem, domenskega inženirstva pa množica podobnih sistemov znotraj domene. Aplikacijsko inženirstvo uporablja:

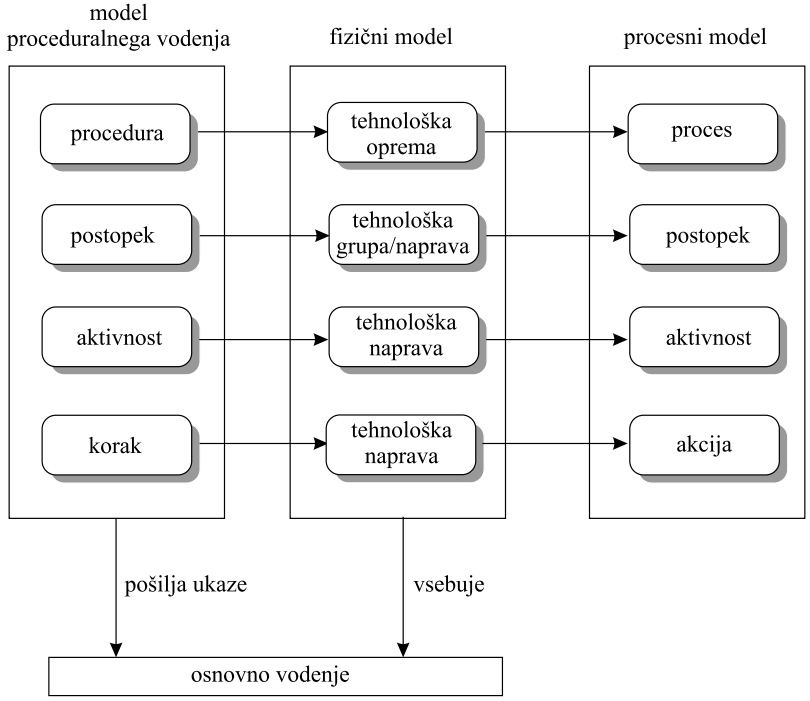
- model domene za identifikacijo zahtev uporabnika;
- model načrtovanja za specificiranje konfiguracije izdelka;
- strategijo dekompozicije in model koordinacije (arhitekturni stil) kot vodilo razvoja;
- aplikacijske generatorje in programske komponente za izdelavo aplikacijske kode.

∇

Primer 7.26: *Domensko inženirstvo vodenja kontinuirnih procesov*

Kot primer bomo vzeli domensko inženirstvo vodenja kontinuirnih procesov. Podali

bomo del modela analize ter shemo generične arhitekture načrtovanja. Sl. 7.179 prikazuje tri modele, skozi katere lahko gledamo na kontinuirni proces: model proceduralnega vodenja, fizični model in procesni model.



Sl. 7.179. Modeli vodenja kontinuirnih procesov

Model smo privzeli iz standarda o vodenju šaržnih procesov (ANSI/ISA-S88.01, 1995) in ga prilagodili za kontinuirne procese. Ti trije modeli se med izvajanjem domenskega inženirstva preslikajo v modela analize in načrtovanja.

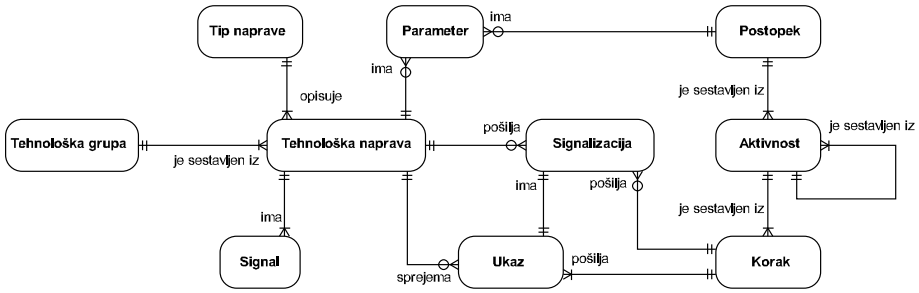
Analiza domene

Fizični model predstavlja dekompozicijo tehnološke opreme na tehnološke grupe in tehnološke naprave. Model proceduralnega vodenja sestoji iz procedure, ki jo dekomponiramo na *postopke*. Postopek je proceduralni element, ki definira izvajanje določene večje procesne aktivnosti. Le-ta praviloma rezultira v fizikalnih ali kemijskih spremembah snovi ali energije. Vsak postopek nadalje dekomponiramo na sočasne ali sekvenčne *aktivnosti*. Tipično je to dekompozicija na *stanja*, v katerih se nahaja postopek, to pa so *zaganjanje*, *obratovanje* (stabilno ali s prehodi med večjim številom podstanj) ter *ustavljanje*. Med vsako aktivnostjo se izvajajo posamezni *koraki,* ki izvajajo procesno usmerjene *akcije* tako, da pošiljajo ukaze *v osnovno vodenje*. Osnovno vodenje služi za doseganje in vzdrževanje določenega stanja procesne opreme ali procesa. Tukaj nastopajo regulacije, krmiljenja, nadzor, alarmiranje. Način dekompozicije procesa na postopke je definiran z tehnološkimi zahtevami in

omejitvami. Ob dekomponiranju se upoštevajo tudi kriteriji dobre modularizacije (visoka notranja povezanost entitet in nizka sklopljenost med entitetami) zaradi obvladovanja kompleksnosti razvoja, uporabe in vzdrževanja sistema vodenja. Sl. 7.180 prikazuje relacije med entitetami fizičnega modela in entitetami modela proceduralnega vodenja.

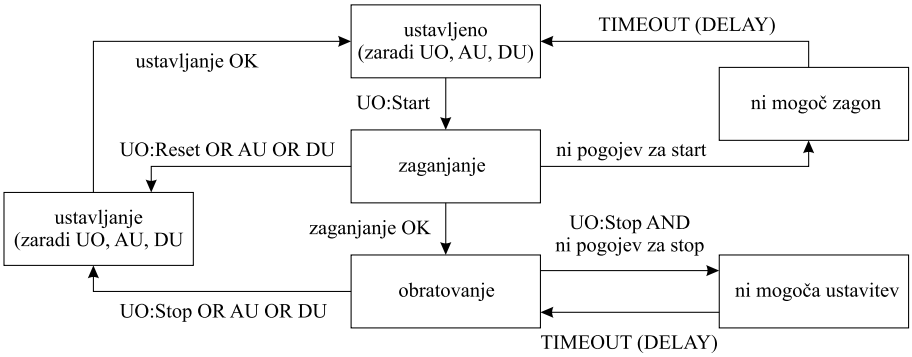
Za opisovanje postopkov uporabljamo abstrakcijo razširjenih sočasnih končnih avtomatov, ki imajo naslednje atribute:

- stanja;
- akcije med stanji;
- vzroki prehodov stanj (npr. ukazi operaterja, avtomatski prehodi stanj, domino prehodi stanj),
- pogoji prehodov stanj;
- prehodi stanj;
- akcije med prehodi stanj;
- relacije z drugimi postopki (razširjenimi sočasnimi končnimi avtomati).



Sl. 7.180. Relacije med entitetami vodenja

Stanje postopka je lahko tudi *agregatno stanje*, to je unija večjega števila elementarnih stanj (npr. unija vseh stanj razen stanja obratovanje - postopek ni v stanju obratovanje).

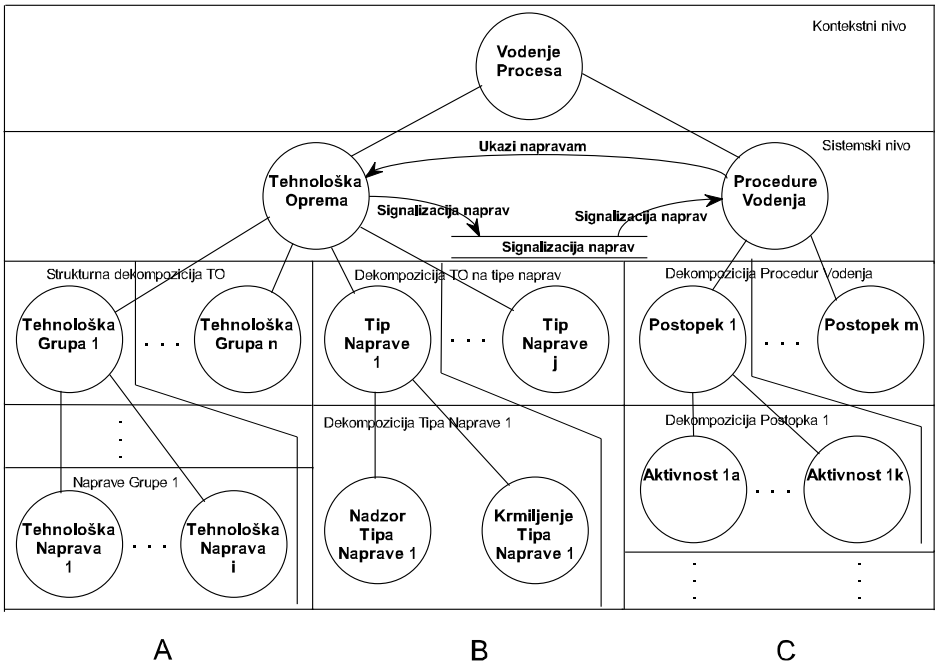


Sl. 7.181. Diagram prehajanja stanj kontinuirnega postopka.

Sl. 7.181 podaja najenostavnejši model prehajanja stanj kontinuirnega postopka.

Načrt domene

Načrt domene je predstavljen z generično arhitekturo specifikacij, ki je realizirana kot razširitev strukturne Ward-Mellor metode za specificiranje sistemov realnega časa. Razširitev je v spremenjeni semantiki 'mehurja', ki ne pomeni procesa, kot v originalni metodi Ward-Mellor, temveč ima različne pomene, glede na del arhitekturnega modela v katerem nastopa. Generična arhitektura sestoji iz dveh delov.



Sl. 7.182. Generična arhitektura načrtovanja

Prvi je del *entitet tehnološke opreme* (ustrezno fizičnem modelu), ki ga predstavlja 'mehur' *Tehnološka Oprema* na sistemskem nivoju. Ta del je dekomponiran na dva načina glede na dva aspekta tehnološke opreme: *strukturni* (agregacijski) aspekt in *klasifikacijski* aspekt. V strukturnem delu, ki ga opisuje del A na Sl. 7.182, gre za dekompozicijo (v skladu z fizičnim modelom) na *tehnološke grupe* in *tehnološke naprave*, pri čemer so posamezne grupe in naprave predstavljene z 'mehurji'. V klasifikacijskem delu, ki ga opisuje del B na Sl. 7.182, pa nastopa samo en nivo dekompozicije, na katerem nastopi po en 'mehur' za vsak tip tehnološke naprave.

Drugi del generične arhitekture je del *entitet proceduralnega vodenja* (ustrezno modelu proceduralnega vodenja) in ga prikazuje del C na Sl. 7.182.

Komunikacija med entitetami proceduralnega vodenja (postopki) in entitetami tehnološke opreme poteka preko ukazov napravam in signalizacije naprav, kot je to razvidno na sistemskem nivoju na Sl. 7.182.

Ob zaključku tega podpoglavja lahko še enkrat ugotovimo, da domensko inženirstvo je realizacija principa "načrtovanje za ponovno uporabljivost" (*design-for-reuse*), aplikacijsko inženirstvo pa principa "načrtovanja s ponovno uporabljivostjo" (*design-by-reuse*). Domensko inženirstvo, oziroma širše, programsko inženirstvo na osnovi modelov, se v zadnjih letih vedno bolj uveljavlja kot najbolj perspektivno sredstvo za uspešno sistematično ponovno uporabljivost.

7.6 Signali v sistemih za vodenje

V dosedanjem delu tega poglavja smo se ukvarjali predvsem z metodami in postopki za načrtovanje funkcij vodenja in njihovo programsko realizacijo. Pričujoče podpoglavje pa je namenjeno znanjem in veščinam, ki jih potrebujemo, da sisteme vodenja povežemo s fizičnim svetom. Gre predvsem za probleme zajemanja, pretvorbe in različnih načinov prenosa signalov.

V poglavju bo poudarek predvsem na obravnavi različnih vrst signalov, problematiki motenj in zaščite pred motnjami, analogno/digitalnimi in digitalno/analognimi pretvorbami signalov, različnimi načini prenosa digitalnih signalov ter opisom najpomembnejših komunikacijskih protokolov.

