KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

INFORMACIJOS SISTEMŲ KATEDRA

Tomas Žilinskas

Klausimynų projektavimo šablonų kalba

Magistro darbas

Darbo vadovas
prof. Rimantas Butleris
Konsultantai
dokt. Vytautas Taujanskas
Šarūnas Toleikis
(UAB „Baltijos IT ekspertų grupė“)

Kaunas, 2007

KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

INFORMACIJOS SISTEMŲ KATEDRA

Tomas Žilinskas

Klausimynų projektavimo šablonų kalba

Magistro darbas

Recenzentas
doc. dr. Vytautas Pilkauskas
2007-01-09

Darbo vadovas
prof. Rimantas Butleris
2007-01-09

Konsultantai
dokt. Vytautas Taujanskas
Šarūnas Toleikis
(UAB „Baltijos IT ekspertų grupė“)

Atliko
IFM-1/4 gr. stud.
Tomas Žilinskas
2007-01-09

Kaunas, 2007

Turinys

Santrauka anglų kalba (*summary*)

A pattern language for questionnaire design

Summary

This thesis describes a methodology presented in a pattern language for design of questionnaire / survey information systems. It is intended to be used by professional programmers with negligible experience in the domain of questionnaire / survey software.

A system of individual and discrete design patterns of various types is interconnected by varying relations enabling the user to navigate effortlessly and to implement a customizable level of expertise contained within the language.

The design patterns contain expert knowledge about construction and contents of a universal high-end questionnaire / survey information system. Data structures, graphical user interface and psychological ramifications of questions' formulations are discussed in detail.

The descriptions of two different prototypes of information systems are supplied. They were created using the expertise of the pattern language and serve as a proof of eligibility for it.

The innovation in the thesis is driven by the absence of pattern language for questionnaire design, usage of various types of design patterns / relations in the created language and the very design of universal metadata-based questionnaire IS described in the created language.

1. Įvadas

Klausimynų IS galutinio vartotojo požiūriu yra programa, kuri pateikia įvairios struktūros anketas su skirtingo tipo klausimais ir leidžia patogiai įvesti atsakymus.

Iš techninės pusės, klausimynai, klausimai, jų sąryšiai ir pateikties charakteristikos gali būti aprašomos metalygmenyje. Metalygmuo tokio tipo sistemose yra būtinas, kad sistema būtų universali ir praplečiama.

Kurti lanksčią klausimynų valdymo IS yra tikras iššūkis.

Kad turėtų bent minimalių komercinių perspektyvų, sistema turi leisti pildyti įvairius, skirtingus ir sudėtingus klausimynus, ir tenkinti kintančius poreikius ateityje. *Todėl klausimynų projektuotojas turi puikiai išmanyti programavimą / modeliavimą metalygmenyje.*

Tokio tipo sistemos turi didelį „kontakto paviršiaus plotą“, t.y. vartotojui skirtingais pavidalais pateikiama daug įvairių klausimų, ir jam turi būti sudarytos sąlygos kuo tiksliau ir kuo lankstesne forma į juos atsakyti. *Dėl šios priežasties, klausimynų projektuotojas turi būti vartotojo sąsajos kūrimo specialistas*.

Akivaizdu, kad modeliuojant vartotojo sąsają reikia minimizuoti duomenų įvedimo klaidų kiekį. Bet to nepakanka – būtina turėti omenyje žmogiškuosius faktorius, kurie lemia įvedamų duomenų atitikimą tikrajai apklausiamojo nuomonei. Į šiuos faktorius būtina atsižvelgti formuluojant klausimus ir sudarant klausimynus IS metaduomenų formoje, nes kokybės prasme žmogiškosios paklaidos eliminavimas ne mažiaus svarbus už teisingą IS algoritmą. *Todėl klausimynų projektuotojas turi išmanyti psichologinius klausimynų sudarymo aspektus.*

Tiriamasis darbas inicijuotas suvokus, kad neegzistuoja teorinis pagrindas, apimantis ir programinius / architektūrinius / ergonominius klausimynų IS projektavimo, ir socialinius / psichologinius klausimynų sudarymo aspektus.

Šis darbas yra bandymas integruoti pakankamai nutolusias – IS projektavimo ir IS metaduomenų turinio optimizavimo, – disciplinas. Kaip rodo literatūros analizė, tai yra vienas *iš labai retų* bandymų.

Darbo tikslas – sukurti ir pateikti profesionalių, bet klausimynų dalykinėje srityje nepatyrusių programuotojų apmokymo metodiką, leidžiančią greitai ir neskausmingai padidinti produktyviam darbui būtinas specifines žinias.

Kaip „bendras mažiausias daliklis“ tarp IS projektavimo ir IS metaduomenų turinio optimizavimo pasirenkamas *projektavimo šablonas*, o darbo esmę sudaro tokių šablonų sistemos (projektavimo šablonų kalbos) aprašymas.

Vienas iš argumentų mokomąją metodiką išreikšti kaip projektavimo šablonų kalbą – šis formatas yra tarpinis tarp formalios matematinės specifikacijos ir nestruktūrizuoto teksto. Kitas ne mažiau svarbus argumentas – profesionaliems IS projektuotojams šis formatas jau puikiai pažįstamas, nes dažniausiai būtent šiuo formatu aprašoma objektinės orientacijos projektavimo geroji patirtis.

Darbe pateikiama metodika turi padėti klausimynų IS vystymu užsiimančiam profesionaliam programuotojui *norimu detalumo lygmeniu* perimti skirtingų disciplinų – IS duomenų struktūrų modeliavimo, vartotojo sąsajos kūrimo ir klausimynų sudarymo, - žinias.

Šiuo formatu aprašyti „socialiniai“ šablonai prieinami ir socialinių tyrimų specialistams, apmokytiems įvedinėti klausimynus metaduomenų forma.

Darbe stengiamasi pagrįsti prielaidą, kad projektavimo šablonai yra optimalus formatas unifikuotai aprašyti tradiciškai nutolusių disciplinų ekspertines žinias, ir į programuotojų kalbą (ir mąstymą!) įvesti sąvokas, kurios sunkiai prigytų, jei būtų aprašytos pagal programuotojams nepatogią sistemą, aprašytos be sistemos, ar neaprašytos visai.

Analizuojant literatūros šaltinius nepavyko aptikti nuorodų į jokias klausimynų IS projektuoti skirtų kalbų specifikacijas. Šis darbas yra bene pirmoji tokiam tikslui skirta projektavimo šablonų kalba.

Darbe pateikta ne tik projektavimo šablonų kalbos bazinės struktūros, bet ir numatyta kalbos išplėtimo galimybė: tai pasiekiama per kokybiškai skirtingus projektavimo šablonų ryšius, bei projektavimo šablonų tipizavimą pačioje kalboje.

Aprašyta metodika buvo išplėtota ir įdiegta UAB „Baltijos IT ekspertų grupė“ (tai patvirtina prieduose pateikiamas įdiegimo aktas). Šio tiriamojo darbo prasmingumą ir pagrįstumą įrodo pagal šią metodiką bendrovės specialistų realizuotos dviejų skirtingų tipų klausimynų IS, kurios trumpai apžvelgiamos ir šiame darbe.

Klausimynų bei projektavimo šablonų bazinė terminologija ir sudėtis aptariami ekspertinių žinių apie klausimynų IS pateikimą analizės dalyje. Apsibrėžus sąvokas ir pasirinkus metodikos variantą, pereinama prie kalbos aprašo, pateikiant visus projektavimo šablonus ir ryšius tarp jų.

Pateiktos projektavimo šablonų kalbos padengimas (šablonų pagrįstumas) ir taikymas aptariami praktinio IS kūrimo skyriuje, pateikiant trumpus dviejų sukurta kalba pagrįstų klausimynų IS aprašus. Darbas apibendrinamas išvadomis ir tolesniais planais; išvardijama naudota literatūra ir terminai. Prieduose pateikiamas konferencijoje pristatytas straipsnis.

2. Ekspertinių žinių apie klausimynų IS kūrimą pateikimo analizė

Nagrinėjama struktūrizuoto informacijos surinkimo priemonė – klausimynai (angl. *questionnaires*), naudojama marketingo, produktų valdymo srityse.

Klausimynų struktūra, taikymas ir elgsena yra beveik nereglamentuoti, formaliai neapibrėžti. Nesusisteminta geriausių, laiko patikrintų sprendimo būdų patirtis, todėl naujose universalių IS, valdančių klausimynus (čia: klausimynų IS), realizacijose kartojamos tos pačios klaidos.

Darbo specifika apima šias sritis:

- IS kūrimo metodologijos;
- Sistemų analizė ir projektavimas;
- IS projektavimo šablonai.

Apibrėžus klausimynų dalykinės srities sąvokas (žr. 2.1, 2.2, 2.3) nustatomas tinkamiausias metodas ekspertinių žinių apie klausimynų IS kūrimą išsaugojimui / formalizavimui (žr. 2.5). Pasirinktas metodas toliau analizuojamas ieškant optimaliausios konfigūracijos (žr. 2.6, 2.7).

2.1. Klausimynų srities esybės

Klausimynų terminologija nėra griežtai reglamentuota. Egzistuoja bendriausios sutartos sąvokos, pvz., [Bra+04]. Psichologiniai aspektai gvildenami [Bie+03]. Darbas reikalauja vienodo sąvokų traktavimo, todėl žemiau pateikiama normalizuota ir esamai klausimynų IS idėjai pritaikyta versija.

2.1.1. Klausimas

Bazinis klausimyno elementas – klausimas, nusakantis konkrečios informacijos poreikį. Klausimo formuluotė gali būti pateikiama įvairiai:

- Išreikštine – tiesiogiai, apibrėžtai;
- Implikuota – klausimo formuluotė skaitytojo turi būti išgryninta iš klausimo konteksto;

Klausimai gali būti pateikiami įvairiose terpėse:

- Tekstu (popierine ar elektronine forma),
- Srautiniu tekstu (popierinėmis arba elektroninėmis formomis),

- Garsu (telefoninės sistemos).

2.1.2. Atsakymas į klausimą

Atsakymo tipas (čia ir toliau - *domenas*) nulemia reikiamos informacijos pobūdį.

Klausimų domenai savo ruožtu gali būti skirstomi į keturis potipius:

- *Diskretieji*: leidžiama parinkti tam tikrą kiekį (apibrėžiamą atsakymo *kardinalumu*) atsakymų iš fiksuotos jų aibės;
- *Elementarieji*: priima vieną ar kelias reikšmes, įvestas vartotojo. Reikšmės gali priklausyti bet kuriam iš paprastųjų duomenų tipų: tekstui, skaičiams, datai. Šiam tipui priskiriama ir grafika bei dvejetainiai duomenys. Tik šio tipo domenai gali būti kombinuoti, t.y. papildantys reikšmę matavimo vienetais, kurių aibė visuomet pateikiama diskrečiąja forma;
- *Sudėtiniai paprastieji*: įgalina priskirti (kaip atsakymą) anksčiau užpildytą tam tikro klausimyno (šiuo atveju - *subklausimyno*) egzempliorių arba priskirti naujai užpildyti tuščią subklausimyno egzempliorių;
- *Sudėtiniai kompoziciniai*: nuo sudėtinių paprastųjų skiriasi kardinalumu: galima priskirti / sukurti naujai ne vieną, o nurodytą skaičių subklausimyno egzempliorių. Negana to, subklausimynų, kurių egzempliorius galima priskirti / sukurti naujai, gali būti keli tam pačiam domenui.

Nepriklausomai nuo domeno tipo, klausimas gali turėti loginius apribojimus (angl. *logical constraints*) atsakymo reikšmėms. Paprastai apribojimą sudaro aritmetinės išraiškos.

2.1.3. Klausimynas

Klausimyną sudaro tam tikrais ryšiais susietų klausimų aibė. Tik čia prasmę įgyja tam tikri klausimo atributai, pvz.:

- *Išbaigtumas* (angl. *completeness*): atsakymas į klausimą laikomas išbaigtu, jeigu įvesta reikšmė(-s) atitinka domeno duomenų tipą, kardinalumo bei loginiai apribojimai yra tenkinami;
- *Būtinumas* (angl. *mandatory state*) – ar atsakymo į šį konkretų klausimą išbaigtumas lemia viso klausimyno išbaigtumo būklę.
- *Matomumas* (angl. *visibility*) – ar klausimas, atsižvelgiant į užpildymo situaciją klausimyne, turi būti rodomas vartotojui;

- *Keičiamumas* (angl. *entry state*) – ar klausimas gali priimti atsakymą iš vartotojo. Kai kurie klausimai įgauna atsakymus, skaičiuojamus automatiškai arba užpildytus prieš pateikiant klausimyną vartotojui. Tokiais atvejais klausimas būna nekeičiamas (angl. *read-only*);

Klausimyno egzemplioriaus minimalus tinkamumas (angl. *minimal fill*) nusako jo panaudojimo galimumą iš anksto nusakytame kontekste (žr. 2.2 punktą). Klausimynas laikomas minimaliai tinkamu, jeigu visi jame esantys klausimai, pasižymintys būtinumu, yra išbaigti. Pavyzdžiui, apklausiamojo vardas, pavardė ir šeimyninė padėtis beveik visada įeina į reikalavimus minimaliam tinkamumui.

Tolimesnė kategorija – klausimyno egzemplioriaus visiškas tinkamumas (angl. *complete fill*), - išplečia minimalaus tinkamumo sąvoką, reikalaujant visų klausimyne esančių matomų klausimų išbaigtumo (žr. skyriaus pradžią).

Aukštame panaudojimo lygyje klausimynai (struktūriškai; praktiškai – klausimynų egzemplioriai) gali būti hierarchiškai įterpiami (angl. *nesting*) vieni į kitus. Tai atliekama sudėtinių domenų priemonėmis (žr. 2.1.2 punktą). Dažniausiai kiekvienas klausimyno egzempliorius užpildytas įgyja unikalų identifikatorių, pvz. pildytojo vardas, pavardė bei klausimyno formos numeris. Šių identifikatorių pagrindu ir įmanoma susieti klausimynus.

2.1.4. Ryšiai tarp klausimų

Klausimai gali būti susieti priklausomybės ryšiais (angl. *dependency links*) klausimyno kontekste. Visais atvejais ryšys sieja du klausimus, vadinamus tėviniu (angl. *parent*) ir vaikiniu (angl. *child*). Ryšiai skirstomi į du tipus:

- *Egzistenciniai*: tėvinio klausimo loginio apribojimo teisingumas nulemia ryšio egzistavimą, o tuo pačiu – ir vaikinio klausimo matomumą (žr. 1.2.3 punktą). Dažniausiai naudojama forma – nulinis apribojimas, t.y. vaikinis klausimas visuomet prijungiamas prie tėvinio (netaikant jungčiai jokių sąlygų);
- *Procedūriniai*: tėvinio klausimo reikšmė ir / ar loginio apribojimo teisingumas keičia vaikinio klausimo charakteristikas: pavadinimą (angl. *piping*), domeno tipą, dinamiką.

2.2. Klausimynų taikymo kontekstas

Klausimynai, kaip lanksti struktūrizuotos informacijos įvedimo priemonė, „filtras“, yra taikomi įvairiose srityse. Svarbiausios iš jų:

- *Rinkodara*: įvairios apklausos naudoja klausimynus, kurie vis dažniau pateikiami kompiuterizuota forma;
- *Biurokratija*: mokesčių, balsavimo, pareiškimų formos labai dažnai apima klausimynus, dažniausia pateikiamus srautinio teksto pavidalu;
- *Produktų valdymas* (angl. *product information management, PIM*): produkto esybės gana sėkmingai gali būti atvaizduojamos klausimynais (produkto atributus / subproduktus pateikiant klausimais). Tokia forma duomenų įvedimą padaro natūralų, nuoseklų. Be to, tie patys klausimynai gali būti panaudoti ir duomenų paieškai, papildomai supaprastinant naudojimąsi sistemomis. Beveik visais atvejais produktų valdymo sistemos yra kompiuterinės.

