ISSN 1392 - 1215 ELEKTRONIKA IR ELEKTROTECHNIKA. 2005. Nr. 5(61)

T 125 AUTOMATIZAVIMAS, ROBOTECHNIKA

Valdymo uždavinių programavimas taikant projektavimo šablonų rinkinį

D. Ezerskis
Projektavimo ir programavimo skyrius, UAB „Limatika“,
Jonavos g. 62A, LT–44192 Kaunas, Lietuva, tel.+370 37 205242, faks. +370 37 205463,
el. p. darius.ezerskis@limatika.lt

R. Simutis
Automatikos ir valdymo technologijų institutas, Kauno technologijos universitetas,
Studentų g. 48, LT–51367 Kaunas, Lietuva, tel./faks. +370 37 300296, el. p. rimvydas.simutis@ktu.lt

Įvadas

Didelė dalis valdymo uždavinių yra universalaus pobūdžio ir nepriklauso nuo konkrečios taikymo srities. Pavyzdžiui, transporto, technologinių apsaugų, programuojamų valdymo įrenginių tarpusavio komunikacijos uždaviniai yra bendri visoms pramonės šakoms ir programuojamoms valdymo sistemoms. Tokių uždavinių sprendimus siekiama tipizuoti. Nuo konkrečios taikymo srities taip pat nepriklauso ir valdymo programų struktūra, ypač kai naudojamasi pagal standartą IEC 61131-3 [1] sukurtomis sistemomis.

Tokių uždavinių sprendimai daugiau priklauso nuo programuotojų patirties bei programuotojo darbino aplinkos. Kartą rastas geras sprendimas naudojamas pakartotinai, neesmingai modifikuotas konkrečiam taikymui.

Analizuojant valdymo programų kūrimo procesą, galima skirti tokius pasikartojančius programų sintezės uždavinius bei rasti tipinius jų sprendimo būdus. Šiems sprendimams aprašyti naudojami projektavimo šablonai [2], kaip būdas perduoti sukauptą sprendimų patirtį. Šablonai jau daug metų sėkmingai taikomi objektinėms programoms kurti. Šie metodai pradedami taikyti ir automatikoje [3] [4], tačiau nuoseklaus jų taikymo automatizuotoms valdymo sistemoms projektuoti atvejų autoriams nėra žinoma. Taip pat reikia pažymėti, kad automatinio valdymo uždavinių problematika yra siauresnė negu bendri informatikos uždaviniai, taigi šablonai gali būti ne tokie bendri ir labiau orientuoti į technologinio valdymo uždavinius [5].

Projektavimo šablonai

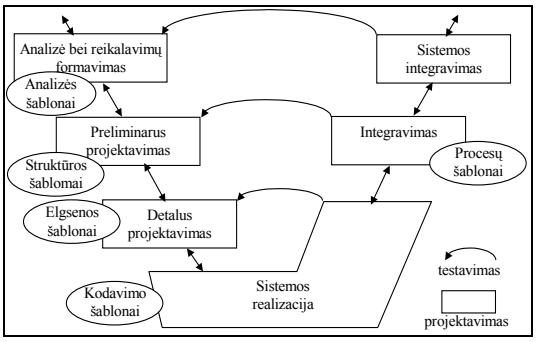
Projektavimo šablonai aprašo išbandytą ir patikrintą pasikartojančių problemų sprendimo būdą. Išbandyti ir patikrinti sprendimai dokumentuojami šablone formaliais metodais, pvz., UML [6], schemomis bei lentelėmis. Šablonai įgalina saugoti ir panaudoti konkrečias ekspertų žinias [7]. Jie papildo tiek UML, tiek specializuotus programavimo standartus, pavyzdžiui, IEC 61131-3, IEC 61499 [8], tuo, kad leidžia kurti algoritminių sprendimų bibliotekas.

Žinomi bandymai taikyti projektavimo šablonus automatizuoto valdymo sistemose. J. H. Christensen [2] nagrinėja informatikoje gerai žinomų šablonų *Distributet Application, Proxy, Model-View-Controller* (MVC) taikymą kuriant paskirstytas automatizavimo sistemas. T. Honkannen [4] siūlo taikyti šablonus aukštesnio, vadinamojo produkcijos gamybos valdymo, lygmens sistemose. P. Breitschink [7] aptaria šablonų taikymą kuriant specifikacijas bei reikalavimus įterptinėms (angl. embedded) sistemoms automobilių pramonėje.