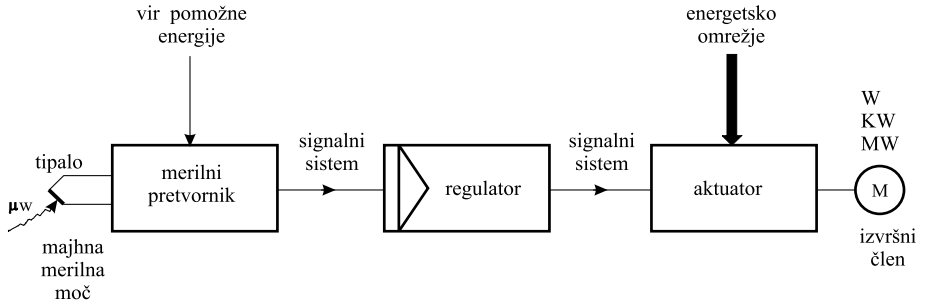
7.6.1 Zajemanje, pretvorba in prenos signalov

Le zelo redki regulacijski sistemi ne vsebujejo komponent za zajemanje, pretvorbo in prenos signalov. Tem ponavadi pravimo regulatorji brez pomožne energije. Tipična predstavnika sta termostatski radiatorski ventil in avtomatski regulacijski ventil za vodovodno omrežje. Pri prvem, na primer, zrak v sobi hladi ali greje v meh zaprto tekočino ali plin, ki se zaradi tega krči ali razteza. Meh pa je neposredno povezan s čepom, ki uravnava pretok tople vode skozi radiator. Celotni ventil je konstruiran tako kompaktno, da posamezni sestavni deli hkrati opravljajo več funkcij, zato bi bilo govoriti o zajemanju, pretvorbi ali prenosu signalov nesmiselno. Podobno velja tudi za drugega: izhodni tlak vode pritiska na membrano, prednapeto z vzmetjo. Membrana je mehansko povezana s čepom ventila, ki pri naraščanju izhodnega tlaka ustrezno pripira ventil.

V nasprotju z navedenima primeroma (ki sta sicer v množični proizvodnji) pa v procesni industriji in drugod prevladujejo sistemi za vodenje procesov, pri katerih so tipalo, izvršni člen, regulator ali procesni računalnik nameščeni na zelo različnih mestih. Med njimi so velike signalne razdalje, ki jih lahko merimo celo v kilometrih (npr. rafinerijska

polja, …). V teh primerih so signalne povezave med gradniki sistema za vodenje zelo pomembne in nemalokrat celo vplivajo na kvaliteto regulacijskega in proizvodnega procesa.

Praviloma želimo meriti procesne (fizikalne) veličine zelo natančno. Da bi to dosegli, moramo že pri izbiri in montaži tipala paziti, da le-to čim manj vpliva na merjeni proces. Termočlen, na primer, lahko povzroča odvajanje toplote preko zaščitnega plašča in priključnih žic. Večina tipal za pretok povzroča dodaten padec tlaka. Nizka vhodna upornost ojačevalnika napetosti redox lahko izhodno napetost močno zniža. Večina merilnih postopkov tako ali drugače generira izhodne signale zelo majhnih moči. Kot taki pa seveda niso primerni za prenos na daljavo - potrebno jim je namreč ustrezno dvigniti signalno moč in jih oblikovati v tàko obliko, ki omogoča čim natančnejši in zanesljivejši prenos na daljavo (od nekaj m do nekaj km). To nalogo opravljajo merilni pretvorniki, ki za svoje delo potrebujejo zunanji vir energije.



Sl. 7.183. Močnostne razmere v tipični regulacijski zanki. Na strani tipala želimo čim manjši odvzem moči zaradi visoke natančnosti meritve, na strani izvršnega člena pa potrebujemo visoke moči. Njima v nasprotju pa se mora signalni sistem "upirati" zunanjim motnjam in slabljenju prenosnega medija.

Podobno velja tudi za krmiljenje izvršnih členov, vendar v obratnem smislu: moči, ki jih izvršni členi "upravljajo" so pogosto tako velike, da jih je nesmiselno prenašati po signalnem sistemu. Zaradi tega izvršnim členom prilagodimo aktuatorje, ki v sorazmerju z velikostjo krmilnega signala dovajajo izvršnim členom veliko moč iz energetskega omrežja. (glej Sl. 7.183).

V nasprotju z navedenima "končnima" členoma (tipalo, izvršni člen) pa mora signalni sistem čim verneje prenašati signalno vsebino na daljavo. Upirati se mora naravnemu slabljenju prenosnega medija in motnjam različnih oblik iz okolice. Obenem pa seveda ne sme biti energijsko potraten (močnejši signal lahko potuje dlje in je manj dovzeten za vpliv motenj). Ker ni mogoče enoumno določiti ustreznih močnostnih nivojev, so konstruktorji naprav za vodenje v skladu s stanjem tehnologije dokaj poljubno izbirali signalne nivoje, vendarle so le-ti danes več ali manj usklajeni in standardizirani. Pri tipičnem tokovnem signalnem sistemu 4-20 mA se signalne moči gibljejo v območju od

50 do 500 mW, kar je celo manj od moči žarnice v manjših žepnih svetilkah.

7.6.1.1 Signalne oblike in nivoji v sistemih avtomatskega vodenja

V sistemih avtomatskega vodenja uporabljamo tako zvezne kot tudi diskretne signale. Z diskretnimi signali prenašamo informacijo o stanju procesne veličine, ki vsebuje končno število stanj - običajno dve ali tri. Rezervoar je npr. prazen ali poln. Elektromotorno gnani ventil je lahko v skrajno odprtem stanju, skrajno zaprtem stanju ali nekje vmes. Z diskretnimi signali krmilimo aktuatorje med dvema, tremi, … stanji, npr. odpremo ali zapremo ventil.

Z zveznimi signali prenašamo vrednosti posameznih veličin, ki lahko zavzamejo katerokoli vrednost v danem merilnem območju, ali postavljamo aktuatorje v katerokoli točko delovnega obsega. V praksi pogosto v to skupino uvrstimo tudi signale, ki zavzemajo končno število nivojev, npr. 256, 1024, 4096, …

Najpogostejše signalne oblike v sistemih procesnega vodenja so električne, pnevmatske in, v nekaterih primerih, optične. Mehanske in hidravlične signalne povezave so še vedno prisotne v letalski tehniki.

Tabela 7.13 prikazuje nekatere najpogostejše signalne oblike v kemijski in procesni industriji, predstavljene prednosti in slabosti pa nakazujejo, v katerih primerih je katera izmed oblik primernejša.

Pnevmatski signalni sistem je več desetletij služil kot osnova razvoja in aplikacij merilnih in regulacijskih sistemov. Že pogled na prednosti, predstavljene v Tabeli 7.13, kaže na to, da ima veliko zelo koristnih lastnosti, med njimi najpomembnejšo protieksplozijsko varnost in korozijsko neobčutljivost. In ravno v teh primerih danes še vedno uporabljamo ta signalni sistem, ki ga je sicer električni signalni sistem v večini primerov izpodrinil iz sodobnih aplikacij. Med najznačilnejše sodobne primere uporabe sodijo pnevmatsko in elektropnevmatsko krmiljeni zvezni ventil ter pnevmatska in elektropnevmatska sekvenčna (logična) krmilja strojev v lesni in podobnih industrijah, pri pakirnih in transportnih linijah, itd.

Optične signalne sisteme uporabljamo predvsem v primerih, ko se želimo izogniti problemom in poškodbam zaradi posrednega ali neposrednega udara strele (npr. avtomatizacija čistilnih naprav, elektroenergetskih sistemov, itd.). (Opozoriti je potrebno, da tu ne obravnavamo optičnih telekomunikacijskih sistemov, pri katerih optične komunikacije agresivno zamenjujejo ostale komunikacijske načine.)

Z optičnimi signalnimi sistemi v sistemih vodenja se uspešno zoperstavljamo vplivom močnih elektromagnetnih polj in premoščamo signalne poti v eksplozijsko nevarni atmosferi (npr. v rudnikih, pri proizvodnji in distribuciji gorljivih snovi, itd.). Vedeti pa moramo, da se praktično vsak optični signalni sistem začne in konča z električnim signalnim sistemom, saj večina sodobnih merilnih sistemov, regulacijskih in računalniških

sistemov temelji na elektronski tehnologiji. Ker so pretvorbe iz enega v drugi sistem zahtevne in drage, optični sistem v regulacijskih sistemih uporabimo izključno tam, kjer za to nastane resnična potreba (zaščita proti streli, izjemne elektromagnetne motnje, eksplozijska atmosfera).

Tabela 7.13. Pregled nekaterih lastnosti električnih, pnevmatskih in optičnih signalnih sistemov v sistemih procesnega vodenja.

	ELEKTRIČNI SIGNALI		PNEVMATSKI SIGNALI	OPTIČNI SIGNALI
	Tokovni	Napetostni		
Tipične diskretne vrednosti		0/24 V_{AC} 0/24 V_{DC} 0/220 V_{AC} TTL	0/1 bar[9] 0/2 bara, …	svetlobni tok DA/NE
Tipične zvezne vrednosti	0 - 20 mA 4 - 20 mA …	0 - 1 V 0 - 5 V 0 - 10 V -10 - +10 V …	0.2 - 1 bar	frekvenčno impulzno moduliran svetlobni tok 0 - 10 kHz, 0-100 kHz, …
Prednosti	• možno sočasno daljinsko napajanje merilnih pretvornikov • visoka natančnost prenosov (tip. 0.1% polnega obsega) • možne zelo kompleksne obdelave signalov in računalniška obdelava		• protieksplozijsko varen sistem • korozijsko neobčutljiv (cevi iz termoplastov) • zelo enostavno spajanje instalacije • signal lahko nosi tudi aktivacijsko energijo • neobčutljiv na elektromagnetne motnje • enostavno hranjenje pomožne energije v rezervoarjih • neobčutljiv na atmosferska praznjenja (strela)	• protieksplozijsko varen sistem • korozijsko neobčutljiv (vodniki iz stekla ali termoplasta) • neobčutljiv na atmosferska praznjenja
Slabosti	• vpliv zunanjih elektromagnetnih motenj • možne obsežne poškodbe pri direktnem ali indirektnem udaru strele • možen vžig eksplozijskih snovi z električno iskro • korozijsko ogroženi instalacijski elementi		• težja pretvorba fizikalnih veličin v tlačni signal • zmrzovanje kondenzata v signalnih ceveh • potrebna dobra tesnost • potrebna visoka čistost napajalnega zraka • omejene možnosti	• težja pretvorba fizikalnih veličin v svetlobni signal (običajno le s posredovanjem električnih signalov) • potrebna modulacija in demodulacija • zahtevnejši postopki instalacije

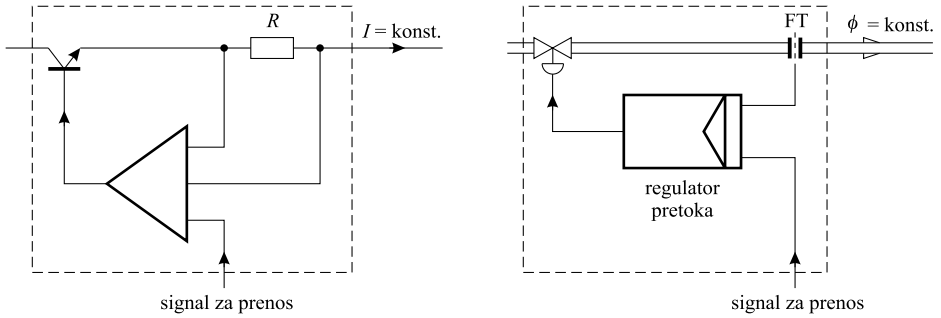
[9] V procesni industriji je uporaba enote za tlak "bar" prevladujoča v primerih podajanja normalno merjenih tlakov. Uporaba osnovne enote "Pascal" je pogostejša pri podajanju nizkih tlakov, merjenih predvsem absolutno. Enota "bar" ne spada v merski sistem SI, vendar je izjemoma dovoljena. 1 bar = 100.000 Pa.

		signalne obdelave • draga finomehanska tehnologija	• potrebne pretvorbe v električni signal in obratno

7.6.1.2 Tokovni in napetostni signali

Kadar želimo merilno-regulacijske signale prenašati z veliko natančnostjo, izvesti tudi zahtevnejše signalne in informacijske obdelave ter hkrati izkoristiti cenenost elektronske tehnologije, pri tem pa problemi z eksplozijsko varnostjo ali zaradi udara strele niso izraziti, uporabimo *električne signalne sisteme*. Pri diskretnih signalih skorajda izključno uporabljamo napetostno obliko prenosov, pri zveznih pa lahko izbiramo med napetostno in tokovno obliko. Napetostna oblika signalov in signalnih poti je običajno cenejša, enostavnejša, a primerna za premagovanje krajših razdalj (do nekaj 10 m). S tokovno obliko dosežemo boljšo natančnost pri daljših prenosih do nekaj km, zato tokovna oblika prevladuje v regulacijskih sistemih procesne in kemijske industrije.

V vsakdanjem življenju imamo običajno opravka izključno z napetostnimi viri in signali. Elektroenergetsko omrežje predstavlja napetostni vir izmenične napetosti. Avtomobilski akumulator je napetostni vir enosmerne napetosti, enako baterija v žepni svetilki. Audio in video signali med napravami zabavne elektronike so tudi napetostni signali. Težko pa bi našli vsakdanji primer tokovnega signala.



Sl. 7.184. Generator konstantnega toka (tokovni vir) in ustrezna hidravlična analogija. Elektronsko vezje (regulator) primerja padec napetosti na uporu R in ga primerja z napetostjo signala za prenos. Pri tem uravnava odprtost tranzistorja tako, da ostane izhodni tok konstanten, ne glede na spreminjajoče se razmere v celotni tokovni zanki. Izhodni tok je odvisen samo od napetosti signala za prenos.