Tradiciniai informacijos surinkimo metodai klausimynams nusileidžia pragmatiniu aspektu:

- *Žodinė apklausa* – ilgiau užtrunka, gautus atsakymus sunku prisiminti, todėl atsiranda poreikis juos užsirašyti, taip nejučia pereinant prie klausimynų naudojimo apklausėjo pusėje;
- *Nestruktūrizuotas žinių teikimas* – apklausiamasis, neturėdamas reikalavimų šablono, apibūdina objektą remdamasis savo pasaulėžiūra. Tai sukelia didelių sunkumų nustatant apklausiamųjų vertybinių skalių santykius, tinkamą informacijos įforminimą fiksuotos struktūros duomenų saugyklose.

2.3. Klausimynų IS samprata

Šiame darbe terminu „klausimynų IS“ apibūdinama IS su specifinėmis savybėmis:

- Vartotojui pateikiamas klausimynas, į kurio klausimus jis gali įvesti atsakymo reikšmę(-es);
- Klausimynas organizuojamas medžio forma (pagal 2.1);
- Minimaliai tinkamas (žr. 2.1.3) klausimyno egzempliorius gali būti išsaugotas, pakeičiamas, užkraunamas iš duomenų bazės;
- IS yra universali šiais bruožais:
    - Visi klausimynų bei jų komponenčių , kaip esybių, aspektai yra saugomi metaduomenų bazėje (t.y. taisyklių saugykloje; angl. *metadata repository*) ir gali būti modifikuojami nekeičiant programinio kodo (angl. *soft-coded*);
    - IS gali turėti metaduomenų apie norimą kiekį klausimynų ir leisti vartotojui aktyvuoti kiekvieną iš jų sukuriant naujus egzempliorius /

atliekant paiešką tarp egzistuojančių egzempliorių / atidarant egzistuojančius egzempliorius;

- Kiekvienas klausimynas turi leisti prie jo egzemplioriaus prijungti metaduomenų lygyje apibrėžiamą vaikinių klausimynų konfigūraciją pagal nurodytą kardinalumą (pagal 2.1.3);
- Klausimynų hierarchijos gylis ir sudėtingumas nėra ribojamas (išskyrus, žinoma, aparatūrinės įrangos / sisteminės programinės įrangos limitus);

- Pageidautina, kad klausimyno egzemplioriuose būtų galima atlikti neriboto sudėtingumo paiešką pagal kintamą kriterijų kiekį;
- Atsakymų į klausimus tipų bei klausimų ryšių apribojimų įvairovė turi būti pakankama, kad klausimynų IS tiktų įvairių tipų (pagal 2.2) klausimynų valdymui;
- IS savarankiškumui įtvirtinti pageidautina turėti metaduomenų redaktorių, įgalinantį redaguoti klausimynų struktūrą.

Tokia IS – kompleksiška, sudėtinga ir daug galinti – yra galutinis tikslas, kuriam pasiekti dažnai pritrūksta laikui bėgant pamažu įgyjamų ekspertinių žinių.

2.4. Ekspertinių žinių vartotojas

Klausimynų dalykinės srities bei klausimynų IS ekspertinės žinios yra reikalingos klausimynų projektuotojui. Tai – rolių amalgama, būdinga mažoms IT firmoms su nedideliu kiekiu aukštos kvalifikacijos specialistų.

Klausimynų projektuotojas atlieka šias roles:

- Kuria, modifikuoja ir palaiko klausimynų IS kodą;
- Įveda, pritaiko, modifikuoja (t.y. valdo) metaduomenis, pagal kuriuos apibrėžiami klausimynai naudojimui klausimynų IS.

Kitaip tariant, klausimynų projektuotojas prižiūri klausimynų IS struktūrą bei valdo aukščiausio lygio turinį (metaduomenis).

Kadangi „klausimynų IS“ buvo apibrėžtas kaip universali, jos sukūrimas reikalauja nemažos patirties „bandymų-klaidų“ metodu. Patirties įgijimo laikotarpio sumažinimui pravartu ekspertines žinias perduoti tiesiogiai, išreikštine forma.

Antroji rolė – turinio valdymas, - išplaukia iš eiliniam vartotojui draugiško metaduomenų redaktoriaus nebuvimo – dažniausiai dėl finansinių / laikinių / nepakankamo

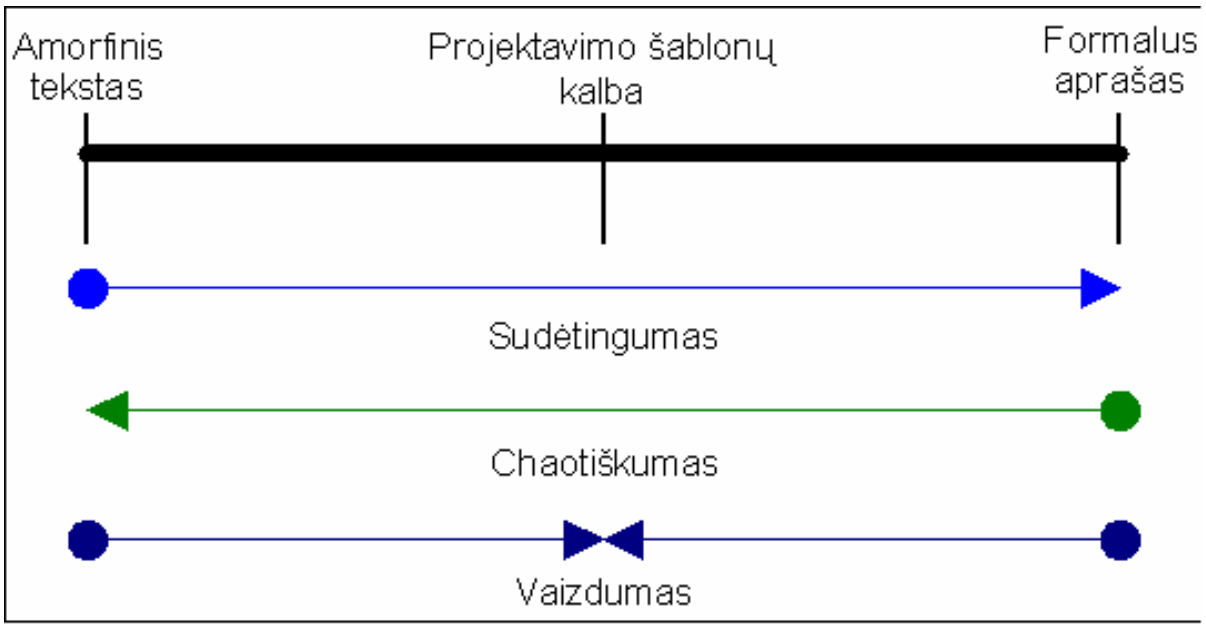
poreikio priežasčių. Tokiu atveju tik klausimynų IS kūrėjas yra pakankamai kompetentingas ir spartus valdyti metaduomenis, todėl ši pareiga papildo esamą.

Gana dažnai užsakovo pateikiami reikiamų įvesti klausimynų aprašymai būna paimti iš tradicinės formos apklausų, išgauti (angl. *harvested*) iš dalykinės srities ar žmonių nuomonių. Tokiais atvejais būtina performuluoti klausimus, pakeisti atsakymų charakteristikas, klausimų apjungimo tipus. Šie veiksmai, įeinantys į turinio valdymo rolę, dažnai šliejasi prie psichologinių / socialinių doktrinų, taigi, reikalauja tarpdisciplininio išsilavinimo.

2.5. Ekspertinių žinių pateikimo metodo parinkimas

Probleminę sritį galima aprašyti ir jos ekspertines žinias perteikti:

- *Paprastu tekstu*: nestruktūrizuota prezentacija, suprantama įvairių sričių specialistams be papildomo pasirengimo. Labai nevaizdi. Didėjant apimčiai painumas didėja eksponentiškai;
- *Projektavimo šablonų kalba* (žr. 2.7) – vidutiniškai struktūrizuotas hierarchinis aprašas, besiremiantis žingsnine „top-down“ kūrimo proceso dekompozicija. Tinkamai parinkus formatus, gali būti suprantamas plačiam specialistų diapazonui, gali aprašyti įvairius dalykinės srities aspektus;
- *Formaliu matematiniu aprašu* (pvz., *Z-notation*) – griežtai teisingas ir labai struktūrizuotas aprašas. Suprantamas tik specialų pasirengimą turintiems tiksliųjų mokslų specialistams, negali aprašyti subjektyvių ar taisyklių neturinčių procesų;



1 pav. Pateikimo metodų palyginimas

Kaip matyti variantų palyginime (1 pav.), spektras svyruoja nuo vieno kraštutinumo (amorfinis, prieinamas) iki kito (labai struktūrizuotas, specializuotas). Todėl pasirenkamas „aukso vidurys“ – projektavimo šablonų kalba, turinti didelį lankstumą ir pritaikomumą.

Išsami paieška neatskleidė kitų mėginimų sukurti klausimynų projektavimo šablonų kalbą (projektavimo šablonų saugyklose (angl. *repositories*), teminių *PLoP* konferencijų medžiagoje), todėl palyginimas su alternatyviais mėginimais nėra galimas.

Toliau pateikiama projektavimo šablonų kalbos esminių komponentų – projektavimo šablonų, - analizė.

2.6. Projektavimo šablonų specifika

„Kiekvienas projektavimo šablonas – tai taisyklė, susidedanti iš trijų dalių, išreiškianti sąryšį tarp tam tikro konteksto, problemos ir jos sprendimo“ [Ale+77] – taip koncepcijos kūrėjai (vedami žinomo architekto Christopher Alexander) apibrėžia vieną pagrindinių ekspertinių žinių nešėjų.

Projektavimo šablonai (angl. d*esign patterns*) įgalina nusakyti sprendimą konkrečiai dažnai pasitaikančiai problemai ir jį aprašyti pakankamai abstrakčiai, jog būtų išsaugoma eksperto interpretacija tuo klausimu. Šiose plotmėse projektavimo šablonus galima interpretuoti ir kaip mokymo priemonę (tam tikros metodikos dalį), ir kaip projektavimo proceso įrankį, pagreitinantį tipinių problemų sprendimą bei pakeliantį projekto kokybės lygį.

2.6.1. Tipai

Pagal ištakas projektavimo šablonai yra skirstomi į du tipus [Ale+77] su skirtingu pritaikymu ir galimybėmis:

- *Negeneruojantys* (angl. *non-generative*) – sukuriami išvedant (angl. *mining*) empirines taisykles iš egzistuojančių sistemų. Kadangi pasikartojantys reiškiniai egzistuojančiose sistemose nebūtinai yra optimalūs ar sektini, šios rūšies šablonai gali būti neoptimalūs arba netikslūs. Dar vadinami *Gamma šablonais* (pagal Erich Gamma, vieną iš *GoF* narių);
- *Generuojantys* (angl. *generative*) – sukuriami remiantis teorija arba dėmesingai ir kruopščiai išvedami iš egzistuojančių sistemų. Pagal šį apibrėžimą, generuojantys šablonai yra pavyzdiniai ir sektini;

2.6.2. Taikymo sritys

Ryškus projektavimo šablonų bruožas – universalumas: jie gali aprašyti labai abstrakčias problemas, gali ir detalizuoti labai siauro pritaikymo patirtį iš konkrečios probleminės srities. Keletas dažnai pasitaikančių projektavimo šablonų taikymo sričių (pavadinimai – sąlyginiai, nesisteminiai):

- Kompiuteriniai - savo taikymo dažnumu ir apimtimi pralenkė kitų dalykinių sričių (įskaitant pradinę - architektūrą) lygius;
    - *Programiniai* – aprašo (dažniausiai) objektinio projektavimo taikomuosius šablonus detaliu realizacijos lygiu (klasės, objektai, kreipiniai). Naudojami profesionalių programuotojų. Pavyzdys - [Gam+95], populiarūs ir dar žemesnio lygio (pvz., individualių programavimo kalbų specifikai atskleisti);
    - *Architektūriniai* – aprašo programinės įrangos architektūrą aukštu (komponentų, posistemių, modulių) lygiu. Pavyzdys – chrestomatinis *MVC* (angl. „*Model-View-Controller*“) [Bus+96] šablonas, paprastai žinomas net ir šablonais nesidomintiems programuotojams. Kai kurie autoriai (pvz., minėtas [Bus+96], šį tipą išskiria į alternatyvią šablonų rūšį – architektūrinius šablonus (angl. *architectural patterns*));
    - *Vartotojo sąsajos* – apibrėžia sprendimus vartotojo sąveikos su skaičiavimo mašinomis (angl. *Human-Computer Interaction*, *HCI*)

problemoms. Dažniausiai nėra detalūs programavimo prasme, apima *HCI* lygio veiksmus;

- *Teoriniai* (įvairios sritys) – apibendrina bendražmogiškąsias žinias; sprendimus galima pritaikyti įvairiose probleminėse srityse (papildant konkrečiomis detalėmis). Naudojami projektuotojų. Pavyzdys – šablonų saugykla [PLT06];
    - *Psichologiniai* – nusako pageidautinas reakcijas į žmogaus-mašinos ar žmogaus-žmogaus santykius, tarpusavio reakcijas. Pavyzdžiai: – „kaip reaguoti į parduotuvėje tyčia verkiantį vaiką“ (tarpžmogiškas) arba [Ada+95] (sąveika su mašina);

2.6.3. Struktūra

Įvairūs autoriai skirtingai įsivaizduoja projektavimo šablonų struktūrą, ją pritaiko savo konkretiems poreikiams. Didelę įtaką daro ir panaudojimo kontekstas, dalykinė sritis. Dažniausiai taikoma jau chrestomatine tapusi „*Gang of Four*“ notacija (žr. 2.6.3.1), kuri savo ruožtu yra išvesta iš projektavimo šablonų atradėjo Christopher Alexander schemos.

2.6.3.1. Alexander formatas

Tai – originali projektavimo šablono forma, skirta architektams. Įvairiai modifikuotos (ir pritaikytos kompiuterijos specifikai) versijos naudojamos vėlesniuose kitų autorių darbuose. Paties autoriaus žodžiais [Ale+77] pateiktas apibrėžimas:

> „Siekiant patogumo ir aiškumo, kiekvienas šablonas aprašomas tuo pačiu formatu. Pradžioje patalpinamas paveikslėlis, vaizduojantis archetipinį šablono pavyzdį. Už paveikslėlio eina įvadinis paragrafas, nusakantis šablono kontekstą – t.y. paaiškinantis, kaip šis šablonas padeda užbaigti nurodytus aukštesnio lygio šablonus. Po to trys rombai žymi problemos pradžią. Po jų rašoma antraštė (pariebintu šriftu). Antraštė pateikia problemos esmę vienu-dviem sakiniais. Toliau pateikiamas pats problemos aprašymas. Tai – ilgiausia dalis. Ji apibūdina empirinį problemos kontekstą, jos aktualumo įrodymus, skirtingų pasireiškimo būdų pastatuose spektrą ir t.t. Po to (vėlgi pariebintu šriftu, kaip antraštei) pateikiamas sprendimas – projektavimo šablono šerdis, - apibūdinanti aibę fizinių ir socialinių sąryšių, kurie reikalingi, norint išspręsti iškeltą problemą nusakytame kontekste. Sprendimas visada pateikiamas

instrukcijos forma – kad tiksliai žinotumėte, ką reikia nuveikti šablonui realizuoti. Sprendimą papildo diagrama su žymomis, nurodančiomis pagrindinius komponentus. Po diagramos dedami trys rombai, žymintys šablono pagrindinės dalies pabaigą. Galų gale, po rombų matomas paragrafas, susiejantis nagrinėjamą šabloną su mažesniais tos pačios kalbos šablonais, reikalingais nagrinėjamam šablonui užbaigti, jį papildyti ar išpildyti.“

Kaip matyti, *Alexander* kreipia didelį dėmesį į šablonų sąsajas „*top-down*“ stiliaus hierarchinėse projektavimo šablonų struktūrose, išsamų problemos aprašymą. Pagrindinės dalys: problema-sprendimas-diagrama kitų autorių buvo plečiamos ir skaidomos dalimis.

2.6.3.2. GoF formatas

Vadinamosios „ketveriukės“ (angl. „*Gang of Four“, GoF*) autoriai savo monumentaliame veikale [Gam+95], atvedusiame projektavimo šablonus į kompiuteriją, naudojo specifinį, detalizuotą projektavimo šablonų formatą, pritaikytą gana žemo lygio programavimo sąvokų abstrahavimui (tapusį pagrindu „programiniams“ šablonams; žr. 2.6 punktą).