Norint šabloną padaryti patrauklesnį projektuotojams ir programuotojams, be tipinių šablono dalių, pristatytų E. Gamma ir kt. [2] naudojamos šablono dalys, siūlomos P. Breitschinko [7], - *problemos pavyzdys* ir *realizacijos*. Šiose dalyse, remiantis konkrečiu pavyzdžiu iš automatinių valdymo sistemų srities, aiškinama problemos esmė bei siūlomas konkretus algoritmo naudojimas.

Valdymo programų kūrimo modelis

Programos kūrimo procesas gali būti suskirstytas į etapus. Šiuos etapus patogu pavaizduoti V modeliu (1 pav.).



1 pav. Programinės įrangos kūrimo procesas

Čia pavaizduota tik modelio dalis: programinės įrangos projektavimas [9]. Nepriklausomai nuo taikymo srities bei apimties ši programos kūrimo eiga daugeliui valdymo sistemų lieka tokia pat, o kiekviename etape sprendžiami iš esmės tie patys uždaviniai.

Automatizuotojo valdymo sistemą galima apibūdinti kaip sistemą, kuri per sensorius iš proceso renka duomenis, apdoroja juos pagal valdymo algoritmą ir veikia procesą per vykdymo elementus. Didėjant technologinių procesų kompleksiškumui, valdymo algoritmų kompleksiškumas didėja taip pat. Valdomo technologinio objekto struktūra bei reikalavimai valdomam procesui turi įtakos ir valdymo sistemos bei jos programinės įrangos struktūrai ir sistemai keliamiems reikalavimams. Pavyzdžiui, valdomo objekto išsidėstymas bei saugumo reikalavimai turės įtakos ir valdymo sistemos architektūrai bei saugumo ir patikimumo reikalavimams.

Automatizuotojo valdymo algoritmai apibūdinami kaip realaus laiko bei ribotų išteklių uždaviniai [7], [4], kuriems keliami didesni patikimumo bei saugumo reikalavimai. Paprastai gamybinės sistemos nuolat modifikuojamos ir modernizuojamos. Dėl to dažnai modifikuojamos ir valdymo sistemos.

Kuriamoms valdymo programoms, be funkcinių reikalavimų - korektiškumo, funkcionalumo, saugumo ir kt., keliami taip pat vadinamieji nefunkciniai reikalavimai - pakartotinis algoritmų panaudojamumas, modifikuojamumas, patikimumas, aptarnaujamumas bei skaidrumas [7].

Naudojant projektavimo šablonus, t. y. patikrintus sprendimus, paprasčiau sukurti programinę įrangą, atitinkančią nefunkcinius reikalavimus programoms. Kai kurie šablonai buvo sukurti todėl, kad jau sukurtos programos neatitiko šių reikalavimų.

Toliau kiekvienas iš programos projektavimo etapų nagrinėjamas atskirai bei pristatomi jame panaudoti šablonai.

Užduoties analizė bei reikalavimų formavimas

Šiame etape atliekama reikalavimų sistemai analizė, reikalavimų specifikacija bei visos sistemos architektūros sintezė. Tipinės šio etapo problemos išvardytos 1 lentelėje.

1 lentelė. Užduoties analizės problemos ir sprendimai

Problema	Sprendimas	Šablonas
Sistemos užduotis ir Tikslas	Technologinio proceso analizė, gamybos ypatumų nustatymas ir pan.	*Analizės šablonai*
Saugumas	Saugumo funkcijų, perėjimo į saugią būseną funkcijų, išskyrimas bei jų formavimas atskirame valdiklyje ar programos modulyje	*Saugumo valdiklis*
Architektūra	Struktūrizavimas į posistemius, funkcijų paskirstymas sukūrimui programinėje ar aparatūrinėje dalyje	Žinomi projektavimo šablonai
Standartų atitikimas	Standartų ir normų atitikimas	*Analizės šablonai*