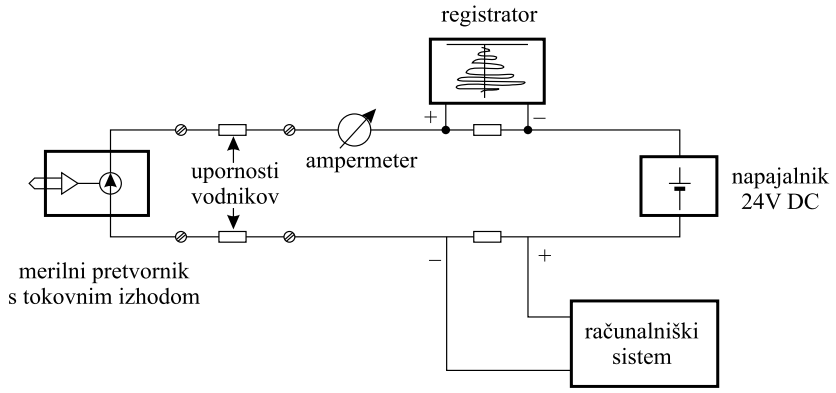
Tokovna oblika signalov v sistemih procesnega vodenja je "umetna" oblika, ki jo je mogoče doseči le z ustreznimi elektronskimi vezji. Gre za to, da v tokokrogu vsiljujemo električni tok, katerega jakost je proporcionalna signalnemu nivoju. Pri tem padci napetosti v tokokrogu takorekoč niso pomembni.

∇

Primer 7.24: *Tokovna zanka*

Primer tokovne zanke prikazuje Sl. 7.185. Kot energetski vir signalnega prenosa služi napajalnik z enosmerno napetostjo (npr. 24V), ki pa je lahko skupen za več signalnih zank.

Tok v zanki uravnava merilni pretvornik s tokovnim izhodom v smislu, kot ga prikazuje Sl. 7.184. V zanko je vključenih več signalnih porabnikov, npr. ampermeter za prikaz trenutnih vrednosti, registrator, računalniški sistem z analognim vhodom, itd. Signalni porabniki so v zanko vezani zaporedno, v nasprotju npr. z električnim omrežjem, kjer porabnike priklapljamo paralelno. Dodatni signalni pretvornik lahko v zanko zopet povežemo serijsko. Vsak signalni odjemnik v zanki sicer poveča padec napetosti, ki pa jo z odpiranjem serijskega tranzistorja merilni pretvornik nadomesti. Če merilni pretvornik priključimo na zelo oddaljenem mestu, se upornosti vodnikov zelo povečajo. Merilni pretvornik tedaj bolj odpre serijski tranzistor in s tem nadomesti večjo napetostno izgubo - tok v zanki pri tem ostane nespremenjen (in seveda proporcionalen prenašanemu signalnemu nivoju). Toda to nadomeščanje povečevanja upornosti ima svoje meje, na kar marsikdo pri razširjanju sistema (dodajanju signalnih odjemnikov) pozabi: vsota vseh padcev napetosti v tokovni zanki ne more preseči napajalne napetosti!



Sl. 7.185. Primer tipične tokovne zanke. Slika prikazuje dvožično povezavo, pri kateri sta merilni pretvornik in napajalnik ločena. Če je napajalnik pridružen pretvorniku, je potrebno do pretvornika napeljati še najmanj dve žici za priključitev na električno omrežje. V tem primeru ponavadi govorimo o štirižični priključitvi.

Δ

Prednost tokovnega načina prenašanja signalov se izkaže prav pri premagovanju velikih razdalj, saj v zanko vsiljeni tok ni odvisen od upornosti vodnikov (za razliko: pri elektrodistribucijskem omrežju od transformatorja bolj oddaljeni porabniki prejemajo nižjo napajalno napetost!). Pogoj za natančni tokovni prenos pa je izvrstna kvaliteta izolacije ožičenja celotne tokovne zanke. Če izolacija kjerkoli "pušča", tok, ki ga uravnava merilni pretvornik, ne doseže vseh odjemnikov v celoti (podobno, kot bi puščalo vodovodno omrežje, ….).

Druga pomembna prednost tokovnega signalnega sistema pa se pokaže pri sistemu z "živo ničlo". V tem sistemu merilni pretvornik del signalnega toka "ukrade" za lastno delovanje. V standardnem sistemu 4-20 mA so za lastno rabo namenjeni točno 4,000 mA, preostanek

toka v obsegu 0-16 mA pa je namenjen osnovnemu signalnemu prenosu. Porabniki signala signal "žive ničle" odštejejo (4,000 mA) in tako sprejmejo koristni del informacije. Tok 4,000 mA, ki je sicer zelo majhen, pri današnji elektronski tehnologiji zadošča za napajanje zahtevnih merilnih pretvornikov. Ti lahko vsebujejo tudi mikrokrmilnike, pomnilniške elemente, prikazovalnik iz tekočih kristalov, analogno-digitalne in digitalno-analogne pretvornike, če le njihova skupna poraba ne preseže predpisanega toka.

Z uporabo sistema z "živo ničlo" prihranimo veliko dela in stroškov pri izvedbi ožičenja sistemov za vodenje, saj za priključitev merilnih pretvornikov potrebujemo le en par žic, le po en par sponk na vsakem odjemnem mestu, itd. V tem smislu je to najracionalnejši znani signalni sistem v sistemih avtomatskega vodenja. Žal pa lahko daljinsko napajamo le nekatere merilne pretvornike, npr. za temperaturo (Pt-100, termočleni), tlak, diferencialni tlak, pretok z vrtinčnim principom (Vortex) ali z merilno zaslonko, ultrazvočni in kapacitivni pretvornik nivoja, pretvornike za pH, redox, težo in nekatere druge. Nekateri merilni pretvorniki namreč za svoje delovanje potrebujejo več energije, kot jo dovaja signal "žive ničle", npr. elektromagnetni indukcijski merilnik pretoka, pretvorniki koncentracij kisika, CO, CO_2, SO_2, NO_x in drugi. Take merilne pretvornike moramo napajati preko dodatnih napajalnih vodnikov.

7.6.1.3 Motnje v signalnih sistemih in zaščita pred njimi

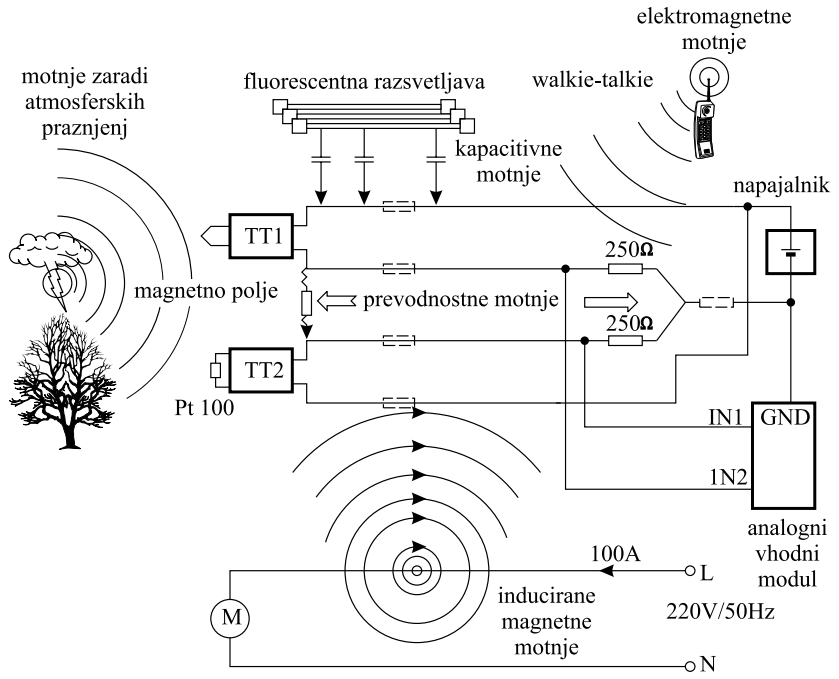
Pri prenosu signalov od merilnih pretvornikov do prikazovalnega, regulatorskega ali računalniškega sistema, ter naprej do aktuatorjev želimo, da bi vsi elementi posameznih signalnih zank prejeli enako informacijo. Če izhod merilnega pretvornika pošilja v zanko tok 17,60 mA kot ekvivalent izmerjenega tlaka 0,850 bar-ov, potem želimo, da ta tok pripotuje do vseh odjemnikov v enaki vrednosti. Toda na svoji poti je signal izpostavljen motilnim signalom, ki ga nepredvidljivo spreminjajo - motijo (motenj, ki v sistem prihajajo že na strani tipal zaradi samih merilnih postopkov, tu ne bomo obravnavali).

Če je električni signalni sistem izpostavljen močno spreminjajočemu se električnemu polju, potem motnje vanj vstopajo *kapacitivno*. (Kapacitivnost v električnem smislu je lastnost dveh nedotikajočih se prevodnih teles v neprevodnem prostoru, da med seboj lahko v obliki električnega polja "ujameta" električni naboj). Če je električni signalni sistem v močnem spreminjajočem se magnetnem polju, potem motnje vanj vstopajo *induktivno*. (Vsak električni vodnik se pri pretoku električnega toka "obda" z magnetnim poljem. Če se to polje spreminja, npr. zaradi spreminjanja toka v vodniku, potem v njemu sosednih vodnikih vzbudi - inducira - tokove, ki želijo te spremembe magnetnega polja kompenzirati). Ponavadi električno in magnetno polje nihata v sožitju - energija enega polja prehaja v drugo in obratno. Prostor, v katerem se to dogaja, vsebuje elektromagnetne valove, ki na druge neodvisne signalne sisteme v njem vplivajo motilno. O elektromagnetnem motenju navadno govorimo tedaj, ko je motilni vir oddaljen od "žrtve" več kot 1/6 valovne dolžine motilnega vira.

Ločimo pa tudi prevodnostne motnje. Te nastajajo tedaj, ko med signalnimi in/ali energetskimi zankami obstaja električno prevodna pot, po kateri signali nenadzorovano

prehajajo iz enega sistema v drugega. Lahko so posledica slabe izolacije med vodniki, vlažnih nečistoč med kontakti sponk ali zaradi skupnih signalnih poti - npr. zaradi skupnega povratnega vodnika.

Ogroženost signalnih sistemov z različnimi oblikami motenj v procesnem okolju je prikazana na Sl. 7.186.



Sl. 7.186. Ogroženost signalnih sistemov v motilnem okolju. Prevodnostne motnje nastopajo zaradi slabe izolacije ali zaradi skupnih (običajno negativnih) vodnikov. Kapacitivne motnje v tem primeru izvirajo od fluorescentne razsvetljave (lahko tudi od drugih vodnikov pod napetostjo). Inducirane magnetne motnje izvirajo od vodnikov, po katerih tečejo veliki tokovi (zagonski tokovi elektromotorjev lahko dosežejo nekajkratno nazivno vrednost - motnje ob tem tudi!). Magnetni viri motenj je tudi bližnji udar strele, katere pot je obdana tudi z močnim magnetnim poljem. Prenosne komunikacijske naprave (walkie-talkie) in bližnji močni oddajniki, radarji, ... so tipični viri elektromagnetnih motenj.

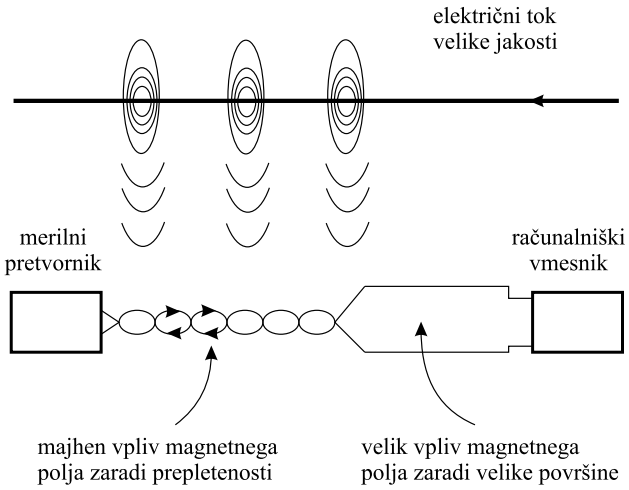
Probleme, ki nastajajo zaradi vdora motenj v signalne sisteme, projektanti, monterji in uporabniki zelo pogosto uvrščajo med "mistične" probleme. Zelo neprijetni pa so, ker se običajno pokažejo šele tik pred zaključkom projekta - v fazi preizkušanja. Skrivnostnost težav nastopi zaradi pogostega sočasnega delovanja več motilnih virov, zaradi navidezne naključnosti, zaradi nejasnih motilnih poti ali zaradi nepoznavanja te problematike. Vse dogajanje pa je mogoče opisati z znanimi zakoni elektromagnetizma oz. fizike.

Glavni problem je določiti, kateri so viri motilnih signalov ter kje in kakšne so poti do "žrtev". Tega pa ni moč storiti le z meritvami s pogosto dragimi merilnimi instrumenti, pač pa je pri tem potrebno imeti tudi precejšnjo mero iznajdljivosti in srečne roke pri ugibanju, kje bi utegnile biti glavne motilne poti.

Pri izvedbi projektov se seveda ne moremo zanašati na "srečno roko". Signalne sisteme zato gradimo in povezujemo po ustaljenih pravilih ter na osnovi lastnih izkušenj in se na ta način poskušamo izogniti pravim težavam.

Z namenom zmanjšanja vpliva kapacitivnih motilnih signalov vse občutljivejše signalne povezave izvedemo z oklopljenimi kabli. Oklop okrog vodnikov namreč prestreže spreminjajoče se električno polje in njegovo posledico - električni tok - preusmeri v ustrezno točko. Oklop praviloma povežemo na skupno točko vira signala, čeprav v sistemih procesnega vodenja pogosto oklope iz drugih razlogov povežemo v skupno točko na strani računalniških vmesnikov.