- *Tikslas* (angl. *intent*) – trumpas projektavimo šablono paskirties aprašas;
- *Motyvacija* (angl. *motivation*) – Pavyzdžiu paremtas problemos bei sprendimo aprašymas sklandžiu tekstu;
- *Pritaikymas* (angl. *applicability*) – sąlygos, kurias patenkinus aprašomasis projektavimo šablonas gali būti pritaikomas aprašytame kontekste;
- *Struktūra* (angl. *structure*) – *OMT* diagrama, vaizduojanti sprendime dalyvaujančių klasių ir / ar objektų ryšius;
- *Dalyviai* (angl. *participants*) – sprendime dalyvaujančių esybių aprašas (žodynas);
- *Sąveikos su aplinka* (angl. *collaborations*) – projektavimo šablone apibrėžtų klasių bendradarbiavimo su aplinka atvejai. Gali būti papildomi sąveikos (angl. *interaction*) diagramomis;
- *Pasekmės* (angl. *consequences*) – nagrinėjamo projektavimo šablono panaudojimo pasekmės: teigiamos ir neigiamos;
- *Realizacija* (angl. *implementation*) – patarimai sėkmingam projektavimo šablono pritaikymui, perspėjimai dėl galimų kliaučių;
- *Pavyzdžio programinis kodas* (angl. *sample code*) – projektavimo šablono siūlomą sprendimą atspindintis kodas *C++* kalba;

- *Praktinio pritaikymo pavyzdžiai* (angl. *known uses*) – aprašomi keli gerai žinomi nagrinėjamo projektavimo šablono taikymo pavyzdžiai;
- *Susiję projektavimo šablonai* (angl. *related patterns*) – kiti projektavimo šablonai, naudojantys nagrinėjamąjį arba naudojami nagrinėjamajame. Panaudojimo aplinkybės;

2.6.3.3. Coplien formatas

Jim Coplien, žymus projektavimo šablonų evangelistas, siūlo paprastesnę schemą [Cop98], perėmusią bruožus iš *Alexander* ir *GoF*:

- *Problema* (angl. *problem*) – detalizuojama, kokie trūkumai sprendžiami aprašomu projektavimo šablonu;
- *Kontekstas* (angl. *context*) – aplinka, kurioje egzistuoja probleminės esybės;
- *Veikiančios jėgos* (angl. *forces*) – aplinkybės, kriterijai ir apribojimai, įtakojantys projektavimo šablono atsiradimą ir veikimą;
- *Sprendimas* (angl. *solution*) – detalizuoti problemos sprendimo žingsniai;
- *Veikiančių jėgų pokyčiai* (angl. *force resolution*) – pasikeitusių aplinkybių / kriterijų / apribojimų kontekste ar nagrinėtuose esybėse aprašas;
- *Trumpas tikslingumo prasmės išaiškinimas* (angl. *design rationale*) – glaustai pateikiama projektavimo šablone nusakytų pokyčių esmė;

Šis formatas nėra tiksliai nusakytas, todėl punktų pavadinimai ir jų kiekis dažnai būna skirtingas. Punktai dažniausiai būna trumpi (po keletą paragrafų), nors galima aprašyti ir žemo lygio programavimo šablonus. Formatas – gana kompaktiškas, paprastai užima keletą puslapių.

2.6.3.4. Portland formatas

Šis formatas yra visiškai nestruktūrizuotas, neturi punktų. Aplinkybės, problema ir sprendimas perteikiami paprastais paragrafais, sklandžia kalba (pavyzdys - [PLT06]). Turinys atitinka *Alexander*, tačiau dar mažiau reglamentuotas, todėl tinka įvairių abstrakcijos lygių ir dalykinių sričių projektavimo šablonams aprašyti.

2.6.3.5. POSA formatas

Formatas, kurio pavadinimą sudaro žinomos projektavimo šablonų knygos [Bus+96] pavadinimo akronimas („*Pattern-Oriented Software Architecture"; POSA*), yra dar labiau struktūrizuotas nei GoF:

- *Apibendrinimas* (angl. *summary*) – trumpas šablono esmės nusakymas;
- *Pavyzdys* (angl. *example*) – sprendžiamos problemos pavyzdys;
- *Kontekstas* (angl. *context*) – probleminės zonos konteksto aprašas;
- *Problema* (angl. *problem*) – sprendžiamos problemos formuluotė;
- *Sprendimas* (angl. *solution*) – projektavimo šablono siūlomi pokyčiai problemos aplinkoje;
- *Struktūra* (angl. *structure*) – sprendime dalyvaujančių esybių aprašai, klasių / objektų diagramos;
- *Dinamika* (angl. *dynamics*) – aprašomas projektavimo šablonu nusakyto sprendimo dinaminis elgesys: sąveikų su konteksto bei vidinėmis esybėmis chronologija;
- *Realizacija* (angl. *implementation*) – punktais (žingsniais) suskirstytos instrukcijos problemos sprendimui;
- *Išspręstas pavyzdys* (angl. *example resolved*) – probleminės situacijos, pavaizduotos 2 projektavimo šablono punkte, išspręsta versija;
- *Variantai* (angl. *variants*) – aptariamos alternatyvios projektavimo šablono versijos;
- *Žinomi pritaikymo atvejai* (angl. *known uses*) - aprašomi keli gerai žinomi nagrinėjamo projektavimo šablono taikymo pavyzdžiai;
- *Pasekmės* (angl. *consequences*) – teigiami ir neigiami efektai nagrinėto probleminio objekto atžvilgiu;
- *Kiti* (angl. *see also*) – alternatyvūs, apgaubiantys ar naudojami projektavimo šablonai nagrinėtojo atžvilgiu;

Šis formatas dažniausiai naudojamas detaliems žemo lygio programavimo šablonams aprašyti, todėl yra išsamiausias ir paprastai užima labai daug puslapių;

2.6.3.6. Lauder & Kent formatas

Šie autoriai savo straipsnyje [Lau98] siūlo gana žymų kokybinį projektavimo šablonų atvaizdavimo paradigmos postūmį. Pripažįstant, jog problema, kontekstas ir jėgos yra būtinos projektavimo šablono aprašui ir grafine notacija tai aprašyti nepraktiška, iškeliama mintis, kad pačią projektavimo šablono struktūrą būtina suskaidyti taip išvengiant užteršimo menkaverčiais praktiniais pavyzdžiais.

GoF / POSA stiliaus vieningą atvaizdavimą klasių lygyje autoriai pataria abstrahuoti iki trijų lygių: *rolių* („atkurti šablono dvasią be menkaverčių detalių“), *tipų* („pridėti dalykinės srities apribojimus“) ir tik tuomet - *klasių* („atvaizduoti konkretų taikomąjį atvejį“).

Taip pat pertvarkomas dinamikos / sąveikos su aplinka aprašas, kurį sudaro sąveikos diagrama lygiagrečiai su pradinės / galinės būsenos atvaizdu, tiksliai pavaizduojant abiejų dalių sutapimo taškus.

Visa tai yra pateikiama vieninga grafine notacija, kuri yra vaizdi, bet nereikalauja matematinio formalizmo (pvz, Z-notacijos). Pasiekiamas toks sluoksniuotas abstraktumo lygis, kai projektuotojas gali pritaikyti šabloną norimu detalumu, tačiau nėra gluminamas primityvių kardinalumu ir pavadinimų sistema apribotų gyvenimiškų pavyzdžių („turime metodą A ir metodą B...“).

2.6.3.7. Formatų tarpusavio palyginimas

Dėl griežtų standartų nebuvimo bei variacijų netgi tų pačių autorių kontekstuose, tiksliai palyginti projektavimo šablonų formatus yra problematiška. Lentelėje 1 pateikiamas empirinis palyginimas pagal kriterijus (sąlyginėje didėjančioje skalėje nuo 1 iki 10):

- *Apimtis* – tipinė projektavimo šablono užimama vieta (su diagramomis);
- *Struktūrizuotumas* – suskaidymo į punktus ir fiksuotos paskirties / pozicijos zonas lygis;
- *Vaizdumas* – grafinės notacijos panaudojimo lygis;
- *Universalumas* – galimų atvaizduoti probleminių sričių spektro plotis;

1 lentelė. Projektavimo šablonų formatų empirinis palyginimas

Metodas / kriterijus	**Apimtis**	**Struktūrizuotumas**	**Vaizdumas**	**Universalumas**
Alexander (žr. 2.6.3.1)	7	3	3	8
GoF (žr. 2.6.3.2)	7	6	6	4
Coplien (žr. 2.6.3.3)	3	6	4	8
Portland (žr. 2.6.3.4)	9	2	1	9
POSA (žr. 2.6.3.5)	10	9	5	3
Lauder & Kent (žr. 2.6.3.6)	8	5	9	4

2.7. Projektavimo šablonų kalbos detalizacija

„Projektavimo šablonai pasižymi struktūra, nusakančia, kaip kiekvienas projektavimo šablonas yra sudarytas iš kitų, mažesnių projektavimo šablonų. Projektavimo šablonuose taip pat įterptos taisyklės, nusakančios jų atkartojimo kelią – įskaitant kitų projektavimo šablonų panaudojimą“ – taip Christopher Alexander apibūdina [Ale+79] (projektavimo) šablonų kalbą / sistemą (angl. *(design) pattern language / pattern system*) - kokybiškai aukštesnio lygio metaforą, pakeliančią projektavimo šablonus į didesnio panaudojamumo lygį.

Terminas „kalba“ naudojamas neatsitiktinai, nes, laikant projektavimo šablonus kalbos elementais, juos taip pat galime laikyti ir tos kalbos gramatika – bet kurios kalbos gramatika gali būti aprašyta tos kalbos elementais. Projektavimo šablonų įterptinis panaudojimas pagal jų pačių nusakomas „gramatikos taisykles“ – vienas esminių projektavimo kalbos bruožų, suteikiančių jai naudingumo.

Alternatyvus terminas – „sistema“ (siūlomas kai kurių teoretikų) – atsižvelgia į faktą, kad sunku apimti visą aprašomą dalykinę sritį, terminas „kalba“ laikomas pertekliniu. Be to, nepateisinamas klasikinis „kalba“ apibrėžimas.

2.7.1. Sudėtis

Projektavimo šablonų kalbos yra sudaromos iš generuojančių (žr. 2.6.1 punktą) projektavimo šablonų, todėl gali būti naudojamos naujų sistemų kūrimui naudojant laiko patikrintus ir rekomenduojamus principus bei metodus.

Kalbos elementų sąsajos gali būti įvairios, ne tik nuosavybinė (angl. *part-of*). Kartais kalbų kūrėjai pritaiko įvairių objektinių metodologijų ir UML ryšių tipus.

Kuriamos kalbos elementai (projektavimo šablonai, sąsajos) paprastai apibrėžiami žodynuose, siekiant paprastumo ir taisyklingesnės struktūros

2.7.2. Savybės

Esminis projektavimo šablonų sistemos bruožas – hierarchinės projektavimo šablonų sąsajos, sukuriančios projektavimo šablonų medį, kuriame projektavimo šablonų mastas ir svarba mažėja, sąsajomis tolstant nuo šaknies.

Pasirinkus bet kurį projektavimo šabloną, jam įgyvendinti galima rinktis kurį nors iš susijusių žemesnio lygio projektavimo šablonų, ir t.t. Taip paskirstomas ir diskretizuojamas ekspertinių žinių bagažas, pasiekiama didžiausia kuriamos sistemos variantų aprėptis.

Dažniausiai minėti bruožai nėra ryškūs; projektavimo šablonų kalba tarnauja kaip patogus ir primityvus projektavimo šablonų integratorius.

2.7.3. Klausimynų projektavimo šablonų formatų parinkimas

Kadangi klausimynai – didelė ir įvairiapusiška probleminė sritis, skirtingų tipų sąvokoms aprašyti vieningo projektavimo šablonų formato gali nepakakti.

Galima išskirti tris projektavimo šablonų tipus, naudotinus šiame darbe (apibrėžti 2.6.2):

- Psichologiniai – talpina žinias, susijusias su psichologiniais klausimynų aspektais, pvz., klausimų formuluotėmis;
- Realizacijos – apima Programinius ir Architektūrinius; nusako pavyzdinę norimų klausimyno aspektų realizaciją;
- Vartotojo sąsajos – aprašo rekomenduotinus klausimynų vartotojo sąsajos elementus ir jų konfigūracijas;

Akivaizdu, kad skirtingų tipų projektavimo šablonai kalboje bus apjungiami skirtingos semantikos sąryšiais.

Taip pat būtina skirtingų tipų projektavimo šablonams naudoti nevienodą aprašo struktūrą, kadangi bereikalingas unifikavimo vaikymasis gali sumažinti lankstumą. Pasirinkta struktūra:

- *Psichologiniams* – pasakojamoji šių projektavimo šablonų prigimtis bei žemo lygio sąvokų nebuvimas sąlygoja elementariausio *Portland* formato naudojimą: neišskiriami atskiri punktai, *Alexander* stiliaus punktai pateikiami neišreikštine forma atskirais paragrafais;
- *Realizacijos* – naudojamas *Coplien* formatas, duomenų struktūras iliustruojant *UML* klasių diagramomis su specialiais simboliais pasikartojančioms esybėms žymėti;
- *Vartotojo sąsajos* – padidintam abstrakcijos lygiui pasiekti pasitelkiamas *Coplien* formatas, išplečiant atitinkamus punktus koncepcinėmis diagramomis;

Vienodumo padidinimui verta apsibrėžti grafinės notacijos elementų bendrą žodyną kalbos kontekste.

2.8. Darbo tikslas

Apibendrinti klausimynų IS kūrimo ir valdymo patirtį projektavimo šablonų pavidalu, apjungiant atskirus atvejus į hierarchines grupes pagal semantinius kriterijus, šitaip suformuojant lengvai įsisavinamą projektavimo šablonų kalbą, tinkamą klausimynų projektuotojų selektyviam apmokymui.

Darbo rezultato tinkamumą iliustruoti prototipinėmis IS, naudojančiomis bent dalį aprašytų projektavimo šablonų.

2.9. Rizikos faktorių analizė

Kūrimo metu neišvengiama tam tikra rizika, tikimybė, jog nepavyks pasiekti pageidaujamo rezultato. Pagrindiniai šiame darbe galiojantys rizikos faktoriai:

- *Laiko trūkumas* – egzistuoja galimybė nepakankamai įvertinti probleminės srities sudėtingumą ir apimtį projektui paskirto laiko kontekste, todėl užbaigti projektavimo šablonai, kaip visuma, gali pasirodyti nepakankamai išsamūs. Šis faktorius gali būti eliminuojamas supaprastinant užduotį arba padidinant projektavimo šablonų granuliaciją;

- *Pernelyg sudėtinga užduotis* – probleminė sritis gali pasirodyti paprasčiausiai neaprašoma projektavimo šablonų kalba. Galimas sprendimas – aprašyti probleminės srities poaibį;
- *Žinių trūkumas* – gali neužtekti probleminės srities išmanymo kryptingam ir prasmingam gerą dizainą skatinančių projektavimo šablonų aprašymui. Tokia problema šalintina klausimynų kūrimo paketų analize ir literatūros studijavimu;

2.10. Klausimynų projektavimo šablonų kalbos alternatyvos

Projektavimo šablonų kalbų kūrimo sferoje nėra standartų ar griežtų normų; formatų ir terminų interpretacijos gana laisvos. Pačiame abstrakčiausiame lygyje kūrimo motyvaciją galima skirstyti į du tipus:

- *Originalių idėjų pristatymą* plačiajai publikai būtent projektavimo šablonų kalbos forma, siekiant struktūrizuoti žinias ir palengvinti jų pasiekiamumą;
- Jau egzistuojančių *žinių formalizaciją / prezentaciją* nauja – projektavimo šablonų kalbos, - forma. Kalba tarnauja kaip adapteris, standartizuojantis priėjimą prie žinių.

Projektavimo šablonų kalbos yra atviros, išplečiamos dėl savo moduliškumo, todėl dvi kalbos, apimančios vienodas ekspertines žinias – labai netradicinis variantas, netgi egzistuojančių žinių formalizavimo atveju. Kiti autoriai visuomet gali praplėsti / pataisyti jau egzistuojančios projektavimo šablonų kalbos turinį.