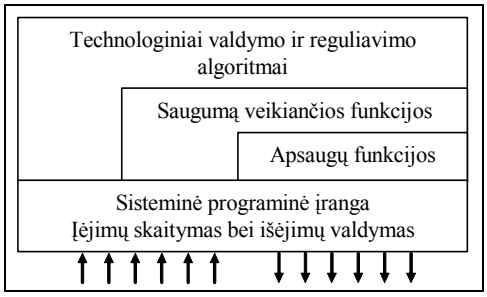
Technologinio proceso analizei galima naudoti projektavimo šablonų atmainą – analizės šablonus, pristatytus M. Fawlerio [10]. Šie šablonai sprendžia ne tik bendras analizės problemas, pavyzdžiui, laiko, ribų, funkcionalumo specifikacijų sudarymo, bet ir pačios analizės proceso organizavimo bei valdymo problemas.

Sistemos saugumo funkcijoms išskirti galima pasinaudoti analizės šablonais, o saugumo funkcijų architektūrai kurti pasitelkti autorių sukurtą ir toliau pristatytą šabloną *saugus valdiklis*.

Šablonas: saugus valdiklis

*Problema***.** Išskirti saugumo funkcijas taip, kad jos būtų aiškiai įvardijamos bei programuojamos.

*Problemos pavyzdys***.** Valdymo sistemoms keliami didesni saugumo reikalavimai. Apsauginės funkcijos turi būti ypatingai testuojamos, o kai kurių sistemų, pavyzdžiui, katilų, turbinų automatikos bei elektrotechnikos sistemų saugumą reglamentuoja specialūs standartai [11]. Tokiu atveju saugumo funkcijos turi būti sertifikuotos, taigi aiškiai išskirtos į posistemį.



2 pav. Saugaus valdiklio programinės įrangos architektūra

*Sprendimas***.** Sprendimo esmė parodyta 2 pav. Skiriamos trys vartotojo programos funkcijų rūšys. Taip siekiama sumažinti saugumo funkcijų apimtį ir padaryti jas kiek galima skaidrias bei testuojamas. Tai pageidautina todėl, kad daugelyje Europos šalių saugos funkcijos sertifikuojamos techninės priežiūros tarnybų (pavyzdžiui, TÜV), todėl jos turi būti aiškiai atskiriamos.

Naudojama nauja funkcijų rūšis kaip saugumą veikiančios funkcijos, kurioms būtina skirti daugiau dėmesio projektuojant sistemą vėlesniuose etapuose. Tai funkcijos, kurios neįeina į kritinių funkcijų sąrašą, nėra sertifikuojamos, bet gali turėti įtakos perėjimui į saugų režimą ar veikia valdomam objektui esant saugiame režime.

*Realizacija***.** Katilų valdymo programoje, funkcijos į kategorijas paskirstomos taip:

1. Apsaugų ir blokuočių funkcijos.
   - Katilo apsaugų algoritmai.
   - Automatinio katilo užkūrimo algoritmas.
2. Perjungimo į saugią būseną funkcijos.
   - Katilo būgno lygio reguliatorius.
   - Katilo būgno avarinio nupylimo valdymas.
3. Normalaus darbo algoritmai.
   - Kitų katilo reguliatorių algoritmai.
   - Katilo rėžiminės kortelės funkcijos.
   - Vartotojo sąsajos funkcijos.

Pasekmės.

- Išskiriamos saugumo funkcijos – tuo įgalinama jų nepriklausoma realizacija.
- Išskiriama sistemos dalis, kurios patikimumo skaičiavimus reikia atlikti.
- Netiesiogiai kritinės funkcijos – sistemos dalis, kuri potencialiai rizikinga, čia aiškiai apibrėžiama.

Preliminarus sistemos projektas

Šiame etape programos posistemiai įgauna savo architektūrą, algoritmas skaidomas į modulius, apibrėžiamos duomenų struktūros, specifikuojamos sąsajos tarp programos dalių bei vartotojo. Sprendžiamos problemos bei galimi šablonai joms spręsti pateikiami 2 lentelėje.