Oklop signalnih vodnikov ne ščiti signalov pred magnetnimi motnjami in tudi ne pred magnetnim poljem bližnjega udara električne strele! Magnetnim induciranim motnjam se upremo, če vodnike posameznih signalnih zank med seboj prepletemo (komercialni oklopljeni kabli imajo v ta namen prepletene pare vodnikov), s prostorsko namestitvijo ožičenja pa poskušamo zmanjšati površino, ki jo signalna vodnika ene zanke oklepata. (glej Sl. 7.187)



Sl. 7.187. Pri prepletenih vodnikih se prispevki sosednjih zankic kompenzirajo. Pri veliki površini, ki jo signalna vodnika oklepata, je vpliv magnetnega polja sorazmeren zajeti površini.

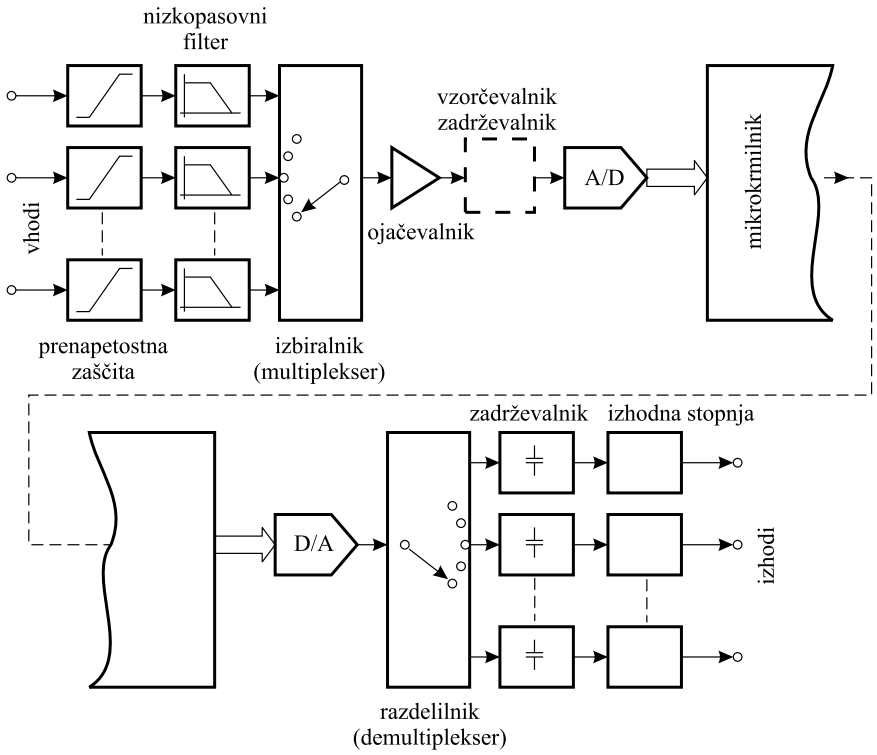
Kombinacija oklopa in prepletenih vodnikov ščiti signale tudi proti elektromagnetnim visokofrekvenčnim motnjam. Za kvalitetno obratovanje regulacijskih sistemov pa je pogosto uporaba prenosnih radijskih postaj, ki lahko povzročijo visoko elektromagnetno

polje, prepovedana.

Še nekaj o ozemljevanju. Zmotno je prepričanje, da je ozemljevanje (predvsem oklopov) postopek za znižanje nivoja motenj - obratno - pogosto se zgodi, da povezava oklopov z ozemljitvijo celo poslabša kvaliteto signalov. Postopek ozemljevanja je varnostni postopek, predpisan z zakonom, namenjen pa je preprečevanju nastanka smrtno-nevarne napetosti dotika.

7.6.1.4 Pretvorba zveznih signalov v digitalno obliko in obratno

Pri skorajda vseh sodobnih gradnikih računalniške avtomatizacije poteka jedro signalne in informacijske obdelave v digitalni obliki s pomočjo vgrajenih mikroprocesorjev ali mikrokrmilnikov. Vse zvezne vhodne signale je zato potrebno najprej pretvoriti v digitalno obliko, po informacijski obdelavi pa nazaj v analogno (zvezno) obliko. V ta namen mora vsak gradnik vsebovati elemente, ki so prikazani na Sl. 7.188, le redke je mogoče v določenih primerih opustiti.



Sl. 7.188. Veriga priprave analognih signalov za digitalno obdelavo.

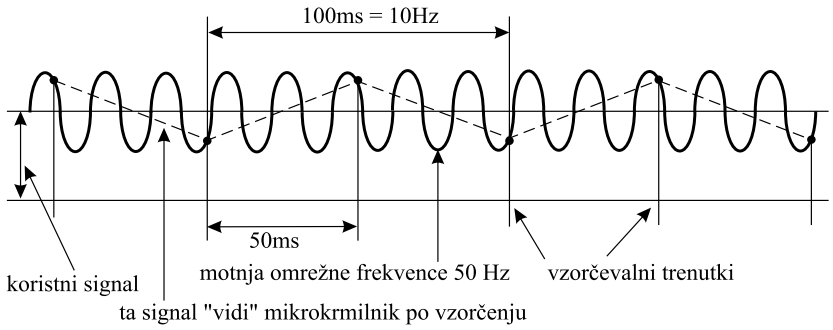
Vhodne signale peljemo najprej preko pre-napetostne ali pre-tokovne zaščite, ki deloma ščitijo preostali del sistema pred poškodbami zaradi kratkotrajnih kratkih stikov, napačne vezave, razelektritve statičnega naboja teles, induciranih napetosti zaradi bližnjega udara

strele, itd. Ceneni gradniki te zaščite pogosto nimajo.

Sledi *nizkopasovni filter*, ki omeji frekvenčni spekter vhodnega signala tako, da je primeren za vzorčenje. Pri vzorčenju moramo biti namreč zelo pozorni: če vzorčeni signal vsebuje komponente, ki se spreminjajo hitreje od frekvence vzorčenja, bodo te komponente preslikane v nižjo frekvenco, kjer jih ni več mogoče ločiti od koristnega dela signala.

Primer na Sl. 7.189 kaže to "nevarno" preslikavo, kjer se motnja omrežne frekvence preslika v signal frekvence 10 Hz, ki pa bi bil že lahko v frekvenčnem spektru koristnega signala marsikaterega merilnega pretvornika.

Z nizkopasovnimi vhodnimi filtri znižamo amplitude vseh tistih komponent vhodnega signala za toliko, da po vzorčenju ne morejo povzročiti zaznavne merilne napake. Znižamo pa predvsem tiste komponente vhodnih signalov, ki jih ni bilo mogoče odpraviti z oklapljanjem in prepletanjem signalnih vodov ter dušitvijo motenj pri povzročiteljih (kar moramo najprej poizkusiti!). Prav tako znižamo prispevek šumnih signalov, ki so posledica merilnih postopkov, npr. turbulenc medija pri merjenju pretokov, valovanja tekočin v reaktorjih pri merjenju nivoja s hidrostatično metodo, itd.



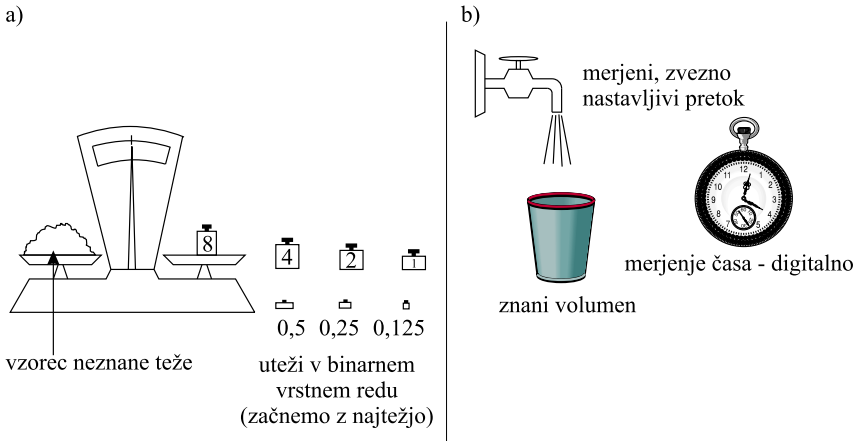
Sl. 7.189. Primer preslikave motilnega signala mrežne frekvence v navidezni signal frekvence 10 Hz. Vzorčevalna frekvenca znaša 20 Hz (50 ms).

Mejne frekvence vhodnih filtrov običajno niso nastavljive. Izbral jih je načrtovalec aparature in za tipične industrijske regulacijske sisteme običajno ležijo v območju od 0,5 Hz do 10 Hz. Tu velja opozoriti, da z uporabo digitalnih filtrov, ki pa jih uporabnik ponavadi lahko uglašuje pri večini gradnikov vodenja, ne odpravimo nevarnosti, ki jo kaže slika Sl. 7.189. Po vzorčenju je prepozno!

Izbiralnik služi zato, da lahko obdelamo več vhodnih signalov le z enim analogno-digitalnim (A/D) pretvornikom, ki je ponavadi najdražji element verige. V vsakem vzorčevalnem ciklu mikrokrmilnik ciklično izbere vse vhodne signale (enega po enega) in vsakega poveže z A/D pretvornikom. Med izbiralnikom in A/D pretvornikom včasih potrebujemo še ojačevalno stopnjo, s katero dvignemo nivo signala, ter vzorčevalno-zadrževalno vezje, ki zadrži vrednost merjenega signala med časom, ki ga A/D pretvornik

potrebuje za pretvorbo. Pri večini naprav za regulacije ta člen ni potreben (v nasprotju npr. z digitalno obdelavo avdio signalov, kjer je nujen!).

A/D pretvornik pretvori zvezni signal v digitalno, najpogosteje v binarno obliko. Najpogostejši sta dve pretvorniški metodi: metoda z zaporednim približevanjem in integracijska metoda. Prva je zelo hitra, druga počasna a bistveno manj občutljiva na preostale motilne signale. Metoda z zaporednim približevanjem deluje na osnovi primerjave amplitude vhodnega signala z utežmi, npr. v naslednjem binarnem vrstnem redu: 5V, 2.5V, 1.25V, 0.625V, 0.312V, itd. Običajno uporablja od 8 do 14 uteži (govorimo o 8 do 14 bitni pretvorbi). V uporabi je največkrat pri hitrih napravah ali napravah z veliko vhodnih signalov. Nasprotno pa pri integracijski metodi pretvornik najprej predpisani čas integrira napetost vhodnega signala, nato pa meri čas, v katerem ta integrirani naboj pri znanem praznilnem toku izprazni. Vrednost vhodnega signala je tedaj proporcionalna času praznjenja, ki pa ga je mogoče meriti s štetjem impulzov - od tu naprej že dobimo digitalno obliko, primerno za obdelavo z mikrokrmilniki. Podobnost obeh pretvorniških metod z vsakdanjimi postopki prikazuje Sl. 7.190.



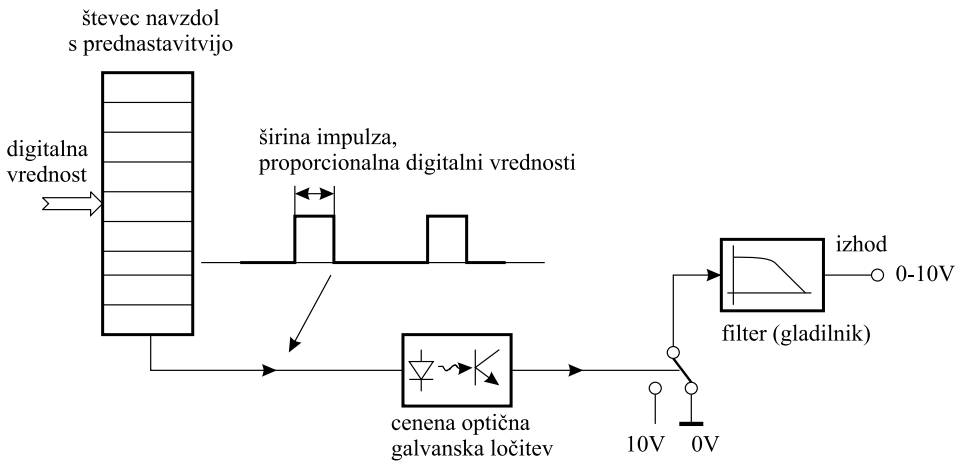
Sl. 7.190. Podobnost A/D pretvorniških metod z vsakdanjimi postopki:
a) tehtanje z zaporednim približevanjem
b) "integracija" pretoka ob merjenju potrebnega časa (tudi tu izmerimo povprečni pretok)

Prednost integracijskega pretvorniškega postopka je prav v integraciji. Integriranje (elektronsko) ima izrazit nizkopasovni filtrski značaj in pravzaprav dá kot rezultat povprečno vrednost signala v določenem času, za razliko od metode z zaporednim približevanjem, pri kateri poteka pretvorba trenutne vrednosti signala. Morebitne neodstranjene motnje integracijska pretvorniška metoda še dodatno oslabi. Žal pa je integracijska metoda počasna in ni primerna za hitra vzorčenja na veliko vhodnih kanalih.