Klausimynų projektavimo šablonų kalba iki šiol nebuvo sukurta (dalykinių sričių, tinkančių formalizavimui aptariama forma, yra labai daug, o laiko nuo projektavimo šablonų kalbos išradimo (1977) praėjo sąlyginai nedaug.). Savaime suprantama, neigiamą faktą (alternatyvų nebuvimą) įrodyti sunku (pagal monotoninės logikos postulatus), tačiau projektavimo šablonų kalbų konferencijose ir interneto paieškoje nieko rasti nepavyko.

Egzistuoja universalūs kriterijai, pagal kuriuos galima lyginti kuriamą kalbą su kitomis, aprašančiomis visai nepanašias dalykines sritis:

- *Projektavimo šablonų bei pačios kalbos vaizdumas* – būtina naudoti pakankamai grafinių konstruktų, kad projektavimo šablonų naudotojams nereikėtų sudėtingų hierarchinių / tinklinių konstruktų įsivaizduoti patiems;
- *Turinio inovatyvumas* – originalių ekspertinių žinių laipsnis projektavimo šablonų kalboje. Pavyzdžiui, vadovėlio perteikimas projektavimo šablonų kalbos forma pagal ši kriterijų gautų žemą įvertį;

- *Kalbos struktūros sudėtingumas* – projektavimo šablonų formatų, ryšių tarp projektavimo šablonų tipų įvairovė, apibendrinus – kalbos išraiškos turtingumas.

2.11. Kokybės kriterijai

Sukurtos projektavimo šablonų kalbos klausimynams vietiniai (netaikomi kitoms projektavimo šablonų kalboms) kokybės kriterijai kokybiškai skiriasi nuo 2.10 aprašytų universalių palyginamųjų kriterijų tuo, kad koncentruojasi į patį darbo rezultatą ir įrodymui nereikalauja lyginimo su alternatyvomis. Pasirinkti kriterijai:

- *Dalykinės srities (arba pasirinkto jos poaibio) realizavimo programinėje įrangoje ekspertinių žinių pilnumas* – projektavimo šablonų kalba turi suteikti projektuotojui pakankamai žinių, jog jis galėtų laiko ir patirties patikrintais principais kryptingai ir neklystamai kurti klausimynus realizuojančią programinę įrangą. Naudingumas kitų profesinių sričių specialistams nėra šio darbo esmė;
- *Vertybinė koncentracija* – projektavimo šablonų kalba neturi apimti temų, nesusijusių su klausimynais, tokių kaip: atsakymynai, mokomieji / ekspertiniai modeliai;
- *Pagrįstumas* - bent pusė programinių / architektūrinių projektavimo šablonų (šio darbo apimtyje) turi būti realizuoti prototipinėse IS kaip pavyzdžiai;
- *Pradinių reikalavimų optimalumas* – projektavimo kalboje pateikiamos klausimynų dalykinės srities ekspertinės žinios neturi reikalauti didesnio IS kūrimo supratimo nei suteikiama aukštojoje mokykloje mokomiems specialistams (procedūrinis ir objektinis programavimas, grafinės vartotojo sąsajos sąvokos, programų architektūra, UML, metaduomenys).

2.12. Kūrimo priemonės

Vizualinės projektavimo šablonų dalys bus rengiamos *Microsoft Visio 2003 Enterprise Architect*, išplečiamu *CASE* įrankiu, palaikančiu *UML* ir įgalinančiu sukurti ir išsaugoti pakartotiniam panaudojimui naujus diagramų grafinius elementus, kurie bus naudojami projektavimo šablonams aprašymui.

2.13. Analizės išvados

Nustatyta, kad norimą tikslą – klausimynų projektuotojo ekspertinių žinių pateikimą, - lengviausia pasiekti sukuriant projektavimo šablonų kalbą, naudosiančią skirtingus projektavimo šablonų formatus atskiriems aprašomiems probleminės srities profiliams nusakyti.

Projektavimo šablonų kalba apims įvairius probleminės srities aspektus, bet naudos vieningą duomenų žodyną.

Darbo teisingumui įrodyti bus atliktas eksperimentas – prototipinių klausimynų IS kūrimas remiantis darbe sukauptomis žiniomis.

3. Siūlomos projektavimo šablonų kalbos sudėtis

Bet kuri projektavimo šablonų kalba yra sudaryta iš tam tikrų struktūrinių elementų (gramatinė analogija: veiksmažodis, daiktavardis...) bei jų realizacijos konkrečios dalykinės srities kontekste (gramatinė analogija: „bėgti“, „žmogus“...). Šiuo atveju dėl įdiegtų inovacijų struktūrinė komponentė (žr. 3.1) yra didesnė ir įvairesnė nei įprasta projektavimo šablonų kalbose (kurios tėra primityviai susieti homogeniški projektavimo šablonai). Esminio projektavimo kalbos elemento – projektavimo šablono, - realizacijos pateikiamos 3.2.3 punkte.

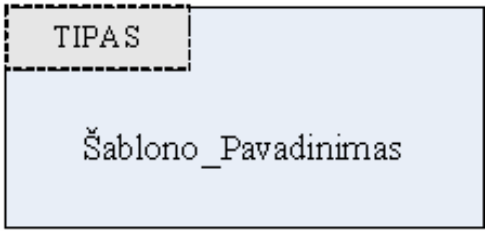
3.1. Architektūra

Projektavimo šablonų kalbų kontekste priimta, kad atskiri projektavimo šablonai ir jų tarpusavio santykiai vaizduojami grafinėje schemoje, pagreitinančioje supratimą ir palengvinančioje navigaciją. Praktikoje dažnai projektavimo šablonų santykiai apsiriboja generalizacija / detalizacija, sudėtingesnių ryšio savybių aprašus perkeliant į pačių projektavimo šablonų dalis „Kontekstas“, „Pasekmės“, „Veikiančių jėgų pokyčiai“.

Ši konkreti kalba grafiniame vaizde sutelkia didesnį nei įprastai semantikos kiekį (laikoma, kad klasikiniai ryšio tipai – nepateisinamai supaprastinti).

3.1.1. Sudėtiniai elementai

Pagrindinis projektavimo šablonų kalbos elementas – projektavimo šablonas, - navigacijos schemoje (žr. 3.2.1) vaizduojamas (2 pav.) stačiakampiu su pavadinimu ir sutrumpintu projektavimo šablono tipo paminėjimu (2 lent.).



2 pav. Projektavimo šablono pavyzdys

Šioje projektavimo šablonų kalboje vienas šablonas yra laikomas baziniu (žr. 3.1.1); nuo jo pradedamas ekspertinių žinių įsisavinimas. Grafiškai navigacijos schemoje toks šablonas išskiriamas pastorintu išoriniu rėmeliu.

2 lentelė. Trumpi aprašomų projektavimo šablonų apibūdinimai

Projektavimo šablono tipas (žr. 2.6.2)	**Schemoje naudojamas kodas**
Programiniai / architektūriniai	PRG
Grafinės vartotojo sąsajos	GVS
Psichologiniai / turinio	PSI

Projektavimo šablonų santykiai (3 pav.) išreiškiami įvairiais binariniais ryšiais (3 lent.), kurie atvaizduojami rodyklėmis. Ryšys ir jo siejami projektavimo šablonai sudaro ontologinį tripletą bei gramatinį derinį, skaitomą kaip „subjektas veiksmas objektas“.

3 pav. Šablonų santykio dalyviai ir kryptis

3 lentelė. Ryšių tarp projektavimo šablonų tipų aprašai

Pavadinimas	**Aprašymas**
<IŠPLEČIA>	Nurodo projektavimo šabloną – objektą, kurio sprendimas papildo kuriamą IS (apskritai) ir tėvinį projektavimo šabloną – subjektą (konkrečiai) naujais struktūriniais komponentais
<VIZUALIZUOJA>	Pateikia nuorodą į grafinės vartotojo sąsajos (GVS) tipo projektavimo šabloną – objektą, sprendime pateikiantį tėviniame projektavimo šablone – subjekte aprašytos programinės abstrakcijos pateikimo vartotojui grafine forma schemą
<GERINA>	Nusako projektavimo šabloną – objektą, kuriame pateikiamas sprendimas tam tikru būdu patobulina / pagerina projektavimo šablone – subjekte siūlomą atvejį
<PAKEIČIA>	Tai – ekvivalentiškumo nuoroda tarp dviejų projektavimo šablonų. Subjekto – objekto parinkimas nesvarbus, nes nuoroda – dvikryptė. Dažniausiai projektavimo šablonai, poromis sujungti šio tipo

Pavadinimas	Aprašymas
	ryšiais, būna unikaliai parinktini (angl. *mutually exclusive*) – yra alternatyvos

Labai svarbus projektavimo kalbos bruožas – pritaikomumas (angl. *scalability*), įgalinantis kalbos vartotoją įsisavinti kalboje pateiktas ekspertines žinias norimu detalizacijos lygiu. Pasitelkiant analogiją: *JPEG*/*wavelet* paveikslėliai internetu atsiunčiami palaipsniui, didinant paveikslėlio ryškumą, detalių kiekį. Projektavimo šablonų kalbos vartotojas taip pat skaito ir gilinasi į norimus projektavimo šablonus, eidamas ryšiais tarp jų norimu keliu grafe. Jis gali pasirinkti dominančią kalbos dalį ir ją išnagrinėti iki pakankamo detalumo lygio.

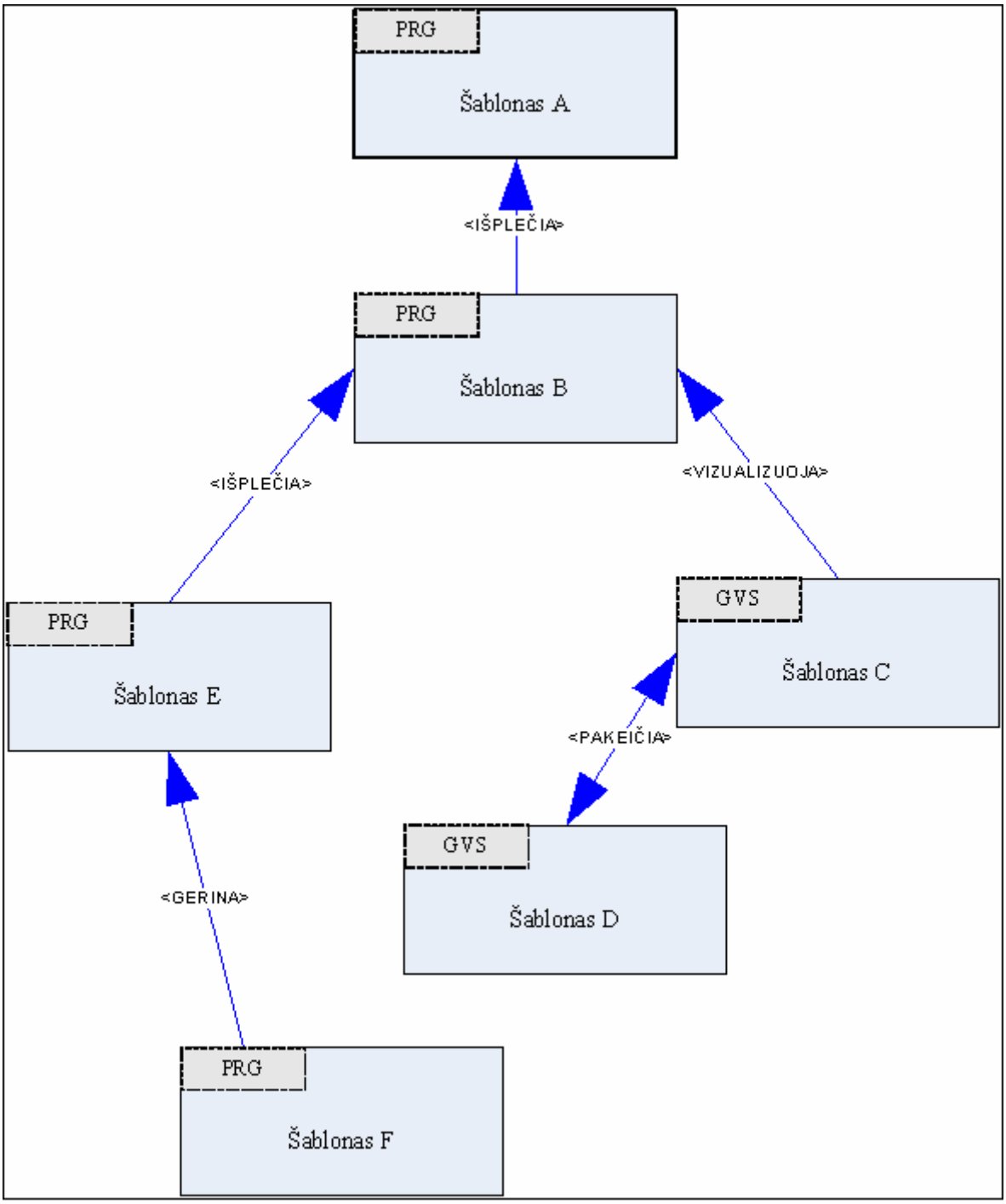
Klasikinėje projektavimo šablonų kalbos koncepcijoje [Ale+77] bet kuris projektavimo šablonas laikomas realizuotu praktikoje tik jei atitinkamai realizuoti visi „detalesni“ projektavimo šablonai, nuosavybės ryšiais sujungti su aptariamuoju. Šis bruožas įpareigoja kalbos vartotoją taikyti „viskas arba nieko“ požiūrį ir riboja mokymosi lankstumą. Šioje projektavimo šablonų kalboje ryšiai (3 lent.) nėra įpareigojantys; kiekvienas projektavimo šablonas yra pakankamai savarankiškas, kad jo idėją būtų galima pritaikyti individualiai, nors, žinoma, susijusių projektavimo šablonų realizavimas sprendimą vienaip ar kitaip pagerina.

Naudojimo dinamiką padeda įsivaizduoti 4 pav. pateikiamas pavyzdys, kurio šablonų turinys toks:

- A – pačiais bendriausiais bruožais aprašo projektavimo šablonų kalboje pateikiamą viziją ir sprendimą. Nepateikia jokių detalių, tik aukšto lygio architektūrinius artefaktus;
- B – detalizuoja A, įvesdamas konkrečią duomenų struktūrų architektūrą ar veikimo algoritmo aprašą;
- C – pateikia projektavimo šablone B išreikšto modelio vizualiąją pusę – pateikimą vartotojui, langų brėžinius, kaskadas. Pavyzdžiui – grafo duomenų struktūrų (aprašytų projektavimo šablone B) pateikimą taškais ir kryptinėmis rodyklėmis
- D – pateikia projektavimo šablone B išreikšto modelio vizualiąją pusę – tačiau skirtingu būdu, nei projektavimo šablonas C. Galima sakyti, kad D yra C alternatyva; jos abi – nesuderinamos tarpusavyje;
- E – įveda papildomą sistemos savybę (angl. *feature*), kuri nėra kritiškai būtina veikimui, bet suteikia naujų galimybių. Tai galėtų būti, pvz., failų eksportavimo į *Microsoft Word* formatą modulis teksto doroklėje;