2 lentelė. Preliminaraus projektavimo uždaviniai

Problema	Sprendimas	Šablonas
Programos struktūra	Programa turi tam tikrą struktūrą: signalų paruošimas, technologinė dalis, tarnybinių funkcijų dalis ir t. t.	*Programa*
Skaidymas į technologinius vienetus	Algoritmas skaidomas iš viršaus į apačią į mažesnius technologinius vienetus iki elementarių technologinių segmentų (ETS).	*Analizės šablonai*
ETS valdymo algoritmo struktūra	ETS algoritmas dalijamas į būsenos valdymo bei būsenos palaikymo dalis, sprendžiamos sąsajos su apsaugų algoritmais	*Technologinis algoritmas*

Kaip ir kiekviena tam tikrą uždavinį atliekanti programa, valdymo programa turi dviejų tipų duomenis:

- technologinius – reprezentuojančius valdomo proceso būsenas, įvykius bei parametrus;
- tarnybinius – reprezentuojančius pačios valdymo sistemos būsenas, įvykius bei parametrus.

Taigi ir valdymo programą tikslinga pirmiausia padalyti į šias dvi dalis: technologinę ir tarnybinę.

Savo ruožtu technologinis procesas susideda iš posistemių, kurių struktūrą tikslinga atkartoti skaidant valdymo algoritmą – taip patogiau programą derinti bei aptarnauti. Technologinį procesą tikslinga skaidyti į dalis, kurias galima apibūdinti aiškia technologine funkcija.

- Reguliavimo kontūras.
- Konvejerinė transportavimo grandinė.
- Reaktorius.
- Produkto paskirstymas.
- Apsaugų grandinė ir t.t.

Tokia technologinio proceso dalis toliau vadinama *elementariu technologiniu segmentu* (ETS). ETS toliau dekomponuojamas į elementarius valdomus įrenginius (EVĮ) pavyzdžiui tokias kaip variklis, šnekas, sklendė ir pan.

Net jeigu kiekvieno ETS atliekamas technologinis uždavinys ir skiriasi, valdymo programos struktūra gali būti tokia pat ir pristatoma šablone *technologinis algoritmas*.

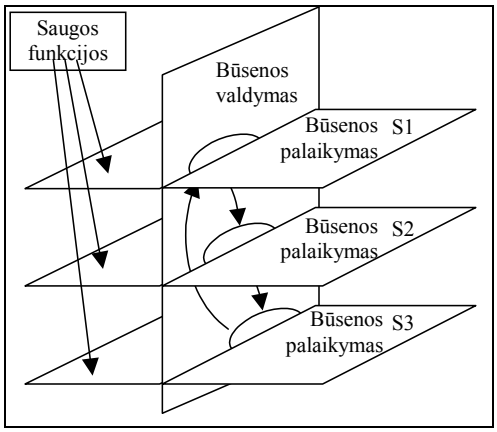
Šablonas: technologinis algoritmas

Problema. Bendra algoritmo struktūra elementaraus technologinio įrenginio (ETS) valdymui.

Problemos pavyzdys. Technologinio segmento valdymas yra kompleksiškas uždavinys susidedantis iš tokių komponentų:

- saugumo funkcijų;
- įrenginio būsenos valdymo;
- tam tikros įrenginio būsenos palaikymo;
- segmento komponentų, kurie patys gali būti kompleksiški, būsenos valdymo.

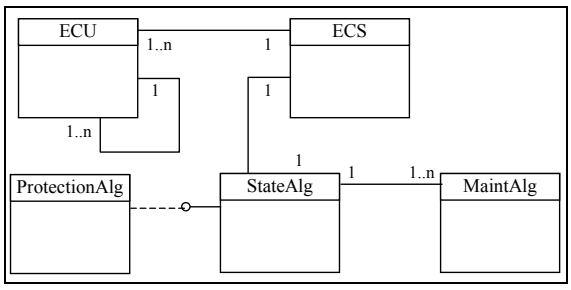
Šie uždaviniai tarpusavyje susiję, kaip parodyta 3 pav.



3 pav. Algoritmo struktūra

Sprendimas. Sprendimo modelis parodytas 4 pav. kaip UML klasių diagrama. ETS susideda iš elementarių valdomų įrenginių (EVĮ) valdymo algoritmų, diagramoje pavaizduotų bendra klase, vadinama - *ECU*.