Pot izhodnega signala (od mikrokrmilnika do signalnega sistema) je običajno enostavnejša in vsebuje manj nevarnosti. Zato poznamo tudi več primerljivih izvedb, ena izmed njih je

prikazana v drugem delu Sl. 7.188. Velikost signala, ki naj bi jo mikrokrmilnik poslal proti aktuatorjem, je v njegovi aritmetično-logični enoti prisotna le kratek čas - shraniti jo mora v pomnilnik za lastno rabo in nato nadaljevati z obdelavo drugih signalov. Ko jo odda navzven, jo moramo na nek način zadržati do naslednjega obdelovalnega ciklusa. To zadržanje vrednosti lahko opravimo v digitalni obliki še pred digitalno-analogno (D/A) pretvorbo (v ta namen pred D/A pretvornik namestimo pomnilniški register), lahko pa po D/A pretvorbi že v analogni obliki v zadrževalnikih, ki vsebujejo kondenzatorje, kot je to prikazano na Sl. 7.188. Če vrednosti signalov zadržimo že v digitalni obliki, potem potrebujemo za vsak izhodni kanal svoj D/A pretvornik. Če pa sistem vsebuje le en D/A pretvornik, lahko njegove izhodne vrednosti razdelimo z razdelilnikom med več izhodnih kanalov in jih začasno zadržimo s kondenzatorji. V prvem primeru je čas zadrževanja neomejen, v drugem pa je treba signale dovolj pogosto osveževati, saj se kondenzatorji neprestano praznijo.

Postopkov D/A pretvorbe je tudi zelo veliko. Načrtovalci naprav poskušajo uporabiti najcenejšo varianto, ki še zadošča kriterijem in ima dovolj visoko izhodno razločljivost. Ta se giblje med 8 in 12 biti, kar ustreza 256 oziroma 4096 nivojem izhodnega signala. Morda najpogostejšo rešitev prikazuje Sl. 7.191, odlikuje pa se v zelo enostavni možnosti vgradnje galvanske (optične) izolacije med digitalno in analogno stranjo pretvornika.



Sl. 7.191. Cenena in učinkovita D/A pretvorba: z digitalno vrednostjo prednastavimo števec - navzdol. Njegov izhodni signal je impulz z nastavljivo širino. Če v njegovem ritmu preklapljamo med 0 V in referenčno vrednostjo (10 V), dobimo na izhodu gladilnega filtra srednjo vrednost → zvezni analogni signal.

Izhodna končna stopnja v verigi priprave analognih signalov pretvori interni izhodni signal v obliko, ki je primerna za prenos. V sistemih procesnega vodenja in za večje razdalje je to tokovni signal 0-20 mA ali 4-20 mA, za krajše razdalje pa različni napetostni signali, npr. 0-10V, -10 do +10V, 0-5V, 1-5V, itd. Med konstrukcijske posebnosti končnih stopenj sodi predvsem vzdržljivost. Biti mora namreč konstruirana tako, da prenese kratek stik,

priključitev zunanjih virov napetosti in tokov ter podobne nepravilnosti in napake, ki se pojavljajo pri prvem zagonu, med obratovanjem in drugih okvarah, zaradi nepazljivosti monterjev in vzdrževalcev, itd.

V sistemih avtomatskega vodenja ima signalni sistem zelo pomembno vlogo. Med seboj povezuje vse elemente vodenja, kot so tipala z merilnimi pretvorniki, regulatorji, prikazovalni in registratorski instrumenti, računalniški sistemi, programirljivi logični krmilniki, in na koncu verige sistema, aktuatorji z izvršnimi členi. Ker so fizikalne in nivojske lastnosti signalnih elementov, ki so povezani na procesno opremo, zelo različne, je standardni signalni sistem več kot nujen. Najbolj razširjeni so električni signalni sistemi, za posebne primere pa tudi pnevmatski in optični. V električnih signalnih sistemih uporabljamo stopenjske signale, zvezne tokovne, napetostne in druge specialne signale, ter v zadnjem času vse bolj tudi komunikacijske (digitalne) signale. Zvezni signali pokrivajo večino spodnjega nivoja signalnih povezav merilnih pretvornikov in aktuatorjev, čeprav jih tudi tu že izpodrivajo digitalni komunikacijski sistemi. Tokovni signali z živo ničlo so izjemno racionalna in zrela tehnična rešitev in zagotavljajo relativno visoko točnost prenosov vrednosti na daljavo.

Za kakršnokoli zahtevnejšo obdelavo signalov danes brez izjeme uporabljamo računalniške, mikroračunalniške in mikrokrmilniške sisteme. Zato moramo signal vsakega tipala po prenosu na daljavo pretvoriti v digitalno obliko, ga nato računsko in algoritmično obdelati, nato pa zopet pretvoriti v ustrezno analogno obliko, ki služi za krmiljenje procesnih aktuatorjev.

Na celotni poti digitalnega procesiranja signalov je prenos in pretvorba signalov tipal v digitalno obliko najzahtevnejši postopek, ki pa, če je pravilno izveden, bogato poplača trud in zagotavlja visoko kvaliteto avtomatske regulacije. Signali morajo potovati skozi prostore s pogosto visokimi nivoji motenj različnih oblik, njihovem vplivu pa se moramo zoperstaviti: najprej pri izvorih, kjer je mogoče z vgradnjo dušilnih elementov nivo motenj bistveno znižati, nato prostorsko čimbolj razdvojiti energetske in signalne vode, nato pravilno uporabiti oklopljeno ožičenje s prepletenimi paricami in nazadnje preostanek motenj v prvem koraku znižati z uporabo analognih filtrov, v drugem koraku pa s kompromisno nastavitvijo digitalnih filtrov razdvojiti koristni del signala od škodljivega.

7.6.2 Digitalne komunikacije v sistemih procesnega vodenja

O digitalnih komunikacijah povsem naravno razmišljamo vsakič, ko želimo med seboj povezati dva ali več digitalnih računalnikov, ali računalnik s tiskalnikom, ali računalnik s programirljivim logičnim krmilnikom. V teh primerih je izvor podatkov (signalov) digitalen, prav tako ponor, zato je digitalna povezava seveda logična. V procesnem regulacijskem okolju srečamo izvore in ponore signalov tako v zvezni, kot tudi v diskretni obliki. Signale v diskretni obliki (npr. signale končnih stikal, tlačnih stikal, pretokovnih stikal, ter signale za vklop črpalk, motorja, dvopoložajnih ventilov) ni težko

pripraviti za digitalni komunikacijski prenos. Signale v zvezni (analogni) obliki je potrebno prej pretvoriti v digitalno obliko z analogno-digitalnimi (A/D) pretvorniki in v obratni smeri z D/A pretvorniki. V tem primeru postane smiselno vprašanje, kaj pridobimo z uporabo digitalne komunikacije, saj so pretvorbe iz analogne v digitalno obliko in obratno dokaj zahtevne in tudi drage.

Nagibov za uporabo digitalnih komunikacijskih povezav tudi na tem spodnjem nivoju industrijske avtomatizacije je več, veliko pa je tudi argumentov, ki takemu načinu gradnje sistemov niso naklonjeni. Zaradi tega mora projektant in/ali uporabnik takega sistema pred izvedbo zelo dobro razmisliti in pretehtati, ali naj gradnike vodenja med seboj poveže s klasičnimi analognimi in digitalnimi signalnimi povezavami ali z digitalnim komunikacijskim omrežjem.

7.6.2.1 Osnove digitalnih komunikacij

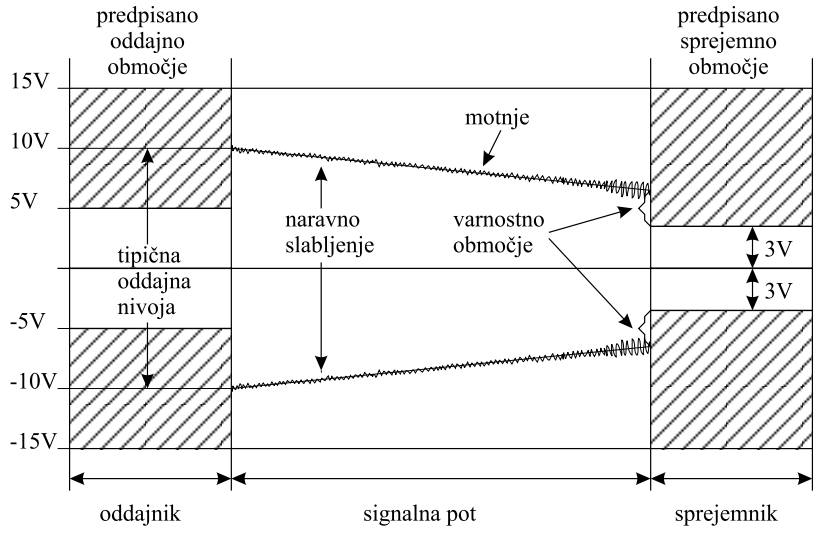
Pri klasični signalni zvezi povežemo signalni izvor s signalnim ponorom preko ene izmed različnih signalnih poti. Signali so lahko diskretni ali zvezni, signalne poti pa električne, optične, pnevmatske, … V sistemih avtomatskega vodenja vsakemu paru izvor-ponor običajno pripada svoja signalna pot, v drugih sistemih (npr. telefonija, telegrafija) pa je mogoče signalno pot časovno dodeljevati različnim parom izvor-ponor (telefonska centrala).

Signali so na svoji poti izpostavljeni naravnemu slabljenju in motnjam iz okolice. Signalni izvor (oddajnik) ter ponor (sprejemnik) medsebojni signal tudi nekoliko različno interpretirata, kar je odvisno od njune točnosti (npr.: signalni izvor želi oddati tok 17,65 mA, v resnici odda 17,60 mA, sprejemnik pa zaradi točnosti svojih elementov odčita 17.71 mA). S pomočjo uveljavljenega in tehnično zrelega tokovnega signalnega sistema 4-20 mA je mogoče na zmernih razdaljah in v šibkem motilnem okolju doseči natančnosti prenosa v območju od 0.2 do 0.5%, bolje pa le z zelo izbranimi komponentami, z zelo pazljivo instalacijo in natančno kalibracijo.

Pri digitalni komunikaciji na oddajni strani signal pretvorimo v digitalno obliko in ga nato odpošljemo po digitalnem komunikacijskem kanalu. Če digitalni podatek uspe priti do sprejemnika brez izrazite motnje, potem sprejemnik sprejme *absolutno enako vrednost*, kot jo je oddajnik oddal (npr. če je oddajnik oddal vrednost temperature 317.5^{o}C, potem takšno vrednost sprejemnik tudi sprejme). Pri pravilno načrtovanem komunikacijskem signalnem vodu naravno slabljenje ne povzroči nobenega odstopanja. Prav tako nimajo nikakršnega vpliva tudi zmerne motnje. Sl. 7.192 pojasnjuje, od kod imunost v digitalnih sistemih. Če je nivo motenj sprejemljiv in naravno slabljenje (dolžina kablov) v predpisanih mejah, potem oddani signalni nivo (visok ali nizek) sprejemnik pravilno interpretira. Do napake pride le pri izjemnih motnjah izven normalnih območij.

Naslednja bistvena različnost med zveznimi signalnimi sistemi in digitalnimi komunikacijskimi sistemi je v prisotnosti signalov. V zveznem sistemu je signal npr.

merilnega pretvornika tudi na koncu signalne poti stalno prisoten - lahko ga povežemo na kazalčni instrument ali na registrator ali klasični elektronski regulator. Nasprotno pa je v komunikacijskem sistemu signalna vrednost prisotna le v določenih časovnih trenutkih, največkrat periodično, npr. vsako sekundo, vsakih 100 ms, ipd. Takega signala ni več mogoče povezati na klasični registrator ali elektronski regulator, ker mu "vmes nekaj manjka". Potrebna je nadaljnja računalniška obdelava.



Sl. 7.192. Imunost digitalnega komunikacijskega omrežja (primer EIA 232 C/D) proti naravnemu slabljenju in motnjam izvira iz širšega sprejemnega območja kot je oddajno območje. Navkljub motnjam in naravnemu slabljenju oddani visoki nivo sprejemnik pravilno interpretira kot visoki nivo in obratno.

Oblikovanje digitalnih sporočil

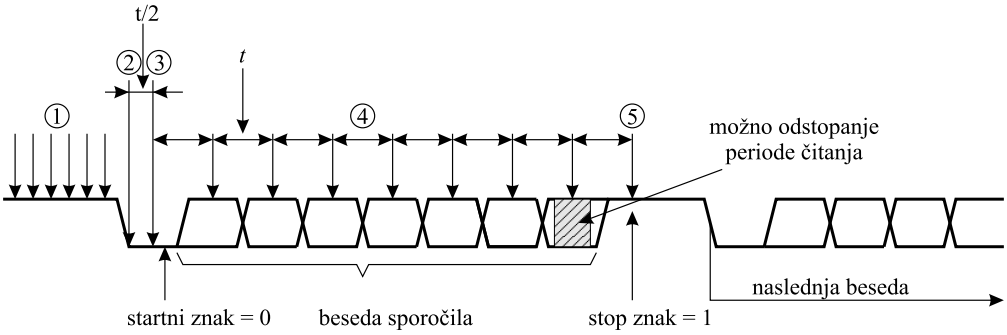
Pred vsakim digitalnim prenosom moramo vse signale pretvoriti v digitalno obliko. Ta ima lahko binarno obliko, še pogosteje pa gre za kodirani zapis. Če npr. merilni pretvornik izmeri temperaturo 317.5^{o}C, potem je mogoče vrednost 317.5 prenašati v binarnem zapisu s plavajočo vejico, npr. (00000010 01111011 10000001). Podatek v tej obliki je primeren za nadaljnjo računalniško obdelavo, ni pa primeren za izpis na zaslon ali tiskalnik. V tem drugem primeru je bolje znake 3 1 7 • 5 kodirati s pomočjo standardne ASCII kode, kjer napisani znaki dobijo naslednje kode: 0110011 0110001 0110111 0101110 0110101. Dolžina digitalne besede je tu seveda bistveno daljša, kodiranje pa ima predvsem to prednost, da poleg številčnih vrednosti omogoča prenašanje tudi raznih krmilnih znakov in besedil.