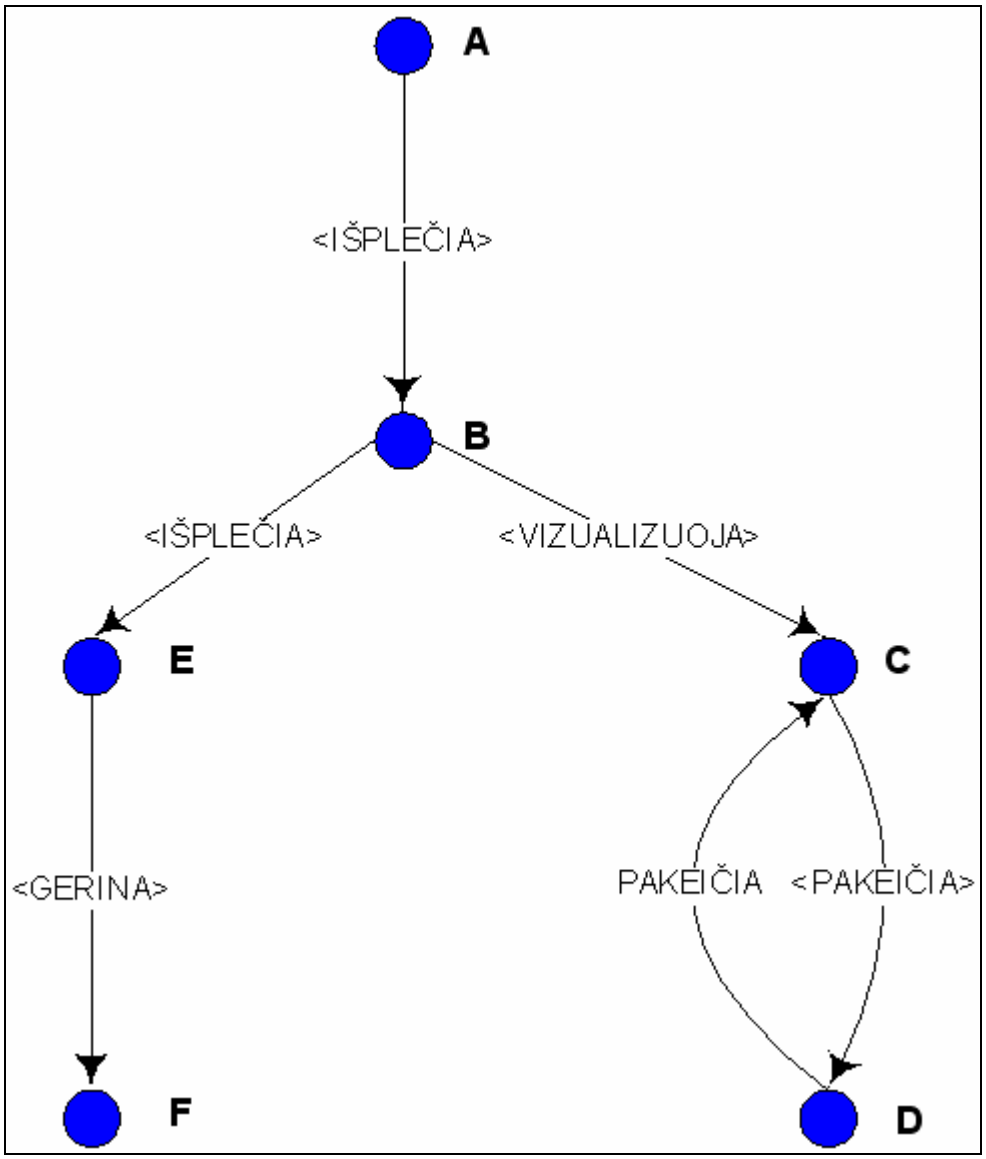
- F – nurodo sistemos savybės optimizaciją pagal tam tikrus parametrus. Savaime suprantama, šis projektavimo šablonas sistemai nesuteikia nieko struktūriškai / kokybiškai naujo, bet vistiek apima vertingų ekspertinių žinių;

Siūloma projektavimo šablonų kalba pradedama konstruoti nuo bazinio projektavimo šablono, nuosekliai prijungiant kitus projektavimo šablonus semantiškai tinkančiais ryšiais ekspertinės vizijos detalumo didėjimo kryptimi. Pavyzdžiui – projektavimo šablonai B ir A apjungiami išplėtimo ryšiu, C ir B – vizualizacijos, F ir E – gerinimo.



4 pav. Projektavimo šablonų kalbos pavyzdys

Pateiktos projektavimo šablonų kalbos vartotojas gali joje sukauptas žinias pasiekti daugeliu kelių. Kelias visuomet pradedamas nuo bazinio projektavimo šablono (A) ir konstruojamas, projektavimo šablonų kalbos schemą laikant orientuotu grafu (5 pav.), kuriame viršūnes atstoja projektavimo šablonai, o jungtis – ryšiai tarp projektavimo šablonų (ryšių kryptis dėl navigacijos schemos – grafo nesuderinamumo reikia sukeisti).



5 pav. Projektavimo šablonų kalbos pavyzdžio naudojimo grafas

Visi įmanomi projektavimo šablonų kalbos naudojimo variantai randami pereinant grafą paieška į gylį (angl. *depth-first search*) su tam tikromis išlygomis (prisimenant nueitą kelią, atsižvelgiant į <PAKEIČIA> ryšį):

A

A – B

A – B – C

A – B – D

A – B – C – E

A – B – D – E

A – B – C – E – F

A – B – D – E – F

A – B – E

A – B – E – F

Savaime suprantama, vartotojas gali pasirinkti ne vieną kelią ir paeiliui plėtoti klausimynų IS keliomis detalumo didėjimo kryptimis.

3.1.2. Projektavimo šablonų struktūra

Projektavimo šablonų aprašai, priklausomai nuo paskirties, gali būti pateikiami įvairiomis formomis (žr. 2.6.3), iš kurių nė viena nėra griežtai standartinė. Patartina pasirinktą formatą prisitaikyti savo individualiems poreikiams; aptariamoje projektavimo šablonų kalboje projektavimo šablonams, priklausomai nuo paskirties, pritaikomi ir atskiri formatai:

- Programiniams / architektūriniams bei grafinės vartotojo sąsajos projektavimo šablonams patiekti naudojamas *Coplien* (žr. 2.6.3.3) formatas, modifikuotas *GoF* (žr. 2.6.3.2) elementais. Projektavimo šablono aprašą sudaro šie punktai:
  - *Problema* – trumpai nusakyta probleminė situacija;
  - *Kontekstas* – klausimynų IS dalis ir / ar kūrimo proceso stadija, kur atsitinka minima problema;
  - *Sprendimas* – nurodytos problemos sprendimas;
  - *Pasekmės* – iš *GoF* paveldėtas punktas, kuriame nusakomos teigiamos ir / ar neigiamos projektavimo šablono pritaikymo pasekmės;
- Psichologiniai / turinio projektavimo šablonai dėl savo amorfinės prigimties ir sunkaus formalizavimo pateikiami *Portland* (žr. 2.6.3.4) formatu – t.y. paprastu tekstu, bendrai prisilaikant aukščiau pateiktų punktų (pvz., problemą aprašyti pageidautina prieš sprendimą; pasekmės neprivalomos).

Paprastai projektavimo šablonai, kurti ne kaip kompleksiškos projektavimo šablonų kalbos dalis, turi punktus, nurodančius sąsajas su kitais šablonais (taip daro, pvz., *GoF* [Gam+95]): „Susiję projektavimo šablonai“, „Veikiančių jėgų pokyčiai“, „Kiti“. Šioje

projektavimo šablonų kalboje naudojami išsamūs ryšių klasifikatoriai sumažina tokį poreikį iki minimumo, todėl atitinkami punktai panaikinami iš projektavimo šablonų formatų.

Tradiciškai nesusieti projektavimo šablonai turi punktą, skirtą taikymo pavyzdžiams, iliustruojantiems projektavimo šablono esmę ir sprendimo įtaką. Šiame darbe svarbiausių nepsichologinių projektavimo šablonų pavyzdžiai pateikti eksperimento dalyje sulyginant su pavyzdinių klausimynų IS realizacijomis (žr. 4.2), bet ne tiesiogiai, siekiant išvengti pasikartojimo bei fragmentiškumo.

Egzistuoja kontraversija dėl projektavimo šablonų pavadinimų – nėra jokių formalių standartų, pavadinimų tinkamumas turiniui / esmei yra labai subjektyvus ir priklauso nuo skaitytojo. Siūlomoje projektavimo šablonų kalboje pavadinimai suteikiami remiantis šiais kriterijais:

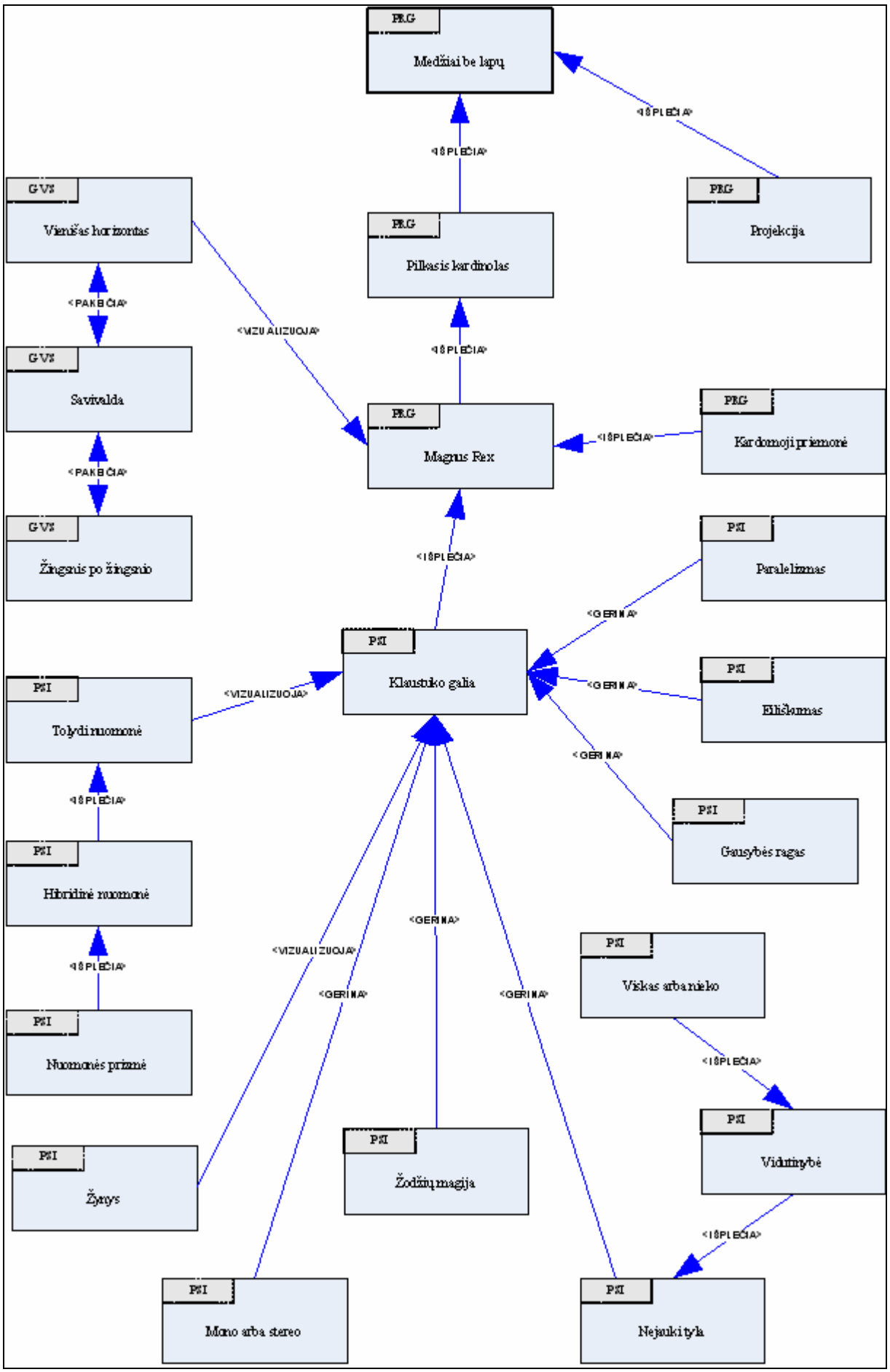
- Pavadinimas turi būti daiktavardis ar frazė / posakis, įgalinantys minėti projektavimo šabloną sakiniuose objekto (veiksnio) rolėje;
- Labai svarbu pavadinimų vaizdumas, kadangi nusivalkiojusius, užterštus pašalinėmis asociacijomis žodžius sunku prisiminti projektavimo šablonų kalbos kontekste;
- Pavadinimas neprivalo atskleisti projektavimo šablono esmę a priori, tačiau vartotojui, žinančiam konkretaus šablono turinį, išgirstas pavadinimas turi iškart sukelti tiesioginę asociaciją. Kitaip tariant – žinant projektavimo šablono turinį, turi būti nesunku prisiminti ir pavadinimą.

3.2. Projektavimo šablonų kalbos turinys

Nusakytos struktūros (žr. 3.1) projektavimo šablonų kalbos turinį sudaro įvairūs elementų egzemplioriai, apjungti apibrėžtais ryšiais. Galima sakyti, jog turinį sudaro projektavimo šablonų egzemplioriai (žr. 3.2.3) bei jų santykius nusakanti navigacijos schema (žr. 3.2.1). Projektavimo šablonų susietumo sekimas (angl. *cross-referencing*) palengvinamas naudojant konceptų žodyną (žr. 3.2.2).

3.2.1. Navigacijos schema

Siūlomos projektavimo šablonų kalbos struktūra ir navigacijos informacija yra pateikiamos navigacijos schemoje (6 pav.). Bazinis projektavimo šablonas – „Medžiai be lapų“ (žr. 3.2.3.1).



6 pav. Navigacijos schema

Projektavimo šablonų kalbos navigacinę schemą suprasti ir norimą vykdymo kelią (žr. 3.1.1) pasirinkti padeda trumpi projektavimo šablonų aprašymai (4 lentelė).

4 lentelė. Trumpi aprašomų projektavimo šablonų apibūdinimai

Šablono pavadinimas	**Problema**	**Sprendimas**
Medžiai be lapų (žr. 3.2.3.1)	Klausimynų architektūra dažnai pasirodo pernelyg ribota ir neapgalvota, apsunkinanti išplėtimą ir tobulinimus	Naudoti MVC pagrįstą architektūrą su griežtai atskirtomis dalimis ir DOM pagrįstomis duomenų struktūromis
Pilkasis kardinolas (žr. 3.2.3.2)	Plačiai naudojamos monolitinės duomenų struktūros	Suskaidyti duomenų struktūras pagal vartojamumą ir keičiamumą
Magnus Rex (žr. 3.2.3.3)	Klausimynas paprastai suteikia menką palaikymą sudėtingiems ryšiams tarp klausimų, išsamiam užpildymo būklės palaikymui.	Klausimynus ir klausimus susieti lanksčia hierarchija, taikyti neriboto sudėtingumo apribojimus
Klaustuko galia (žr. 3.2.3.4)	Esminis klausimyno elementas – klausimas, - atvaizduojamas neuniversalia forma ir nepakankamu tikslumu	Pasirinkti optimalią klausimo vidinę sudėtį, apsibrėžti reikalingus atributus
Kardomoji priemonė (žr. 3.2.3.5)	Apribojimus realizuojančios struktūros paprastai būna nelanksčios ir ribotos	Išskaidyti apribojimą atominiais elementais ir apjungti lenkiškąja notacija
Projekcija (žr. 3.2.3.6)	Sunku pasirinkti universalų ir taupų resursams klausimynų egzempliorių duomenų (de)serializavimo formatą reliacinėms DBVS	Kiekvienam klausimynui skirti lentelių porą ir klausimų reikšmes saugoti pagal aktualumą
Vienišas horizontas (žr. 3.2.3.7)	Vidines duomenų struktūras būtina pateikti vartotojui kuo paprasčiau	Naudoti vientisą hierarchinį medį
Savivalda	Vidines duomenų struktūras	Realizuoti hibridinį (hierarchinį-

Šablono pavadinimas	Problema	Sprendimas
(žr. 3.2.3.8)	būtina pateikti vartotojui optimalia prieinamumo ir apimties kriterijais forma	plokščią) medžio pateikimą
Žingsnis po žingsnio (žr. 3.2.3.9)	Vidines duomenų struktūras būtina pateikti glausta forma mažaekraniams įrenginiams	Pasitelkti fokusavimą pavienių klausimų atvaizdavimui
Žynys (žr. 3.2.3.10)	Įprastinis diskrečiojo domeno atvaizdavimas yra nepatogus ir gali įvesti paklaidų	Minimizuoti užimamą ekrano vietą ir išankstinį nusiteikimą atvaizdavimą pakeičiant optimizuotu valdikliu (angl. *widget*)
Žodžių magija (žr. 3.2.3.11)	Klausimo formuluotė gali stipriai įtakoti rezultatų tikslumą	Minimizuoti paklaidas tiksliai aprašant klausimą
Gausybės ragas (žr. 3.2.3.12)	Pernelyg trumpas galimų atsakymo variantų sąrašas kenkia tikslumui	Naudoti ilgą sąrašą, taikant optimizacijas vartotojo nuovargiui sumažinti ir tikslumui padidinti
Tolydi nuomonė (žr. 3.2.3.13)	Mažas galimų atsakymo variantų kiekis sukelia apvalinimo paklaidą, „kompromisą“	Naudoti tolydinę ar didelę diskrečiąją variantų aibę
Hibridinė nuomonė (žr. 3.2.3.14)	Tolydinė galimų atsakymo variantų aibė nevaizdi ir / arba pernelyg ribota	Naudoti diskrečiąją aibę, atsakymo stiprį nusakant pagalbine tolydine
Nuomonės prizmė (žr. 3.2.3.15)	Kai kurie klausimai turi nevienareikšmius atsakymus, neišreiškiamus nei diskrečiąja, nei tolydine aibėmis	Pasitelkti daugybinius diskrečiuosius variantus, papildomus tolydiniais stiprio fiksatoriais
Nejauki tyla (žr. 3.2.3.16)	Klausimynų projektuotojo pažiūros įtakoja klausimų šališkumą	Kurti neutralias klausimų formuluotes
Mono arba stereo	Nesuvokiama vienpolių ir	Taikyti abu tipus priklausomai