ETS, kurio valdymo algoritmą reprezentuoja klasė *ECS*, turi vieną būsenos algoritmą, kurį sudaro bent jau būsenos „paleidimas", „darbas", „stabdymas", „avarija". Čia klasė *StatAlg* yra bazinė klasė konkrečiam būsenos algoritmui. Būsenos valdymo algoritmas taip pat turi sąsają su apsaugų algoritmu, kurį šioje diagramoje vaizduoja klasė *ProtectionAlg*. Būsenos palaikymo algoritmų gali būti ir daugiau ir tai priklauso nuo konkretaus šablono taikymo atvejo. Čia klasė *MaintAlg* yra šių algoritmų bazinė klasė.



4 pav. ETS valdymo algoritmo klasių diagrama

Pavyzdys. Panagrinėkime grūdų valymo linijos valdymo algoritmą. Valdymo algoritmo požiūriu tai yra konvejerio uždavinys: grūdai transportuojami transporteriais bei elevatoriais per valymo mašinas. Tačiau net toks paprastas technologinis procesas turi kelis darbo režimus.

1. Paleidimas: prieš pradedant veikti įrenginiams, turi būti duotas garsinis signalas, įsijungti aspiracija, atliekų perdirbimo linijos, grūdų sandėliavimo įrenginiai.
2. Valant grūdus būtina stebėti visų įrenginių darbą (valymo mašinos gali kimštis).
3. Stojant linijai, aspiracija dar kurį laiką veikia.
4. Avarijos atveju priklausomai nuo to kuris įrenginys sustojo, būtina stabdyti visą liniją arba tik jos dalį.

Tokios sistemos valdymo algoritmą patogu programuoti skirtingomis standarto IEC 61131-3 kalbomis.

1. SFC (nuoseklių sekų diagrama) realizuojamas būsenos valdymo algoritmas. Šis algoritmas turi tokias būsenas: „pradinė", „aspiracijos įjungimas", „atliekų sandėlio įjungimas", „sandėlio įjungimas", „darbas", „stabdymas", „avarija".
2. FBD (funkcinių blokų diagrama) realizuojami technologinių įrenginių (šnekų, transporterių bei valymo mašinų) valdymo algoritmai ir sąsajos tarp jų.
3. FBD realizuojami apsaugų algoritmai, kurie inicijuoja linijos stabdymą sutrikimo atveju.

Tokio programų skaidymo teigiamybės.

- SFC aiškiai vaizduoja linijos valdymo algoritmo būseną. Tai palengvina sistemą paleisti bei testuoti. Pagal poreikį būsenas galima detalizuoti – kaip šiuo atveju paleidimo būsenas – tai palengvina nustatyti linijos neįsijungimo priežastį.
- FBD kalboje programuojamas konvejerio valdymo uždavinys, šiuo atveju tai kitas, transporto šablonas (žr. lentelę 4).
- Apsaugų išskyrimas įgalina padaryti apsaugų programą mažesnę, o kartu ir skaidresnę bei lengviau suprantamą – taip sumažinama klaidų tikimybė [12].

Pasekmės. Galima derinti kiekviena ETS algoritmo dalį atskirai.

- Paprasta modifikuoti būsenų aibę keičiant valdymo algoritmą.
- Aiškios sąsajos tarp programos modulių.

Sąsajos su kitais šablonais. Saugus valdiklis, EVĮ.

3 lentelė. Detalaus projektavimo uždaviniai

Problema	Sprendimas	Šablonas
Įėj./Iš. signalų apdorojimas	Signalai turi būti paruošti naudoti paprogramėse: - analoginiai signalai konvertuoti į fizikinius dydžius, formuojamos signalų struktūros: avarinei signalizacijai, riboms, priminėms reikšmėms. - diskretiniai signalai pagal poreikį centralizuotai invertuojami, paliekamos galimybės juos filtruoti ir forsuoti.	*Signalas*
Technologinių įrenginių valdymas	Technologinio įrenginio valdymo algoritmas turi tam tikras savybes: statusą, režimo indikaciją; sąsajas žemo lygmens įrenginių tarpusavio komunikacijai.	*EVĮ*
Technologinių segmentų valdymas	Tipinių, pasikartojančių valdymo algoritmų šablonai.	žr. 4 lentelę
Reguliavimo uždaviniai	Praktikoje naudojamas reguliavimo algoritmas papildomai turi turėti reguliatoriaus užduoties perjungimo algoritmus, režimo valdymą, tolygaus režimų perjungimo algoritmus, reguliatoriaus išėjimo signalo paruošimą išvesti ir pan.	*Reguliatorius*