Ko imamo na voljo direktni ali kodirani binarni zapis, ga moramo pripraviti za odposlanje ("Položiti ga moramo v ovojnico".) V nizu "1" in "0", ki potujejo po komunikacijskem omrežju, moramo določiti, kje sta začetek in konec sporočila ali

podatka. Tu ločimo dva pristopa: besedno urejen in bitno urejen protokol. Pri *besednem načinu* prenašamo besedo za besedo. Vsako binarno besedo, običajno dolžine 7 ali 8 znakov, je potrebno opremiti z začetnim in končnim znakom. Pri *bitno urejenem načinu* sporočila *večjih* dolžin opremimo z začetnim in končnim *sporočilom*, ki poleg označitve začetka in konca sporočata tudi, kako dolgo je sporočilo. Na ta način je prenos daljših sporočil zelo učinkovit, pri kratkih sporočilih pa ne, saj je sporočilni okvir (začetno in končno sporočilo s sinhronizacijskimi znaki) relativno obsežen. Za odčitavanje vrednosti merilnih pretvornikov je običajno učinkovitejše besedno urejeno sporočilo, saj so podatki kratki.

Sinhroni in asinhroni komunikacijski prenos

Naslednja značilnost digitalnih komunikacijskih povezav je sinhronost med sprejemnikom in oddajnikom. Pravilno sestavljeno sporočilo na koncu oddajnik odda prenosnemu mediju v obliki serijskega niza znakov 1/0. Sprejemnik "čuti" pri sprejemu spremembe signalnih nivojev med 1 in 0 - toda kje je tu začetek? Kaj pripada posameznemu znaku? S kakšno hitrostjo je bilo sporočilo oddano? Sprejemnik mora na vsak način nekaj vedeti o oddajniku! Tehnično poznamo dve različici: sinhrono in asinhrono. Asinhroni način prenosa je enostavnejši - tudi izvedbeno. Primeren je predvsem za kratka, besedno orientirana sporočila. Je izjemno razširjen, toda primeren le za nižje hitrosti prenosov. Na Sl. 7.193 je prikazan primer, kako sprejemnik v asinhronem načinu delovanja "tipa" sprejeti signal. Sprejemnik v tem primeru "ve" le za približno frekvenco oddajanja sporočila.



Sl. 7.193. Razpoznavanje sprejetega sporočila v asinhronem načinu prenosa:

1 - iskanje start znaka
2 - detekcija začetka start znaka
3 - preizkus, ali je start znak res prisoten
4 - odčitavanje posameznega znaka v časovnih razmikih, ki so približno enaki razmikom oddaje
5 - detekcija stop znaka (pričakovanega)

V asinhronem načinu delovanja sprejemnik najprej s 4-16 kratno frekvenco (kot je frekvenca prenosa) tipa signal, ki ga sprejema (točka 1 na Sl. 7.193). Takoj, ko zazna

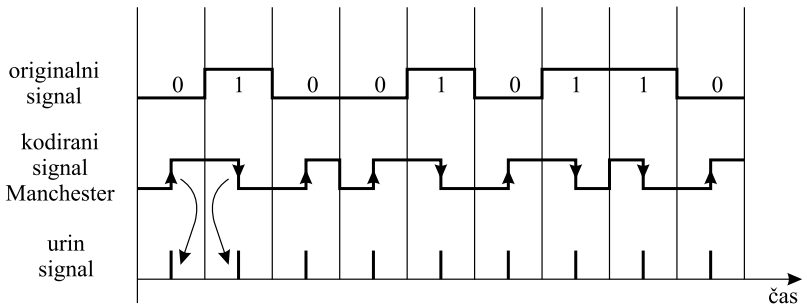
padec - začetek start znaka - (točka 2), prične meriti čas. Ko preteče približno polovico časa, ki je sicer namenjen prenosu enega znaka (npr. 52 μs pri hitrosti prenosa 9600 baud-ov), preveri ali je to res startni znak, ki mora imeti nizek nivo (točka 3). Če je test pozitiven, potem s časovnim razmikom t (t = 104 μs pri 9600 baud-ih) odtipa še preostale vrednosti prihajajočih znakov. Nato v točki 5 sprejemnik pričakuje visoki nivo, ki označuje stop-znak. Če je tudi takrat test v redu, prične sprejemnik znova "tipati", dokler ne zazna startnega znaka nove besede, prejšnjo sprejeto besedo pa preda mikroprocesorju v nadaljnjo obdelavo.

Na osnovi Sl. 7.193 je mogoče razumeti, zakaj ni potrebno, da bi sprejemnik tipal prihajajoči signal točno z isto frekvenco, kot je bil oddan. Frekvenci se lahko razlikujeta tudi za nekaj %, pa vendar poteče sprejem brez napak. Pomembno je le, da se odčitavanje zadnjega znaka izvrši še v tistem času, ko je ta znak še ali že "v veljavi".

Asinhroni način prenosa je zelo enostaven in tudi dokaj zanesljiv. Je pa počasen, saj se na zgoraj opisani način sprejemnik in oddajnik pred vsako besedo sporočila ponovno sinhronizirata. Običajno je mogoče na ta način komunicirati s hitrostmi do 19200 znakov na sekundo (Baud-ov), možno pa je doseči tudi standardne hitrosti 38.4 kBaud, 62.5 kBaud ali celo 375 in 500 kBaud.

Za vse hitre prenose, npr. za prenose množice podatkov med računalniki, pa je potrebno uporabiti sinhroni prenos. Pri tem (npr. Ethernet, ipd.) je treba zagotoviti popolno frekvenčno in fazno skladnost med oddajnikom in sprejemnikom. Tedaj lahko sprejemnik odčitava samo koristne znake sporočil, ki so lahko zelo dolgi, ni pa treba izgubljati časa s ponovno sinhronizacijo pred vsako besedo.

V sinhronem načinu prenosa je urin signal (signal, ki proži oddajanje posameznih znakov) običajno skrit kar v samem oddajnem signalu, in sicer s postopkom dodatnega kodiranja. Sl. 7.194 prikazuje primer dodatnega kodiranja sporočila z uporabo t.i. Manchester kode (možne pa so tudi druge kode).



Sl. 7.194. "Skrivanje" urinega signala v oddani signal z uporabo t.i. Manchester kode. ("0" originalnega podatka se kodira kot prehod navzgor, "1" pa kot prehod navzdol).

Na sprejemni strani sprejemnik iz sprejetega signala najprej posebej izlušči skriti urin

signal, ki ga fazno in frekvenčno vskladi s svojo sprejemno uro. Za to vskladitev potrebuje nekaj časa, zato v komunikacijski zvezi oddajnik med oddajanjem koristnih sporočil oddaja tudi nekaj informacijsko nepomembnih sinhronizacijskih besed, ki omogočajo, da se sprejemniki sinhronizirajo z oddajniki. Po vzpostavitvi sinhronizacije sprejemnik preprosto odčitava sprejeti signal v ritmu svoje ure in ga z obratnim postopkom Manchester kodiranja dekodira.

Najpogosteje besedno urejena sporočila prenašamo z asinhronim načinom prenosa, bitno urejena pa s sinhronim načinom prenosa (to ne pomeni, da preprosta sporočila, ukaze, itd. prenašamo s sinhronim prenosom ali dolga besedila z asinhronim prenosom - obratno - preprosta sporočila, urejene vrednosti, itd., uredimo v 7 ali 8 bitne besede in jih prenašamo z asinhronim prenosom; dolga sporočila ali cele podatkovne baze pretvorimo v dolge nize in jih nato prenašamo sinhrono).

Preproste komunikacijske povezave, kot npr. povezava osebnega računalnika s tiskalnikom, potekajo v asinhronem načinu delovanja. Tudi omrežje tipa PROFIBUS (DIN 19245), namenjeno procesnemu vodenju, deluje v asinhronem načinu. Nasprotno pa so sinhrona računalniška omrežja naslednja: ETHERNET, komunikacijsko omrežje tipa Bitbus, IBM-ov komunikacijski sistem SDLC in drugi.

Prenos sporočil po prenosnih medijih

V dosedanjem opisu smo spoznali, kako je potrebno podatek oblikovati, da je primeren za serijski prenos (v komunikacijski tehniki podatki praviloma potujejo v serijski obliki znak za znakom). Za uspešen prenos sporočil pa potrebujemo še postopek priprave podatkov na eni strani in prenosni medij na drugi strani.

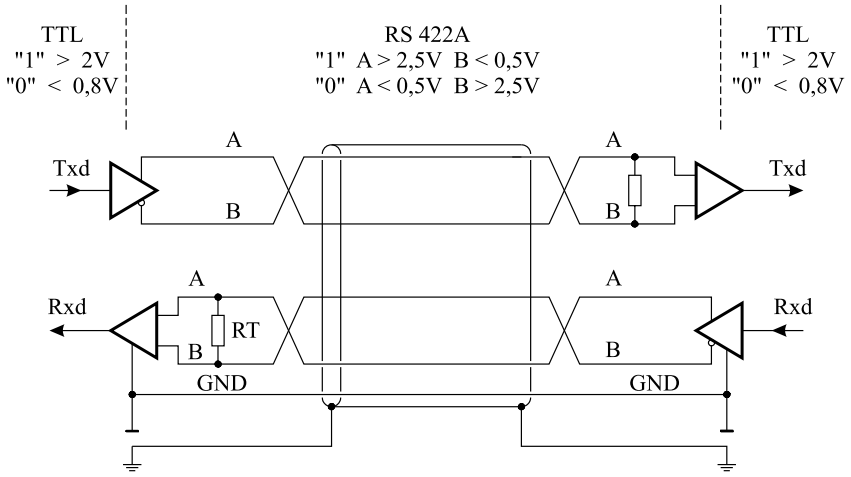
Prenosni mediji so lahko električni-žični, elektromagnetni (radijski) in optični. V okolju procesnega vodenja so najpogostejši električni prenosni sistemi, v posebnih primerih (npr. visok nivo elektromagnetnih motenj ali nevarnost udara strele) pa so še posebej primerni optični sistemi.

Po večletnem razvoju in po neusklajeni konkurenčni tekmi številnih proizvajalcev tovrstne opreme so se izoblikovali nekateri prenosni načini, ki so postali standardni v svetovnem merilu. Najpogostejši in široko razširjen je sistem po standardu EIA/RS 232d, na katerega naletimo pri vsakem osebnem računalniku, tiskalnikih, pa tudi pri nekateri profesionalni opremi za procesno vodenje. Z njim lahko povežemo med seboj le dve oddajno-sprejemni napravi, omogoča pa hitrosti prenosov do 9600 Baudov (1 Baud je en znak na sekundo), izjemoma tudi 19200 Baudov ali celo več. Pri hitrosti 9600 Baud-ov je priporočljiva največja razdalja približno 15 m, kar pa je v procesnem okolju običajno bistveno premalo. Dolžino prenosa je mogoče sorazmerno povečati, če pri tem zmanjšamo hitrosti prenosa. Toda že hitrost 9600 Baud je največkrat premajhna. Če so npr. prenašane besede dolge 11 znakov (1-start, 8-podatek, 1-test parnosti, 1-stop), potem za prenos ene besede potrebujemo 11 * 104.2 μs = 1.15 ms. Za prenos 100 podatkov z ASCII kodo (6 besed/podatek), bi tako potrebovali najmanj 690 ms, realno pa bi prenos zaradi dodatnih omejitev težko opravili prej, kot v 1 s. To pa je običajno za

sisteme v realnem času mejna pogostnost prenosa podatkov merilnih pretvornikov.