Šablono pavadinimas	Problema	Sprendimas
(žr. 3.2.3.17)	dvipolių klausimų taikymo specifika	nuo konteksto: vienpolius – esant vienam objektui, dvipolius – siekiant palyginimo
Paralelizmas (žr. 3.2.3.18)	Nevienmačiai klausimai iškraipo formuluotės prasmę	Tam tikrose aplinkybėse naudoti tik vienmačius klausimus
Eiliškumas (žr. 3.2.3.19)	Klausimų tarpusavio tvarka sukelia rezultatų paklaidą	Deramai rikiuoti pagrindinius ir šalutinius klausimus; išmaišyti atsakymų variantus
Viskas arba nieko (žr. 3.2.3.20)	Socialinio priimtinumo sindromas įtakoja pasirenkamųjų klausimų perteklinį atsakymo lygį	Suteikti galimybę nepriimtinus veiksmus nusakyti netiesiogiai
Vidutinybė (žr. 3.2.3.21)	Visuomenės prievarta įtakoja atsakymų teisingumą	Klausimus formuluoti aptakiai, neagresyvia forma

3.2.2. Konceptų simbolių žodynas ir vizualizacija

Projektavimo šablonų struktūrinėse diagramose dažnai naudojamos pasikartojančios esybės – tiek kompozicijų, tiek paveldimumo kontekstuose. Vaizdumui ir struktūriškumui padidinti plačiai naudojamos esybės žymimos (7 pav.) specialiais simboliais (apibrėžiamais projektavimo šablonuose, kur esybės pirmąkart aprašomos).

7 pav. Simbolio grafinis žymėjimas

Siekiant palengvinti simbolių turinio nustatymą, jie pateikiami kartu su nuorodomis į projektavimo šablonus, kuriuose jie yra apibrėžiami (5 lent.):

5 lentelė. Konceptų simbolių ir tėvinių projektavimo šablonų indeksas

Simbolis	Tėvinis projektavimo šablonas	Tėvinis konceptas
[Klausimynas]	Pilkasis kardinolas	Klausimynas (::Klausimynas)

Simbolis	Tėvinis projektavimo šablonas	Tėvinis konceptas
[Klausimyno_egzempliorius]		Klausimyno egzempliorius (::Klausimyno_egzempliorius)
[Klausimas] [Klausimo_egzempliorius]	Klaustuko galia	Klausimyno klausimas (::Klausimas) Klausimyno klausimo egzempliorius (::Klausimo_egzempliorius)
[Apribojimas]	Kardomoji priemonė	Loginis apribojimas (::Apribojimas)

Struktūrinių diagramų laukai projektavimo šablonų tekstuose yra nurodomi atitinkamos sudėties nuorodomis:

::*Klasės_pavadinimas***:***Lauko_pavadinimas*

3.2.3. Projektavimo šablonai

3.2.3.1. „Medžiai be lapų“

- Problema

Klausimynų IS sudėtingumas ir potencialas dažnai nuvertinami; kuriami vienkartiniai, architektūriškai apriboti sprendimai. Aptinkami neigiami faktoriai:

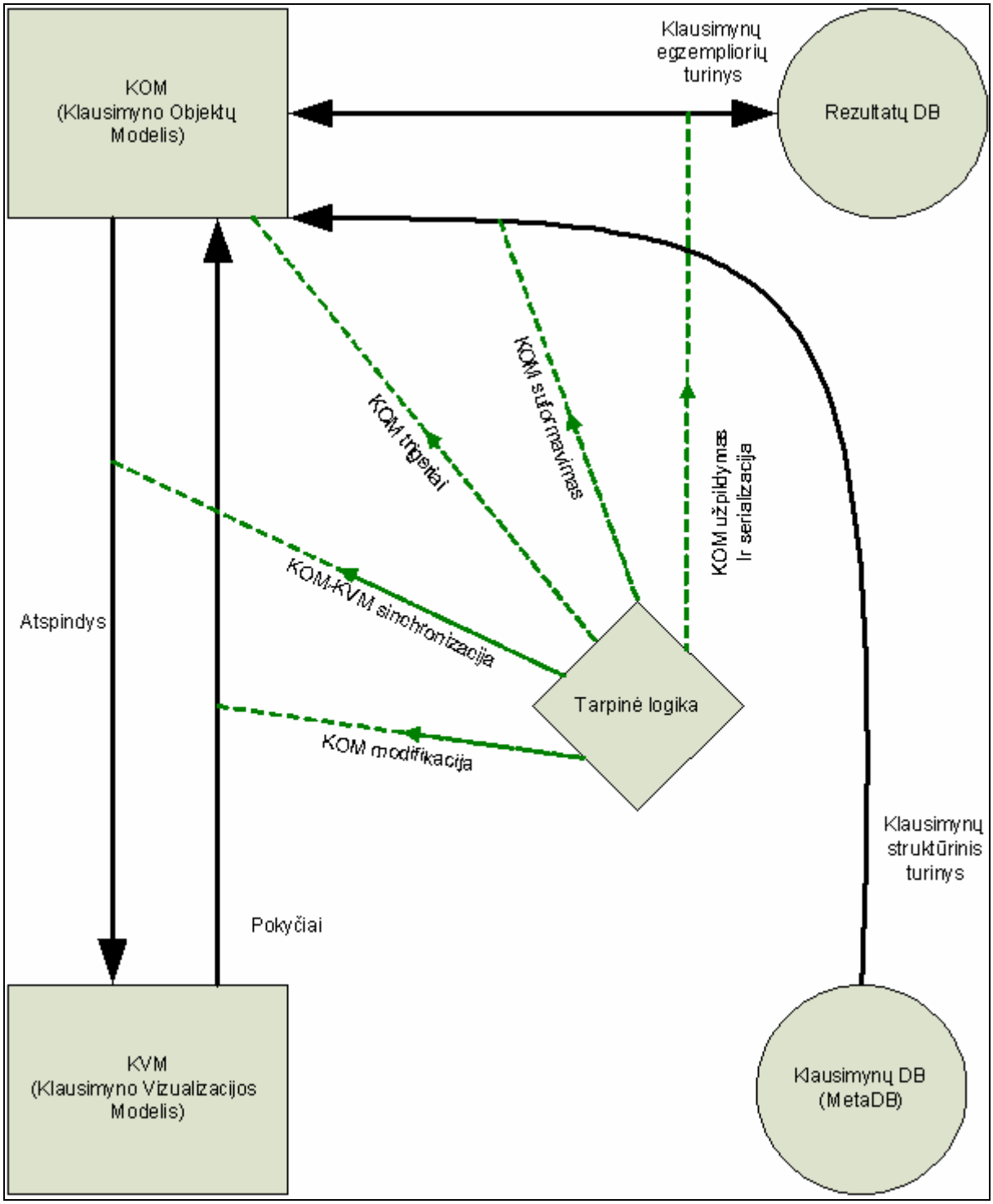
- Labai pasunkėja naujų savybių diegimas, kadangi tenka perrašyti nemažą dalį programinio kodo dėl duomenų struktūrų pokyčių;
- Klausimynų turinio (t.y. klausimų) atnaujinimai sukelia neatitikimą tarp vartotojo sąsajos išraiškos elementų ir klausimų formuluočių, ką tampa sunku ir / arba neįmanoma išspręsti be didelio masto pakartotinės inžinerijos;
- Palaikančiosios aplinkos (angl. *runtime environment*) keitimas / atnaujinimas gali įtakoti ne tik IS veikimą, sąveiką su sisteminiais servisais, bet ir vartotojo interakcijos su IS stilių. Tokiais atvejais nelanksti IS architektūra gali sąlygoti alternatyvių stilių trūkumą, taip sumažinant IS vertę naujojoje aplinkoje.

- Kontekstas

Atsiranda poreikis atspariai reikalavimų pokyčių sukeliamoms programinio kodo tobulinimo kaskadoms ir išplečiamai (angl. *extensible*) klausimynų IS.

- Sprendimas

Geras architektūrinis sprendimas gali būti pasiekiamas pasitelkiant klasikiniu MVC (*Model-View-Controller*; žr. [Bus+96]) principu pagrįstą IS komponentų atskyrimą (8 pav.).



8 pav. Siūloma bazinė klausimynų IS architektūra

Esminiai sprendimo komponentai yra:

- *Klausimyno Objektų Modelis* (KOM) - pasitelkiant žiniatinklio naršyklių patirtį, pakankamas išplečiamumas ir struktūrizuotumas gali būti pasiekiamas duomenų struktūrų šerdį projektuojant kaip HTML DOM primenantį objektų tinklą (iš čia - KOM). Tai – hierarchinė duomenų struktūra, bendrais bruožais atitinkanti teorinę klausimyno sandarą (žr. 2.1.3). KOM kontekste „objektas“ visiškai nebūtinai reiškia OOP objektą; realizuoti galima ir struktūrinio programavimo priemonėmis (žr. [Žil06]).
- *Klausimyno Vizualizacijos Modelis* (KVM) – KOM vizualus atitikmuo, suteikiantis priemones įvairiais būdais sukurti norimų KOM elementų prezentaciją IS vartotojui.
- *Rezultatų DB* (RDB) – logiškai atskira duomenų saugojimo sritis klausimynų egzempliorių (užpildytų) informacijai saugoti. Kaip taisyklė, RDB realizuojama duomenų baze reliacinėje DBVS. Universalumas reikalauja duomenis saugoti ganėtinai amorfišku formatu; interpretacija įgaunama tik apjungiant su klausimynų struktūriniais duomenimis.
- *Klausimynų DB* (KDB; metaDB) – saugo klausimynų struktūros duomenis tam tikru sutartu formatu. Klausimynų IS specifikoje struktūros duomenys klausimynų egzempliorių atžvilgiu yra metaduomenys, taigi, KDB galima vadinti metaduomenų DB arba metaDB (metaduomenų saugykla; angl. repository). KDB egzistavimas suteikia klausimynų IS daug lankstumo, padeda greičiau prisitaikyti prie reikalavimų pokyčių.
- *Tarpinė logika* – valdo informacijos mainus tarp kitų sistemos komponentų: transformuoja KOM saugomus egzempliorių duomenis į RDB formatą, RDB esančius duomenis – į atmintyje laikomą KOM apibrėžtą klausimyno egzempliorių, KDB esančius struktūrinius aprašus – į KOM tinklą, vartotojo sprendimus – į KOM pokyčius, KOM trigerių rezultatus – į numatytus KOM pokyčius, o šiuos – atgal į grafinę vartotojo sąsają per KVM.

- Pasekmės

Sudarant klausimynų IS architektūrą iš griežtai nustatytas roles užimančių komponentų, kurių susiejimas – silpnas (angl. *loose coupling*), sukuriama bazė lengvoms tolimesnėms modifikacijoms.

3.2.3.2. „Pilkasis kardinolas“

- *Problema*

Klausimynų IS vidinė būklė dažnai atvaizduojama monolitiškomis duomenų struktūromis, kurias sunku plėsti ir sudėtinga efektyviai pasiekti iš įvairių išeities teksto vietų.

- *Kontekstas*

Projektuojama klausimynų IS struktūra ir bazinė duomenų architektūra.

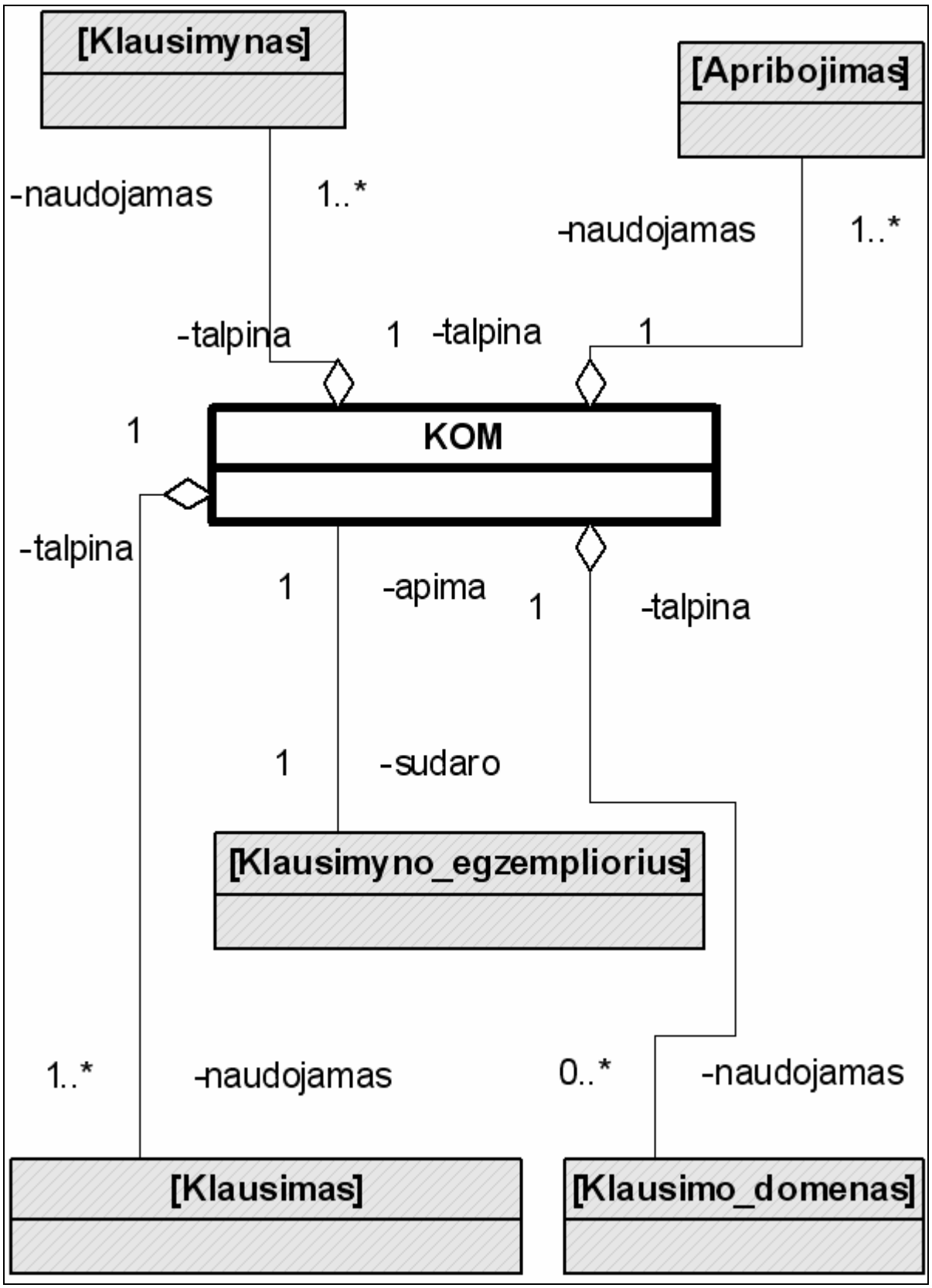
- *Sprendimas*

Svarbu suskaidyti visas duomenų struktūras į dalis pagal vartojamumą ir keičiamumą:

- *Metalygmuo* aprašo sistemines charakteristikas (pvz., „Ar galima keisti reikšmes“). Tokių metaobjektų vykdymo metu užtenka vienos kopijos, nes galima nuorodomis jį prijungti prie egzempliorių;
- *Egzempliorių lygmuo* talpina objektų / egzempliorių duomenis. Besinaudodamas metalygmens informacija, interpretatorius keičia savo elgseną objekto požiūriu.