4 lentelė. Bendri technologinių segmentų valdymo uždaviniai

Problema	Sprendimas	Šablonas
Transporto uždavinys	Įrenginių interakcija atliekama žemame lygmenyje. Įrenginys paleidžiamas tada, kai prieš jį įrenginys jau dirba, stoja iškart, sustojus įrenginiui už jo, stoja po išsivalymo laiko, sustojus įrenginiui prieš jį.	*Konvejeris*
Įrenginių grupės perjungimas į tam tikrą būseną	Sukurti duomenų struktūrą, apimančią visų įrenginių būsenų aibę. Šioje struktūroje, struktūrų masyve saugoti visas norimas būsenas. Pagal tam tikros būsenos poreikį, priskirti įrenginiams atitinkamą masyvo elementą kaip būsenos užduotį. Žr. kelio išrinkimo šabloną.	*Kelio išrinkimas*
Paleidimo seka žingsniais	Sąveikoja su šablonu *technologinis algoritmas*. Parodo, kaip sudaryti būsenų valdymo bei perėjimo tarp būsenų algoritmus, kad jie būtų lengvai derinami, aptarnaujami bei modifikuojami.	*Paleidimo seka*
Prietaiso ar komunikacijų sutrikimo nustatymas	Generuoti ir siųsti periodinius (periodas = τ) stačiakampius impulsus ir atlikti jų periodiškumo kontrolę. Impulsui ilgiau kaip intervalą $\tau*n=2,3,...$ išbuvus vienos būsenos (TRUE arba FALSE), šio kanalo duomenis traktuoti kaip nepatikimus.	*Dinaminė kontrolė*
Proceso įvykių registravimas	Signalizacijų bei blokuočių suveikimo registravimas.	*Aliarmas*

Detalus programos projektavimas

Šiame etape projektuojami ir aprašomi algoritmo moduliai ir komponentai jau atsižvelgiant į konkretų diegimą, sukuriamos valdymo funkcijos, skirtos įrenginių valdymui, duomenų administravimui, klaidų bei sutrikimų apdorojimui.

Kadangi objekto skaidymas atliktas jau atsižvelgiant į elementarias technologines sekas, dauguma konkrečių valdymo uždavinių gali būti išspręsti naudojant šablonus. 3 lentelėje išvardyti tipiniai valdymo uždaviniai bei šablonai jiems spręsti.

Remiantis E. Gamma [2], iki šiol dauguma išvardytų projektavimo šablonų buvo struktūriniai, tačiau tipiniai algoritminiai sprendimai gali būti dokumentuojami ir elgsenos šablonais. Praktika rodo, kad 4 lentelėje išvardyti šablonai valdymo algoritmuose dažnai pasikartoja. Detaliai šie šablonai pristatyti D. Ezerskio ir R. Simučio straipsnyje [13].

Programos rašymas

Sukūrus detalų valdymo algoritmų projektą, viena iš pasirinktų programavimo kalbų rašomas programos kodas, konfigūruojami valdymo įrenginiai bei atliekamos kitos būtinos funkcijos.

Didžioji programuojamų valdymo įrenginių dalis programuojama IEC 61131-3 standarto programavimo kalbomis. Jų naudojimas motyvuojamas tokių programavimo sistemų paplitimu rinkoje ir programuotojų patirtimi programuoti šiomis kalbomis. Naudojant aukštesnio lygio kalbas, ypač grafines, gerėja programos skaitomumas bei palengvėja programos paleidimas ir derinimas.

Programos kodas rašomas darbo grupei tipine maniera bei algoritminiais sprendimais. Tokius tipinius sprendimus galima suformuoti kaip kodavimo šablonus (angl. Codding patterns). Šiuo atveju kodavimo šablonai nenaudojami patirčiai perduoti, bet reglamentuoja tam tikrą darbo grupės programavimo, komentavimo stilių.