Za hitrejše prenose na večjih razdaljah je primernejši ustrezni sistem po standardu EIA RS/422A ali CCITT X27, ki omogoča hitrosti prenosa do 10 M-Baud-ov na razdaljah do 1200 m. Toda tudi ta sistem omogoča povezavo le dveh oddajno-sprejemnih naprav. Primeren je npr. za povezavo inteligentnega procesnega regulatorja ali programirljivega logičnega krmilnika s procesnim računalnikom. Po tem standardu potrebujemo za vsako smer prenosa po eno parico - dvojico žic v oklopljenem kablu (glej Sl. 7.195)

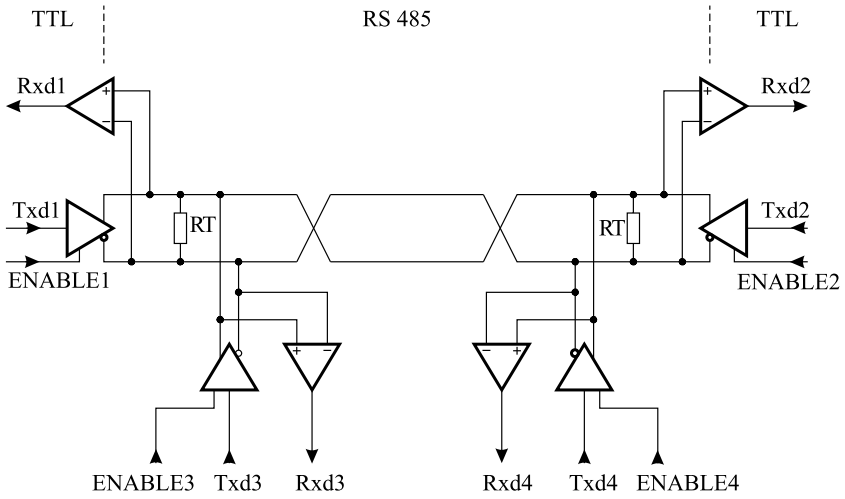
V tem sistemu električni signal potuje po signalni parici simetrično, saj ima vsak oddajnik normalni in komplementarni izhod. Če je normalni izhod na visokem logičnem nivoju, je komplementarni na nizkem in obratno. Če na tako signalno parico "gledamo od daleč", potem pri prenosu električnih signalov navzven ne čutimo nobenih sprememb - v popolnoma simetričnem sistemu ne dobimo zunanjega elektromagnetnega polja (to je v prispodobi tako, kot bi se tehtali, pa bi pri tem prestavljali dodatno utež iz žepa v žep - tehtnica bi vedno kazala isto težo). Ker se signalna parica obda le z zelo šibkim elektromagnetnim poljem, je potrebno za prenos precej manj energije, kot npr. v asimetričnem sistemu po standardu EIA RS232d. Zaradi tega so prenosne razdalje lahko bistveno daljše in prenosne hitrosti bistveno večje. Ker je zunanje elektromagnetno polje zelo šibko oziroma zanemarljivo, ta sistem pri prenosu podatkov zelo malo moti okolico - in obratno - zunanje elektromagnetno motilno polje težko prodre v signalni sistem, zato je prenos zelo zanesljiv. Hude omejitve razdalje in hitrosti pri sistemu po priporočilih prej omenjenega standarda RS-232d so namreč postavljene predvsem zaradi motenj, ki jih ta sistem oddaja v okolico in jih od tam tudi sprejema.



Sl. 7.195. Dvosmerna povezava po priporočilih standardov RS 422A/X27

Odlične lastnosti signalnega medija po priporočilih standarda RS 422A so vodile do razvoja sistema, ki ga priporočata standarda EIS/RS485 oziroma CCITT V11. Osnova komunikacijskih povezav po tem standardu je prikazana na Sl. 7.196. Na tej osnovi

temeljijo številni komunikacijski sistemi za povezavo gradnikov procesnega vodenja, med drugimi tudi sistem PROFIBUS DIN 19245, ki počasi postaja resnični standard za sisteme procesnega vodenja.



Sl. 7.196. Več komunikacijskih naprav na skupnem vodilu po priporočilih standardov RS485/V11 (izravnalni vodnik GND in vezava oklopov niso prikazani).

Sistem po priporočilih standardov RS485/V11 je zgrajen na osnovi le ene signalne parice, na katero je mogoče paralelno povezati večje število komunikacijskih naprav (oddajnikov in sprejemnikov), tipično do 32 naprav. Električni signalni nivoji so seveda simetrični (kot pri RS422A) in omogočajo hitrosti prenosa do 10 M-Baud-ov na razdaljah preko 1 km. Ker je na omrežje tipa RS 485 vezanih več oddajno-sprejemnih naprav, mora sistem vsebovati protokol, po katerem bodo naprave zasedale skupno komunikacijsko vodilo, oddajniki pa vsebujejo posebno zaščito, ki prepreči, da bi prišlo do okvare omrežja, če bi dva ali več oddajnikov pomotoma pričela oddajati sporočila sočasno.

Poleg naštetih tipov komunikacijskih medijev je uporabnikom na voljo še mnogo drugih rešitev, npr. ETHERNET kabelsko omrežje, Thin Ethernet, omrežje za MAP sistem in drugi. Ti sistemi so primerni za hitre prenose zelo obsežnih podatkovnih paketov in so namenjeni predvsem za medračunalniški prenos podatkov. Gre za zelo obsežne in kompleksne komunikacijske sisteme, ki so jih v sodelovanju razvili veliki proizvajalci tovrstne opreme na svetu. Pojasnitev podrobnosti teh načinov bi presegala obseg tega podpoglavja, zato lahko bralec podrobnejše informacije najde v priporočeni literaturi.

Komunikacijski protokoli

Že na osnovi na Sl. 7.196 je mogoče ugotoviti, da komunikacijski sistem potrebuje "sistem vzpostavljanja povezav" - komunikacijski protokol, po katerem naprave na

skupnem komunikacijskem mediju vzpostavljajo komunikacijske poti, pri tem pa se ne motijo oziroma ovirajo med seboj. Tudi če med seboj povežemo le dve napravi, kot npr. pri povezavi tipa EIA-RS 232d, potrebujeta napravi protokol, po katerem si izmenjata sporočila, podatke. V teh primerih je primeren protokol po sistemu MASTER-SLAVE, v katerem ima ena komunikacijska naprava vodilno vlogo, druga pa podrejeno. Vsak prenos podatkov vedno začne in zaključi vodilna naprava. Podrejena naprava vedno čaka na poziv vodilne naprave in šele nato odda zahtevane podatke. Primere komunikacijskih protokolov te vrste podaja standard ANSI X3.28. V praksi je mogoče zaslediti celo vrsto izpeljank tega priporočila.

V sistemih z več postajami, kot npr. na Sl. 7.196, je sicer možno eni postaji dodeliti vodilno vlogo in vsem ostalim podrejeno, naletimo pa na celo vrsto težav (neučinkovitost prenosov, preobremenjenost vodilne postaje, občutljivost sistema za okvaro vodilne postaje, itd.). Tedaj lahko uporabimo enega izmed dveh uveljavljenih protokolnih načinov: sistem z žetonom (Token bus, Token ring) ali sistem z naključnim dostopom (CSMA/CD). Prvi sistem služi kot osnova večine sistemov za digitalno komuniciranje med napravami za procesno vodenje, drugi pa je zelo primeren in uveljavljen v sistemih medračunalniških komunikacij.

Pri protokolu na osnovi žetona potuje po komunikacijskem omrežju posebno sporočilo - žeton. To sporočilo si med seboj pošiljajo enakovredne postaje. Postaja, ki je nazadnje sprejela "žeton", začasno prevzame vlogo vodilne postaje in sme oddajati sporočila drugim postajam ali od njih zahtevati določene podatke. Ko ta postaja zaključi s komuniciranjem, odda posebno sporočilo - žeton naslednji postaji, ki tako začasno postane nova vodilna postaja. Postaja, ki ima "žeton", pa sme delovati kot vodilna postaja le določen čas. Na ta način so preprečeni komunikacijski zastoji.

Komunikacijski sistem za procesno vodenje PROFIBUS-DIN 19245, vsebuje kombinacijo protokola z žetonom in protokola MASTER-SLAVE. Med seboj enakovredne postaje si izmenjujejo vodilno vlogo s predajanjem "žetona", podrejene postaje (npr. "inteligentni" merilni pretvorniki, enozančni regulatorji, aktuatorji) pa vedno čakajo na sporočilo ali poziv katere izmed nadrejenih postaj.

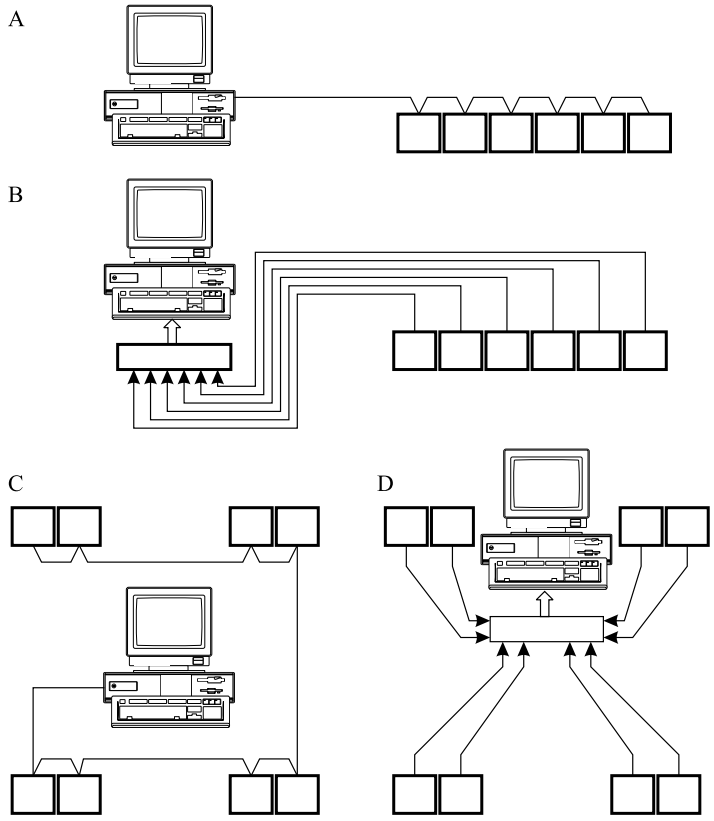
Komunikacijski protokoli na osnovi naključnega dostopa, kot npr. pri uveljavljenem računalniškem omrežju tipa ETHERNET, potekajo na ta način, da vse aktivne postaje na skupnem omrežju "poslušajo", kaj se v omrežju dogaja. Postaja, ki želi komunicirati z drugo, čaka toliko časa, dokler se trenutno potekajoča komunikacija drugih postaj v omrežju ne zaključi. Ko se zaključi, čakajoča postaja prične z oddajo svojih sporočil ali zahtev. Seveda pa se lahko zgodi, da sta na "priložnost" čakali dve ali več postaj in sta sočasno pričeli z oddajo svojih sporočil. V omrežju pride do komunikacijske zmede, ki jo udeležene postaje s posebnimi sistemi hitro detektirajo. Zato prenehajo z oddajanjem in postopek ponovijo - vendar ne takoj, ampak po preteku določenega časa, ki ga določi posebni naključni generator. Tako je malo verjetno, da bi ponovno dve ali več postaj sočasno zasedli komunikacijsko omrežje. Če pa se to vseeno zgodi, sledi ponovitev poskusne zasedbe z novimi naključnimi zakasnitvami.

Komunikacijski protokoli na osnovi "žetona" omogočajo ponovljiv in predvidljiv potek komunikacije med postajami - mogoče je napovedati, v kakšnem času lahko pride sporočilo ali podatek od ene postaje do druge. Zaradi tega so tovrstni protokoli primernejši za delovanje sistema za vodenje v realnem času. Nasprotno pa pri prenašanju obsežnejših podatkovnih paketov med računalniki časovni potek komunikacije ni strogo vezan na realni čas, zato so protokoli z naključnim dostopom primernejši, saj so učinkovitejši od protokolov na osnovi "žetona".

7.6.2.2 Nagibi za uvedbo digitalnih komunikacij v sisteme procesnega vodenja

Prvi in najpomembnejši argument, ki ga proizvajalci komunikacijske opreme navajajo v prid uporabe digitalnih komunikacij, je *bistveno* zmanjšanje obsega ožičenja. Vendarle - ali je vedno temu res tako? Sl. 7.197 prikazuje različne primere, ki ilustrirajo omenjeno problematiko.

V primerih A in B so npr. merilni pretvorniki in inteligentni aktuatorji nameščeni blizu skupaj in so nato povezani z bolj ali manj oddaljenim računalnikom. Prednost v smislu zmanjšanja obsega omrežja je v primerjavi s klasičnim ožičenjem v primeru B več kot očitna (primera A in B sta tipična primera postavitve komponent pri vodenju zaključnih proizvodnih strojev in naprav). V primerih C in D so inteligentni merilni pretvorniki nameščeni zvezdasto okrog računalniškega sistema za vodenje. To je zelo pogost primer v procesni industriji, ko se komandna kabina nahaja v središču tehnološkega procesa zato, da ima operater do posameznih naprav čim krajšo pot, ki jo mora prehoditi v primeru intervencij. Pri primerjavi variant C in D pa razlika v dolžini ožičenja ni več tako očitna, kot v primerih A in B. Če še upoštevamo, da je mogoče v primeru D posamezne pretvornike in aktuatorje povezati dvožično s sistemom z živo ničlo, v primeru C pa ima komunikacijski kabel najmanj 5 žil (komunikacijski par + skupni vod + napajalni par), potem je lahko skupna dolžina žic v primeru D celo bistveno krajša, prav tako pa je manjše tudi število sponk ali konektorskih priključkov! Izkaže se torej, da je potrebno pred odločitvijo ali uporabiti klasično signalno povezavo ali digitalno komunikacijsko omrežje, temeljito pretehtati razloge za in proti. Nekateri izmed njih so navedeni v nadaljevanju, veljajo pa seveda za medsebojno povezavo gradnikov procesnega vodenja in ne za računalniški prenos podatkov.



Sl. 7.197. Primeri komunikacijskih (A ,C) in klasičnih povezav (B, D) v sistemih procesnega vodenja. V primerih A in B so merilni pretvorniki blizu drug drugega, v primerih C in D pa so razvrščeni po prostoru.