Lanksti ir universali klausimynų IS turi naudoti metaduomenis (9 pav.); pagrindiniai iš jų tiesiškai prijungiami indeksuojant, todėl gali būti pasiekiami su minimaliu algoritminiu sudėtingumu.

Dinaminės struktūros turi nuorodas į atitinkamus metaduomenų objektus. Aukščiausio lygio dinaminis objektas – pagrindinis klausimyno egzempliorius, turintis vidines nuorodas į visus smulkesnius KOM elementus.



9 pav. KOM struktūra

- *Pasekmės*

Sumažinamas duomenų pertekliškumas, padidinamas universalumas, išlaikoma galimybė naudoti procedūrinį programavimą.

3.2.3.3. „Magnus Rex“

- *Problema*

Klausimynas paprastai suteikia menką palaikymą sudėtingiems ryšiams tarp klausimų, išsamiam užpildymo būklės palaikymui.

- *Kontekstas*

Kuriamai klausimynų IS projektuojama aukščiausio lygio duomenų struktūra – klausimynas.

- *Sprendimas*

Klausimyno duomenų struktūros (10 pav.) išskaidomos į du sluoksnius: statinį (deklaracija) ir dinaminį (egzempliorius), taip atskiriant skirtingo keičiamumo duomenų grupes:

- Statinė dalis (::Klausimynas) nusako charakteristikas, nepriklausančias nuo egzempliorių būklės ir būdingas kiekvienam egzemplioriui:
    - Pavadinimas (::Klausimynas:Pavadinimas) – vartotojui draugiškas (angl. *user-friendly*) klausimyno pavadinimas, matomas IS antraštėje vykdymo metu ir sudėtinio domeno komponentėse („Klaustuko galia“, žr. 3.2.3.4);
    - Priklausomybės faktorius (::Klausimynas:Ar_nepriklausomas) – nusako kompozicijos statusą įtraukimo į kitus klausimynus vaiko teisėmis požiūriu:
        - Nepriklausomo (angl. *standalone*) klausimyno egzempliorių gyvavimo ciklas nesusietas su kitų klausimynų egzempliorių, prie kurių jie prijungti, gyvavimo ciklu; be to – nepriklausomas

klausimyno egzempliorius gali būti prijungtas prie neriboto kiekio kitų klausimynų egzempliorių;

- Priklausomo (angl. *dependent*) klausimyno egzemplioriai sukuriami ir sunaikinami kartu su tėviniu kito klausimyno egzemplioriumi; negali būti prijungiami kituose klausimynuose, taigi, atitinka kompozicijos ryšio priklausomąją dalį;

- Dinaminė dalis (::Klausimyno_egzempliorius) paveldi statinėje dalyje nusakytas charakteristikas ir prideda savų, kintančių programos vykdymo eigoje ir reaguojančių į vartotojo elgesį:
  - Redagavimo būklė (::Klausimyno_egzempliorius:Ar_redaguojamas) nurodo, ar vartotojas gali keisti klausimyno egzemplioriaus reikšmes bei jas išsaugoti. Naudojama programinio konkurencingumo (angl. *concurrency*) kontrolei – keli IS vartotojai negali turėti to paties atidaryto egzemplioriaus, kuriame šis atributas yra „true“;
  - Modifikavimo būklė (::Klausimyno_egzempliorius:Ar_modifikuotas) (angl. *dirty bit*) – nurodo poreikio išsaugoti klausimyno egzempliorių buvimą pakeitus jo būseną. Atributas įgyja „true“ reikšmę, kai pakinta bet kurio klausimyno klausimo egzemplioriaus reikšmių masyvas;

Statinė-dinaminė pusiausvyra ir ryšiai palaikomi klausimyno egzemplioriuje pateikiant nuorodą į statinę dalį, taip suteikiant galimybę nuskaityti reikiamus statinius parametrus.

Nors vaikiniai klausimynai (t.y. jų statiniai aprašai) prijungiami per klausimų sudėtinius domenus, egzemplioriams tai netinka, nes pernelyg ilgas paieškos kelias, nepatogu. Vietoj to visų vaikinių klausimų egzemplioriai talpinami į kolekciją tėvinio klausimyno egzemplioriaus duomenyse.

Analogiškai sprendžiamas ir klausimų egzempliorių klausimas – jie visi pateikiami kolekcijoje, indeksuojamoje pagal klausimo unikalų identifikatorių. Pagal nuorodą bet kuriuo metu nesunku pasiekti norimą klausimo egzempliorių, o ten – ir paties klausimo statinį aprašą.

Kiekvienas klausimynas gali turėti vieną ypatingą klausimą, kurio duomenų tipas – paveikslas. Šio klausimo atsakymo reikšmė būna naudojama sisteminiuose sąrašuose grafinei klausimyno egzemplioriaus reprezentacijai. Klausimyno statiniame apraše nurodomas tokio klausimo unikalus identifikatorius ar kitokia nuoroda.

Klausimų hierarchijos konstruojamos pažingsniui: kiekvienas klausimas turi informaciją apie visus savo vaikinius (žr. 2.1.4) pirmo lygio klausimus, bet pats gali būti

įtraukiamas kaip pirmo lygio vaikinis į neribotą kiekį kitų klausimų. Taip sutaupoma atminties ir apdorojimo greičio, nes nereikia dublikuoti klausimų hierarchijų daugybinio panaudojimo atvejais.

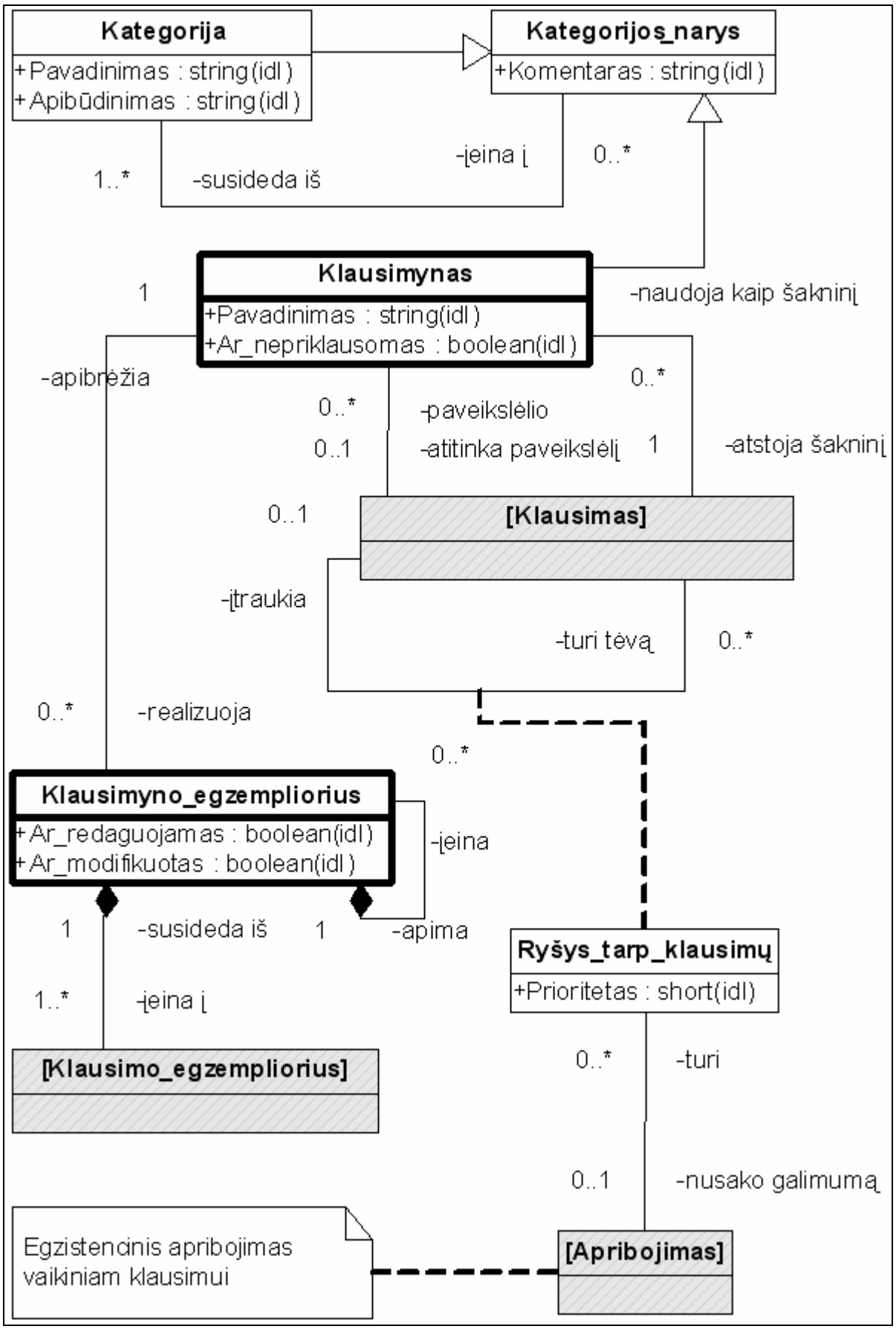
Ryšiai tarp klausimų, būdami egzistenciniais (žr. 2.1.4), gali turėti nenulinį apribojimą (::Apribojimas, žr. „Kardomoji priemonė“, 3.2.3.5), kuris, įvertintas vykdymo metu pasikeitus bent vieno klausimo reikšmėms, gali ryšį nutraukti („false“ atveju).

Klausimynai turi savo taksonomiją, kurios pagrindas – kategorija (::Kategorija):

- Pavadinimas (::Kategorija:Pavadinimas) yra rodomas vartotojui;
- Apibūdinimas (::Kategorija:Apibūdinimas) gali būti rodomas pagalbos sistemoje.

Kiekvienos kategorijos sudėtyje gali būti kitos kategorijos ar individualūs klausimynai; šis dualumas išreiškiamas per polimorfinį (arba, procedūriškai, papildomą tipo lauką) kategorijos narį (::Kategorijos_narys).

Kaip matyti, klausimynas gali įeiti į norimą skaičių kategorijų; taip taksonomija įgyja lankstumo ir leidžia tuos pačius klausimynus pateikti įvairiais rakursais.



10 pav. Klausimyno duomenų struktūros

- Pasekmės

Klausimynas tampa lengvai išplečiamu, patogiai pildomu vaikiniais klausimynais. Sukuriama lanksčios sąlyginė klausimų hierarchija ir klausimynų taksonomija.

3.2.3.4. „Klaustuko galia“

- *Problema*

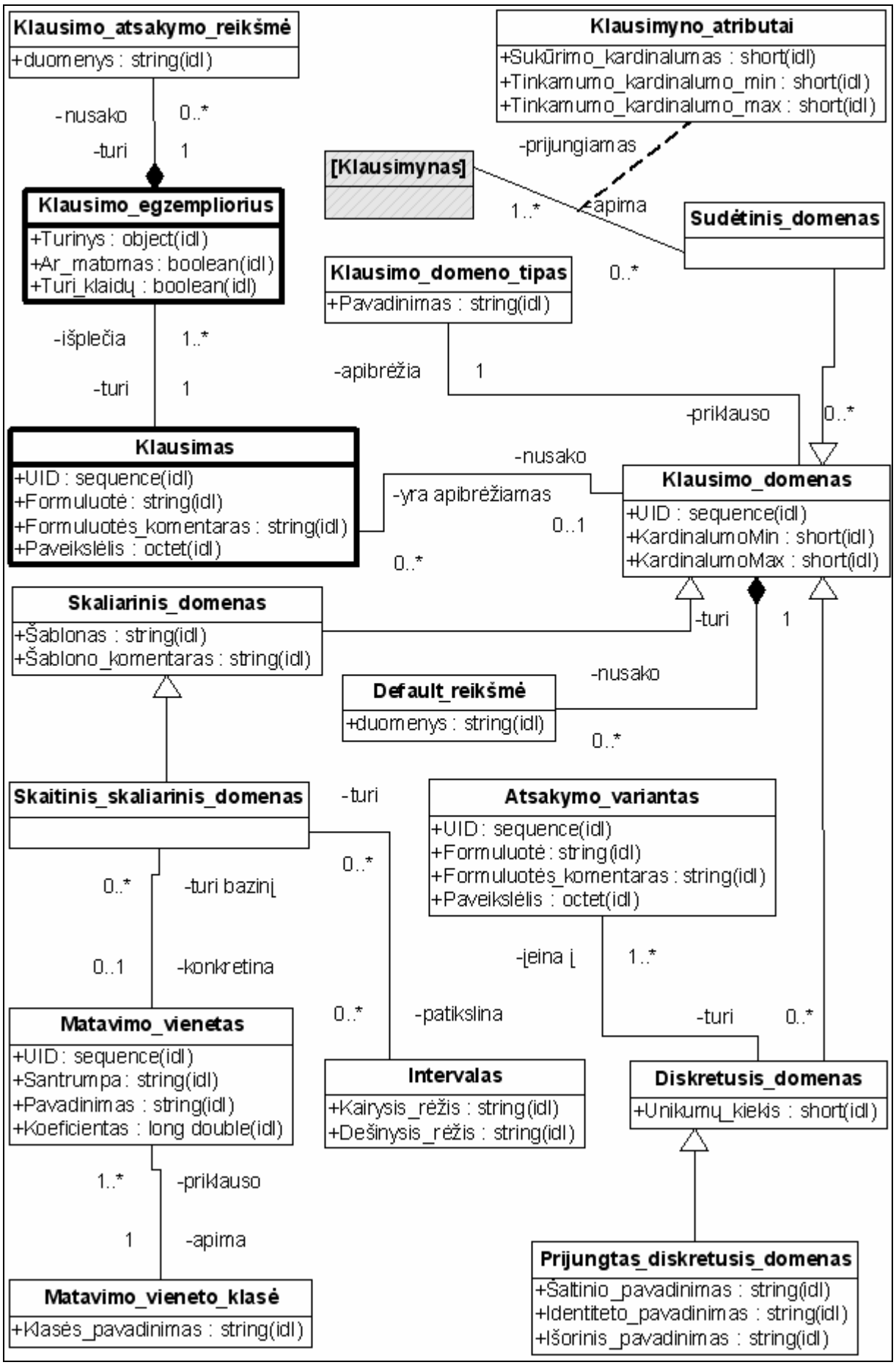
Klausimas, kaip bazinis klausimyno elementas, pernelyg primityviai realizuojamas duomenų struktūromis, nepakankamai atsižvelgiant į išplečiamumo ir struktūrizuotumo poreikius.

- *Kontekstas*

Kuriamai klausimynų IS projektuojamos bazinės duomenų struktūros pagal KOM principus.