Sistemos dalių integracija

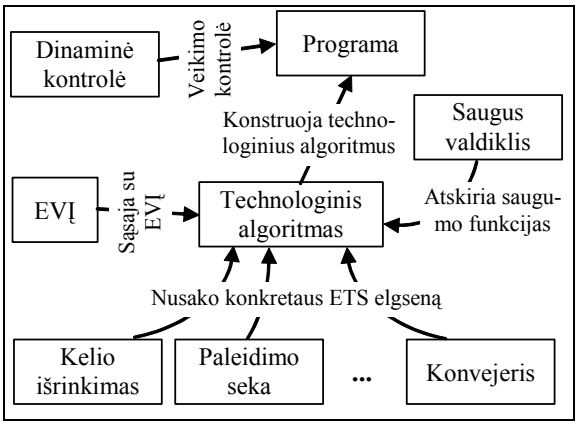
Šiame etape programos moduliai integruojami į komponentus, komponentai integruojami į posistemius (jei jų yra), o posistemiai integruojami į visą sistemą.

Iš dalies tai organizacinis uždavinys. Jis yra bendras programinės įrangos kūrimo procesams, todėl čia galima pasinaudoti gerai žinomais sprendimais, pavyzdžiui, *process patterns*, aprašytas J. Coplieno ir detalizuotas FORSOFT [14].

Reikia pasakyti, kad sistemos dalis integruoti paprasčiau naudojant šablonus, kadangi šablonai jau kuriami su tikslu, kad sistemos dalis būtų lengviau integruoti.

Sąsajos tarp šablonų

Projektavimo šablonai kuriami taip, kad spręstų tam tikrą projektavimo problemą bei savo visuma sudarytų tam tikros srities uždavinių sprendimų visumą. Anksčiau nagrinėti šablonai ir sudaro 5 paveikslėlyje vaizduojamą valdymo programų kūrimo proceso šabloninių sprendimų visumą. Vieno šablono sprendimas neretai papildo kito šablono sprendimą, taigi dalis šablonų gali būti kombinuojami tam tikrai valdymo problemai spręsti. Kiekvienas naujas patikrintas ir gerai įvertintas projektavimo sprendimas gali būti įtrauktas į šį šablonų rinkinį, ir taip laikui bėgant gali atsirasti alternatyvūs kurios nors problemos sprendimai. Tokiu atveju šablonų pasekmių dalyse tūrėtų būti nurodyta, kokiam taikymo atvejui šios alternatyvos labiau tinka.



5 pav. Sąsajos tarp šablonų

Išvados

Kuriant automatines valdymo sistemas, galima pasinaudoti tiek bendrais, žinomais informatikoje, tiek konkrečiais, savo kontekstu artimais valdomam objektui, šablonais.

Parodytas nuoseklus šablonizavimas visam valdymo algoritmo kūrimo procesui nuo sistemos analizės iki programos realizacijos bei jos paleidimo ir derinimo.

Pasiūlyti praktikoje patikrinti programos struktūros projektavimo šablonai bei elgsenos projektavimo šablonai dažnai pasitaikantiems transporto, transporto kelių išrinkimo, įrenginių paleidimo sekų uždaviniams spręsti. Be to, straipsnyje pateikti šablonų pavyzdžiai rodo patirties perdavimo galimybes kiekviename projektavimo abstrakcijos lygmenyje.

Literatūra

1. **Overview IEC 61131-3**. Standard of Programmable Controllers. Part 3: Programming languages. www.plcopen.org. [2005-02-23]
2. **Gamma E., Helm R., Johnson R., Vissides J.** Entwurfsmuster. Addison-Wesley (Deutschland), Bonn, 1996.
3. **Christensen J. H.** Design patterns for IEC 61499. Konference papers: Distributed Automation 2000. Magdeburg, Germany, 2000. http://www.holobloc.com/papers/1499_despat.zip [2003-01-29]
4. **Honkanen T.** Design Patterns in Automation. AS-116.140 Postgraduate Seminar on Information Technology in Automation. Helsinki University of Technology, 2002. http://www.automationit.hut.fi/julkaisut/documents/seminars/sem_s02/Honkanen_paper.pdf