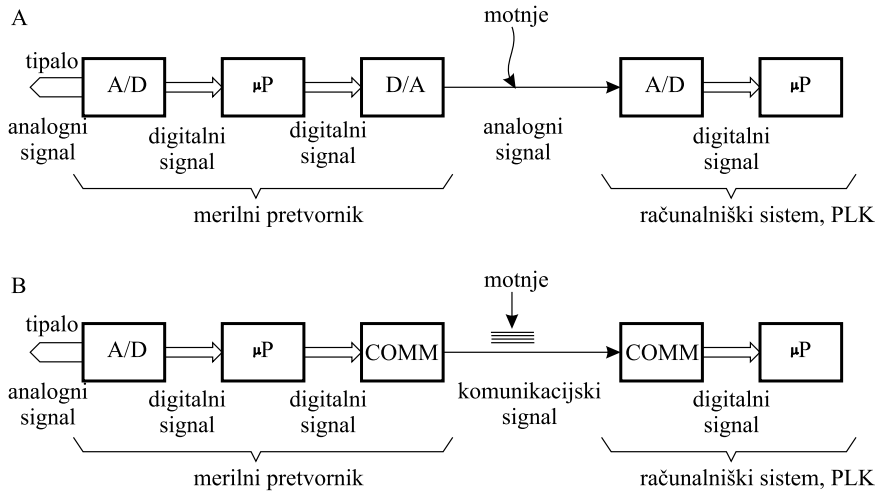
Natančnost in hitrost prenosa signalov merilnih pretvornikov in aktuatorjev

Pri sodobnih merilnih pretvornikih, ki jih povežemo na računalniški sistem in praviloma vsebujejo mikroprocesorski sistem, imamo zaporedje signalov in pretvorb, kot ga prikazuje Sl. 7.198.

Pri klasični signalni povezavi je potrebno merjeno vrednost ponovno pretvoriti v analogno obliko; pri tem jo izpostavimo motnjam iz okolice. Pri digitalnem prenosu motnje vplivajo le, če so izjemno visokega nivoja. Digitalne komunikacijske povezave zato pripomorejo k natančnejšim merjenjem procesnih veličin in krmiljenju procesnih aktuatorjev. Poleg tega vsi sprejemniki določenega signala v digitalnem komunikacijskem omrežju sprejemajo isto vrednost, v klasičnem sistemu pa se vrednosti med seboj nekoliko razlikujejo.

Klasične tokovne signalne povezave so primerne za prenos počasi se spreminjajočih signalov s frekvenčno mejo pod približno 10-20 Hz. Signal je sprejemnikom vedno na

voljo. V komunikacijskih omrežjih je pogostnost prenašanja merjenih in krmilnih vrednosti odvisna od števila komunikacijskih naprav na omrežju. Če je teh naprav malo, je mogoče vrednosti prenašati tudi nekaj 100 krat v sekundi. Pri nekoliko obsežnejših sistemih ta pogostnost hitro pade na 1 krat na sekundo, kar je mejno sprejemljiva hitrost.



Sl. 7.198. Povezava "inteligentnega" merilnega pretvornika na računalniški sistem. A - klasični signalni sistem, B - komunikacijski sistem. (A/D, D/A: analogno digitalni in digitalno analogni pretvornik, µP: mikroprocesor, mikroračunalnik, COMM: komunikacijski vmesnik)

Za uporabo digitalnih komunikacijskih postopkov so posebej primerni večsistemski merilni pretvorniki, sodobni procesni regulatorji in programirljivi logični krmilniki, pri katerih lahko po komunikacijskem vodilu prenašamo več podatkov hkrati. Primer: merilnik masnega pretoka tekočin, s katerim je mogoče hkrati meriti masni pretok, gostoto medija in njegovo temperaturo. Klasični signalni sistem bi v tem primeru zahteval 3 signalne zanke.

Izgradnja in vzdrževanje digitalnih komunikacijskih sistemov

Že iz predhodnega podpoglavja o osnovah digitalnih komunikacij je mogoče ugotoviti, da je komunikacijska tehnika relativno zahtevno področje, ki zahteva široko paleto znanj o električnih signalnih sistemih, kodiranju signalov, komunikacijskih protokolih, o pripravljanju podatkov za prenos, itd. Sodobni komunikacijski sistemi postajajo sicer modularni in jih je mogoče sestavljati tudi brez temeljitega znanja o tem področju - toda le, dokler ne gre kaj narobe. Kaj kmalu se namreč pokaže, da je potrebno pri vzdrževanju sistema in odkrivanju okvar še kako dobro poznati: kakšne so oblike in kakšni signalni nivoji v omrežju, kako so podatki oblikovani, kakšen je tip sinhronizacije med oddajnikom in sprejemnikom, kako poteka komunikacijski protokol, itd. Za vzdrževanje je potrebno imeti posebno opremo (analizatorje omrežij, namenske

osciloskope, itd.). Odkrivanje okvar zahteva za to posebej usposobljene strokovnjake, čas od nastanka (zaznave) okvare do popravila pa je zaradi kompleksnosti sistema praviloma daljši, kot pri klasičnih signalnih sistemih.

Posebna pozornost velja zanesljivosti celotnega procesnega sistema, katerega del je tudi komunikacijsko omrežje. Pri marsikaterem stroju ali napravi okvara enega merilnega ali aktuatorskega sistema zahteva ustavitev celotnega stroja in odpravo okvare. Tako tudi okvara na komunikacijskem vodilu zahteva ali sproži ustavitev celotne naprave. Pri mnogih procesih pa je ustavitev sistema nedopustna, povezana z nadaljnjim uničenjem materiala, procesne opreme, energije (primeri: zastoj steklarske peči bi pomenil zatrditev stekla in dokončno uničenje peči; parni kotel je potrebno ustavljati počasi ali nadaljevati z ročnim vodenjem, saj bi sicer povzročili hude termične ali hidravlične šoke; vsebino reaktorja za kuhanje celuloze je treba skuhati do konca, sicer se pri praznjenju zamašijo praznilne poti). V teh primerih je uporaba sistemov za vodenje na osnovi digitalnih komunikacij zelo vprašljiva. Okvara na komunikacijskem vodilu bi namreč povzročila, da bi operater "postal slep in nepokreten". Ne bi mogel odčitati vrednosti procesnih veličin, zato ne bi mogel ročno ukrepati. V klasičnih signalnih sistemih praviloma okvara v eni signalni zanki ne more ali ne sme vplivati na ostale zanke. Operater ali sistem za vodenje lahko tako vodita proces, izpadlo meritev pa je mogoče pogosto nadomestiti z lokalnim odčitavanjem procesne veličine na pomožnem tipalu, izpadli aktuator pa nadomestiti z ročnim vodenjem. Klasični signalni sistem je tako sicer obsežnejši od komunikacijskega, vendar pa ohranja visoko stopnjo robustnosti. Kjer je kljub temu digitalni komunikacijski sistem primernejši, je potrebno povečati zanesljivost s podvojitvijo komunikacijskega omrežja in nekaterih merilno-regulacijskih elementov, da ostane vodeni sistem v vsakem primeru pod nadzorom.

7.6.2.3 Komentar o uvajanju komunikacijskih sistemov v sisteme procesnega vodenja

V sodobnih sistemih procesnega vodenja se ni več mogoče izogniti uporabi digitalnih komunikacijskih sistemov med gradniki sistema za vodenje. Pri tem morajo naročnik, projektant in uporabnik skrbno pretehtati, za katere namene in v kakšni meri je smiselno namesto klasičnih signalnih povezav uporabiti digitalno komunikacijsko tehniko. Pri tem ne gre samo za vprašanje cenovnih kazalcev in trenutne "mode", pač pa je potrebno različne rešitve skrbno pretehtati.

Zmanjšanje obsega ožičenja v primeru uporabe digitalnih komunikacij, ki ga pogosto na prvo mesto postavljajo ponudniki tovrstne opreme, ni vedno najpomembnejši kriterij. Mnogo pomembneje je pretehtati, koliko digitalni komunikacijski sistem zmanjša ali poveča zanesljivost sistema, ter kdo in za kakšno ceno bo sposoben vzdrževati tak sistem. Če pride do okvare na komunikacijskem vodilu, potem se zelo verjetno zgodi, da odpovedo vse funkcije sistema, ki so vezane na medsebojno komunikacijo med elementi. V procesnih sistemih lahko to pomeni, da operater ali sistem za vodenje ne zmoreta več odčitati stanj procesa, zato ga je potrebno ustaviti. Če taka ustavitev nima

posebnih posledic (to je značilno za veliko proizvodnih strojev), potem je uporaba digitalnega komunikacijskega sistema zelo verjetno smiselna, sicer pa je bolje uporabiti klasični signalni sistem, pri katerem okvara na ožičenju povzroči izpad le enega gradnika vodenja, pri čemer pa je nadaljevanje obratovanja sistema zelo pogosto možno.

Vzdrževanje komunikacijskega sistema je še eden izmed pomembnih odločitvenih kriterijev. Klasični signalni sistem lahko vzdržuje, preizkuša in odpravlja napake oseba s srednjo ali poklicno izobrazbo ob uporabi večnamenskih in cenenih instrumentov, npr. ročnih ali namiznih multimetrov, enosmernih napajalnikov, generatorjev tokovnih signalov. Za vzdrževanje in hitro odkrivanje napak v komunikacijskih omrežjih pa mora imeti vzdrževalec specialno merilno opremo ter obvladati osnove komunikacijske tehnike. Med drugim mora poznati kodiranje signalov, sinhronizacijo med sprejemnikom in oddajnikom, obliko in signalne nivoje, komunikacijski protokol. Veliko teh informacij je žal uporabniku nedostopnih, saj jih proizvajalci komunikacijske opreme skrivajo. Uporabnik tako postane odvisen od dobaviteljev opreme!

Digitalni komunikacijski sistemi so tudi v najpreprostejših primerih relativno kompleksni in kljub številnim mednarodnim standardom omogočajo celo vrsto izvedbenih variant (npr. koliko je samo različnih variant konektorskih povezav, pa čeprav vsi proizvajalci deklarirajo skladnost s standardom EIA/RS232d!). Zaradi tega je nekompatibilnost med gradniki različnih proizvajalcev prej pravilo kot izjema, uporabnik oz. projektant pa je tako prisiljen izbirati predvsem opremo enega ali skupine proizvajalcev in s tem plačevati monopolno ceno. Velike razvojne potrebe čedalje bolj zahtevajo skladnost med gradniki na področju digitalnih komunikacij, zato bo ta omejitev v prihodnosti postajala čedalje manjša.

Že pri razvoju komunikacijskega sistema PROFIBUS se je pokazalo, da samo en komunikacijski sistem ne more pokriti vseh zahtev, ki se pojavljajo pri postavitvi sodobnih sistemov vodenja procesov. Sistem, ki omogoča zbiranje signalov merilnih pretvornikov in krmiljenje osnovnih aktuatorjev, mora zadovoljevati druge kriterije (npr. nizka cena, preprostost vzdrževanja, enostavnost ožičenja, nižje hitrosti prenosa, velika zanesljivost), kot pa komunikacijski sistem, po katerem poteka izmenjava podatkov med distribuiranimi računalniki sistema. Uporabnik in projektant morata tudi v tem primeru z izostrenim čutom za tehnični kompromis izbirati med različnimi možnostmi in zgraditi sistem z ustreznimi - skladnimi lastnostmi.

7.7 Zaključek

Kot smo videli iz podrobnejše vsebine tega poglavja, je različnih metod in postopkov, ki jih uporabljamo pri načrtovanju in izgradnji sistemov za vodenje zelo veliko, čeprav je to kar smo prikazali le izbor. Zato je mogoče ob zaključku pomembno, da še enkrat poudarimo različen namen posameznih predstavljenih skupin metod in postopkov.

Prva skupina, ki se nanaša na modeliranje in simulacijo, predstavlja zelo splošno uporabne metode, ki določajo način reševanja problema. V tem primeru gre za način

reševanja, ki temelji na abstrakciji procesa, ki ga vodimo in njegovem zapisu v obliki matematičnega modela. Takšen pristop je pri uporabi nekaterih metod vodenja, kot smo lahko videli, predpogoj, saj matematični model tako ali drugače potrebujemo pri sintezi vodenja.

Druga skupina metod je bolj specifične narave, saj zajema posamezne metode za načrtovanje različnih funkcij vodenja. Videli smo, da je stopnja kompleksnosti postopkov, ki omogočajo realizacijo posamezne funkcije zelo različna. Marsikaj se da rešiti na zelo preprost način, drugje pa je potrebno precej več znanja, če želimo doseči želeno učinkovitost vodenja.

Tretja skupina metod in postopkov, bi lahko nosila tudi ime infrastrukturna. Gre namreč za metode in postopke, ki omogočajo izvedbo funkcij vodenja. V to skupino seveda spadajo znanja in veščine s področja programske opreme in obdelave ter procesa signalov.

Kakorkoli že, dejstvo je, da pri realizaciji tudi relativno enostavnih sistemov za vodenje, bolj ali manj potrebujemo znanja iz vseh treh naštetih skupin. To seveda pomeni, da morajo biti strokovnjaki, ki se s tem ukvarjajo precej univerzalni. Ker pa je to mogoče samo do neke mere, je pri večjih projektih seveda nujno timsko delo, kjer se prepleta znanje specialistov za načrtovanje funkcij vodenja z znanjem strokovnjakov za razvoj programske opreme ter znanjem tistih, ki obvladujejo elektroniko. Sodobna orodja in gradniki, o katerih bomo govorili v naslednjih dveh poglavjih nam lahko marsikaj olajšajo, vendar pa problema učinkovite integracije sistema za vodenje ne morejo rešiti.