5. **Ezerskis D., Simutis R.** Using of Design Patterns for Development and Evaluation of Computer Based Automation Systems // Information technology and control - Kaunas: Technologija, 2002. – No. 4(25). –P. 7–12.
6. **UML - OMG Unified Modeling Language Specification Version 1.5.** Object Management Group, Inc. 2003. www.omg.org/uml [2003.11.11]
7. **Braitschink P., Reuss H.C.** Formal Techniques: Patterns as a Means of Specifying Development Guidelines // Proceedings of Symposium FORMS/FORMAT 2004: Formal Methods for Automation and Safety Railvway and Automotive Systems. – TU Braunschweig, Germany, 2004.
8. **IEC 61499 Tutorial.** http://www.holobloc.com/ [2005-01-13]
9. **V-Modell.** The V-Model as Software Development Standard. http://www.v-modell.iabg.de/ . [2005-02-21]
10. **Fawler M.** Analysis patterns. http://www.martinfowler.com/articles.html [2005-02-22]
11. **IEC 61508.** Functional safety and IEC 61508. A basic guide. Geneva, 2002. [2004-11-22] http://www.iec.ch/zone/fsafety/fsafety_entry.htm
12. **Hatton L.**. Programming technology, reliability, safety and measurement // IEE Computing and Control Engineering. – 1998(9) 1. – P. 23–27 http://www.cs.ukc.ac.uk/people/staff/lh8/pubs/pubier298/PTRel_IER298.pdf.gz
13. **Ezerskis D., Simutis R**. Using design patterns for design and programming of complex automation control algorithms // Information technology and control. – Kaunas: Technologija, 2005. – No. 1(33). –P. 7–14
14. **FORSOFT**. Beyerischer Forschungsferbund Software Engineering. Technische Universität München. (Bavarian Research Association for Software Engineering. Technical Universitz Munich) http://www.forsoft.de/ [2005-02-23]

Pateikta spaudai 2005 02 10

D. Ezerskis, R. Simutis. Valdymo uždavinių programavimas taikant projektavimo šablonų rinkinį // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2005. – Nr. 5(61). – P. 89–94.

Apžvelgiamas nuoseklus projektavimo šablonų taikymas kuriant valdymo programas programuojamiems valdymo įrenginiams. Aptariamas nuoseklus, viską apimantis ir sistemingas projektavimo šablonų taikymas visuose projektavimo lygmenyse nuo uždavinio analizės iki programos kodo rašymo bei paleidimo ir derinimo. Remiantis V modelio notacija, nagrinėjama valdymo programos kūrimo eiga bei aptariamas šablonų taikymas kiekviename šių etapų. Siekiant šabloninį projektavimo problemų sprendimą pristatyti kuo aiškiau, keli šablonai paaiškinti detaliai. Pateikti šablonai patikrinti praktikoje. Il. 5, bibl. 14 (lietuvių kalba; santraukos lietuvių, anglų ir rusų k.).

D. Ezerskis, R. Simutis. Programming of Automation Control Algorithms using Set of Design Patterns // Electronics and Electrical Engineering. – Kaunas: Technologija, 2005. – No. 5(61). – P. 89–94.

There is presented an application of design patterns in a software development process for programmable automation devices. Design patterns are applied in a whole software development process from analysis to program coding and test. The software development process and the analysis of all development activities are discussed using a V-model. Typical development tasks of each activity are presented, and design patterns for each task are proposed. Some specific patterns are presented in more detail as an example for a better illustration. This set of design patterns is using in practice. Ill. 5, bibl. 14 (in Lithuanian; summaries in Lithuanian, English and Russian).

Д. Эзярскис, Р. Симутис. Примененние набора шаблонов проектирования для программирования задач управления // Электроника и электротехника. – Каунас: Технология, 2005. - № 5(61). – С. 89–94.

Рассматривается применение шаблонов проектирования в процессе разработки программ для программируемых устройств управления. Представлен путь постепенного и систематического использования шаблонов во всех этапах проектирования: от анализа проблемы до создания программного кода и пуска-наладки программы. Используя V-model рассматриваются поэтапный ход проектирования программы и шаблоны проектирования для каждого этапа. С целью нагляднее представить специфику применения шаблонов для решения проблем проектирования, некоторые шаблоны представлены детально. Представленный в статье набор шаблонов применяется на практике. Ил. 5, библ. 14 (на литовском языке; рефераты на литовском, английском и русском яз.).