- *Sprendimas*

Pagrindinė idėja – aprašomai dalykinės srities esybei (šiuo atveju - klausimui) suteikti lengvai išplečiamą atributų aibę. Kiekvienas klausimo aspektas turi būti aprašomas viename pasiekiamumo lygyje esančiais OOP klasių laukais (angl. *properties*) – arba procedūrinio programavimo atitikmenimis (11 pav.):



11 pav. Klausimo duomenų struktūros

Būtina atskirti klausimo struktūrą (::Klausimas) ir būseną (::Klausimo_egzempliorius):

- Struktūrą sudaro iš KDB gaunami duomenys, nusakantys statines klausimo charakteristikas (dažniausiai nepriklausančias nuo konkretaus klausimo egzemplioriaus reikšmių):
    - Unikalų klausimo identifikatorių (::Klausimas:UID) – naudojamą atsekti klausimo struktūros sąsajoms su klausimo egzemplioriais;
    - Pavadinimą (::Klausimas:Formuluotė) – klausimo formuluotę, visuomet matomą vartotojui;
    - Paaiškinimą (::Klausimas:Formuluotės_komentaras) – įvairiomis formomis vartotojui pateikiamą išplėstinį klausimo formuluotės perfrazavimą;
    - Domeną – atsakymo į klausimą struktūrinį apibrėžimą (::Klausimo_domenas; bus aptartas detaliau);
    - Paveikslėlį (::Klausimas:Paveikslėlis) – rodomą kaip paaiškinimo grafinę versiją arba kaip būtiną kai kurių žodžiais sunkiai išreiškiamų klausimų dalį;
- Būseną atitinka klausimo egzemplioriaus iš RDB (t.y. užpildymo reikšmėmis) duomenys bei į RDB neišsaugomi klausimo būklės parametrai:
    - Reikšmės (::Klausimo_atsakymo_reikšmė) – masyvas, talpinantis vartotojo parinktas (diskrečiojo paprastojo domeno atveju; variantų ID) ar įvestas reikšmes (diskrečiojo domeno atveju – „Kita“ stiliaus vartotojo nurodyti variantai). Kadangi bet kuris elementarusis duomenų tipas (netgi slankiojo kablelio skaičius) gali neprarandant tikslumo būti išreiškiamas tekstinės eilutės (angl. *string*) tipu, būtent eilučių masyvas ir turėtų būti naudojamas šioje vietoje;
    - Aktyvumas (::Klausimo_formuluotė:Ar_matomas) – nurodo galimybės vartotojui įvesti / parinkti / keisti atsakymą buvimą (atitinka keičiamumo atributą pagal 2.1.2);
    - Teisingumas (::Klausimo_formuluotė:Turi_klaidų) – ar esamų atsakymo į klausimą reikšmių visuma patenkina domeno implikuojamas sąlygas;

Įvairių tipų klausimų vartotojo įvestas / nurodytas reikšmes saugant tekstinių eilučių kolekcijoje galima izoliuoti atsakymo į klausimą tipą ir specifiką nuo reikšmes naudojančio

KOM kodo. Reikšmių įvedimas aprašomas klausimo domene (::Klausimo_domenas; žr. 2.1.2), kurį galima priskirti keliems klausimams, taip sumažinant dubliavimą.

Reikšmių kiekį, patenkinantį visišką klausimyno tinkamumą (žr. 2.1.3) galima aprėžti iš abiejų pusių skaitiniais rėžiais (::Klausimo_domenas:KardinalumoMin / ::Klausimo_domenas:KardinalumoMax). Sutarta dešiniojo rėžio reikšmė (angl. *magic number*) gali žymėti teigiamą begalybę (pvz., -1, kuris paprastai nenaudotinas rėžiams nusakyti).

Kiekvienas domenas gali nurodyti norimą kiekį atsakymo reikšmių „pagal nutylėjimą“ (angl. *default*), kurios (::Default_reikšmė) priskiriamos klausimui vos jį parodžius vartotojui. Taip tampa įmanoma priskirti „Jonaitis“ pavardės klausimui ar patogumo dėlei pažymėti (angl. *pre-select*) dažniausiai pasitaikantį pasirenkamąjį atsakymo variantą.

Trys baziniai domeno tipai (skaliarinis, diskretusis, sudėtinis; žr. žemiau) gali būti apjungiami vieningame domeno interfeise (turinčiame KVM komandas: gauti naujausias reikšmes, patikrinti reikšmės tinkamumą pagal domeno apribojimus, ištrinti reikšmes, atstatyti reikšmes „pagal nutylėjimą“, ir t.t.) naudojant polimorfizmą (pvz., virtualias / abstrakčias funkcijas *C++* kalboje). Negalint naudoti *OOP*, procedūriniu aspektu tinka „viskas viename“ principas: visų tipų laukus patalpinti viename duomenų bloke, bet naudoti tik reikiamą poaibį, priklausomai nuo domeno tipo (::Klausimo_domeno_tipas – nurodo ne vieną iš trijų galimų bazinių tipų, bet įtraukia ir galimus potipius, pvz. skaliarinio tipo sveikojo skaičiaus potipį).

Skaliarinis domenas (::Skaliarinis_domenas) – leidžia įvesti elementariųjų duomenų tipų (t.y. tekstinės eilutės, sveikojo / slankiojo kablelio skaičių, datos / laiko) reikšmę(-es), kurios konvertuojamos į tekstinės eilutės pavidalą ir priskiriamos klausimui. Nepriklausomai nuo aprėžto elementariojo duomenų tipo, įvestas reikšmes galima papildomai apriboti šablonu (::Skaliarinis_domenas:Šablonas) – *RegExp* stiliaus išraiška, validuojančia tekstinę įvestų reikšmių formą. Tais atvejais, kai reikšmė neatitinka šablono, vartotojui galima parodyti paaiškinamąjį tekstą (::Skaliarinis_domenas:Šablono_komentaras), kuris pateikia šablono reikalavimų žodinį paaiškinimą.

Tais atvejais, kai skaliarinio domeno potipis yra skaitinis (t.y. sveikasis ar slankiojo kablelio skaičius), galima įvesti papildomų specifinių apribojimų, kurie prijungiami per skaliarinio domeno taksonominį potipį – *skaitinį* domeną (::Skaitinis_skaliarinis_domenas). Pagrindiniai papildomi apribojimai – rėžiai ir matavimo vienetai (žr. žemiau).

Skaitinio domeno reikšmės gali būti aprėžtos intervalais (::Intervalas) - norimu kiekiu aibių sąjunga (angl. *union*) ryšiu susietų individualių skaitinių rėžių, kurie saugomi įprasta tekstinių eilučių forma ir gali būti konvertuojami į nurodytą elementarųjį tipą vykdymo metu.

Vartotojo patogumui skaitiniai domenai gali parūpinti reikšmių konverterį tarp įvairių matavimo vienetų. Sistemoje matavimo vienetai (::Matavimo_vienetas) yra grupuojami į klases (::Matavimo_vieneto_klasė) – pvz., svorio, ilgio, tūrio, ploto. Savaime suprantama, matavimo vienetas negali priklausyti daugiau nei vienai klasei. Jo aprašą sudaro:

- Unikalus identifikatorius (::Matavimo_vienetas:UID), kurį galima naudoti nuorodose (angl. *references*);
- Įprastos santrumpos (::Matavimo_vienetas:Santrumpa), matomos vartotojui;
- Pilno matavimo vieneto pavadinimo (::Matavimo_vienetas:Pavadinimas), kurį galima rodyti kaip paaiškinimą ar naudoti vartotojo sąsajoje vietoje santrumpos;
- Koeficiento (::Matavimo_vienetas:Koeficientas) – svarbiausio lauko, nusakančio vienai klasei priklausančių matavimo vienetų tarpusavio konversijos koeficientus (savaime suprantama, tai - slankiojo kablelio skaičiai). Bent vienas klasės matavimo vienetas šiam laukui nustatyti bazinę reikšmę 1.0; visi kiti tos klasės matavimo vienetai koeficientus nurodo reliatyviai šio atžvilgiu.

Diskretusis domenas (selektorius) – leidžia pasirinkti nurodytą kiekį iš anksto pateiktų variantų. Parinktų variantų (::Atsakymo_variantas) reikšmės tekstinės eilutės forma gali būti pateikiamos kaip tekstinės variantų formuluotės (::Atsakymo_variantas:Formuluotė), bet optimaliau naudoti unikalius identifikatorius (::Atsakymo_variantas:UID), kurie nekinta ir yra glaustesni. Galimų atsakymo variantų vaizdumą išplečia deskripciniai laukai ::Atsakymo_variantas::Formuluotės_komentaras (gali būti rodomas kaip paaiškinamasis tekstas (angl. tooltip)) bei ::Atsakymo_variantas:Paveikslėlis (ypač tiems variantams, kuriuos sunku aprašyti tekstu).

Turint omenyje, kad sunku ir / ar nepatogu pateikti teoriškai išbaigtą atsakymo variantų aibę („Gausybės ragas“, žr. 3.2.3.12), kartais pravartu leisti vartotojams įvesti savus variantus (vadinamasis „Kita“ atvejis). Skaitinis laukas (::Diskretusis_domenas:Unikumų_kiekis) su reikšme, didesne už 0, nurodo būtinybę pateikti „Kita“ variantą su standartiniu skaliariniu (tekstinės eilutės potipio) domenu vartotojo variantų įvedimui. Įvesti variantų tekstai priskiriami klausimo reikšmėms; jie neturi unikalių identifikatorių. KOM / KVM kode įvesti variantai atpažįstami pagal reikšmių vidinį tipą: įvestų variantų negalima konvertuoti į skaitinius unikalius identifikatorius (::Atsakymo_variantas:UID).

Atskiras diskrečiojo domeno atvejis (::Prijungtas_diskretusis_domenas) apima galimų atsakymo variantų gavimą (angl. extraction) iš išorinių duomenų šaltinių – prenumeruojamų

DB, sukauptos statistikos, vartotojo anksčiau įvestų reikšmių ir kt. Atsakymų variantai ištraukiami klausimyno struktūros užkrovimo metu. Apibendrinta domeno sudėtis įtraukia:

- Duomenų šaltinio pavadinimą (::Prijungtas_diskretusis_domenas:Šaltinio_pavadinimas), kuris gali būti kompozicinis (*URL*, DB adresas / pavadinimas / lentelė, ir t.t.) ar abstrahuoti detales per izoliuojantį interfeisą;
- Varianto unikalų identifikatorių (::Prijungtas_diskretusis_domenas:Identiteto_pavadinimas), nurodantį šaltinyje esančius variantus unikaliai apibrėžiantį sveikojo skaičiaus tipo lauką / duomenų bloką (atitinka ::Atsakymo_variantas:UID);
- Varianto pavadinimą (Prijungtas_diskretusis_domenas:Išorinis_pavadinimas), pagal prasmę atitinkantį ::Atsakymo_variantas:Formuluotė;

Sudėtinis domenas (::Sudėtinis_domenas) įgalina prijungti norimą kiekį klausimynų tėvo – vaiko ryšiais (žr. 2.1.4) prie esamo klausimo, o globaliame kontekste – prie klausimyno, kuriam priklauso einamasis klausimas („Magnus Rex“, žr. 3.2.3.3). Reikšmių požiūriu klausime esančiam atitinkamam masyvui (::Klausimo_atsakymo_reikšmė) priskiriami klausimynų egzempliorių, prijungtų prie klausimo, kontekste unikalūs identifikatoriai.

Kiekvienas vaikinis klausimynas prijungimo metu gali būti konfigūruojamas (::Klausimyno_atributai):

- Pradinis kardinalumas (::Klausimyno_atributai:Sukūrimo_kardinalumas) – nurodo kiekį vaikinio klausimyno egzempliorių, sukuriamų tėvinio klausimyno egzempliorių;
- Tinkamumo kardinalumo intervalas (::Klausimyno_atributai:Tinkamumo_kardinalumo_min / Tinkamumo_kardinalumo_max) – nurodo intervalą leistino egzempliorių skaičiaus visiškam klausimyno užpildymui pasiekti duoto klausimo požiūriu (t.y., jeigu egzempliorių skaičius nepatenka į intervalą, klausimo atributas ::Klausimo_egzempliorius:Turi_klaidų tampa teigiamu ir klausimynas automatiškai praranda visiško užpildymo statusą.

- *Pasekmės*

Pasiekiama universali klausimo reprezentacija, pasižyminti didžiąja dalimi įvairių dalykinių sričių klausimams reikalingų savybių bei pakankamai modulinė tolesniems atributų papildymams.

3.2.3.5. „Kardomoji priemonė“

- *Problema*

Dažniausiai pasitaikanti apribojimų (angl. *constraints*) struktūra yra statiška (pvz., „klausimo reikšmė = konstanta“) - suvaržanti klausimynų projektuotojo išraiškos laisvę ir apimanti nykstamai menką galimų apribojimų variantų aibės dalį, - yra kliūtis vėlesniam klausimynų IS išplečiamumui ir tinkamumui.

- *Kontekstas*

Egzistenciniams ir / ar procedūriniams ryšiams tarp klausimų (žr. 2.1.4) realizuoti klausimynų IS projektuotojui būtina taikyti loginius apribojimus.

- *Sprendimas*

Klausimynų IS loginiai apribojimai yra specifiniai, duomenis lyginamosioms operacijoms ima (dažniausiai) iš esamo klausimyno egzemplioriaus nurodytų klausimų reikšmių. Nereikia pamiršti ir būtinybės realizuoti galimybę įtraukti predikatus, kurie tikrina klausimyno išorėje esančių objektų būklę.

Apibendrinant, universali ir išsami loginių apribojimų forma turi pasižymėti:

- Smulkiu suskaidymu (angl. *fine granularity*) – susidėti iš atominių loginių operacijų (t.y. predikatų), apjungtų loginiais operatoriais iki norimo išraiškos sudėtingumo;
- Didele predikatų bei funkcijų gausa:
    - Klausimynų konteksto: klausimų reikšmių / reikšmių kiekio lyginamosios operacijos su konstantomis, su kitų klausimų reikšmėmis;
    - Sistemos konteksto: data, lokalizacija, vartotojo asmeniniai duomenys (paprastai tokiems predikatams apskritai nereikia argumentų);

- Nesudėtingu interpretavimo algoritmu;

Norimas savybes realizuojančių duomenų struktūrų sudėtis priklauso nuo apribojimo interpretacijos prigimties, kurių galima išskirti dvi:

- Tiesioginė – apribojimo turinys išreiškiamas klausimynų IS programavimo kalbos išeities teksto forma (pvz, „(500 > 500“) && („Jonas“.contains(„o“)) C# kalboje) bei dinamiškai įvykdomas (kai kurios programavimo kalbos turi specialius konstruktus teksto forma pateikto išeities teksto vykdymui, kitos gali dinamiškai konstruoti kodo modulius). Toks atvejis galimas interpretuojamoms programavimo kalboms (*Javascript*, *PHP*, *LISP*), į kurių tarpą įeina ir valdomosios (angl. *managed*): *.NET C#* / *C++* / *VB* / *JScript*, *Java*;
- Speciali (tikroji interpretacija) – duomenų struktūros, talpinančios apribojimo elementus, yra nuosekliai nuskaitomos. Algoritmas vykdo rastas komandas ir konstruoja loginį rezultatą. Savaime suprantama, toks interpretatorius veikia lėčiau nei tiesioginiu atveju, kadangi dinaminis vykdymas dažniausiai pasižymi papildomomis greitaveikos optimizacijomis, neprieinamoms toje pačioje aplinkoje veikiančiam interpretatoriui;

Elementarus sprendimas – saugoti apribojimus kaip norimos programavimo kalbos kodo sakinius, - naudingas pakankamai retai ir nereikalauja bent kiek detalesnio aptarimo. Toliau aptariamas sudėtingesnis, bet universalesnis sprendimas, naudotinas dviem dažniausiai pasitaikančiais atvejais:

- Esant specialiajai apribojimų interpretacijos prigimčiai (taikomas dėl vykdomosios aplinkos / programavimo kalbos / greitaveikos problemų);
- Esant tiesioginei apribojimų interpretacijos prigimčiai, bet turint tikslą apribojimus saugoti struktūrizuotai suskaidytus (pvz., norint supaprastinti klausimynų / apribojimų redaktoriaus algoritmą)

Esminė idėja – apribojimui aprašyti naudoti lenkiškąją notaciją (angl. *polish notation*), kurios savybės – aiškumas, neriboto sudėtingumo išraiškos bei paprastas nuskaitymas, - praverčia tiek duomenų struktūrų sudėtingumo, tiek ir greitaveikos požiūriais.

Svarbiausia apribojimo (12 pav.) sudaromoji dalis – apribojimo elementas (::Apribojimo_elementas), iš esmės apimantis paprastą ar predikatinę (t.y. grąžinančią Būlio logikos reikšmę) funkciją (::Funkcija). Predikatinei funkcijai gali prireikti neriboto kiekio argumentų (::Apribojimo_argumentas), kurie gali būti vieno iš trijų tipų:

- *Konstanta* – pateikiama tekstinės eilutės forma; funkcija ją interpretuoja pagal poreikį;

- *Nuoroda į reikšmę*, grąžinamą analogiško apribojimo elemento (dažniausiai elemente naudojama ne predikatinė, o paprasta funkcija);
- *Nurodyto klausimo egzemplioriaus dabartiniame klausimyne reikšmė(-s)*. Klausimų egzemplioriai gali būti atsekami žinant klausimus pagal KOM sąrašus, žr. „Pilkasis kardinolas“, 3.2.3.2;

Turint aibę elementarių predikatinių apribojimo elementų, juos galima apjungti jungtyse (:Jungtis:) įvairiais loginiais sąryšiais (nurodomais ::Jungties_predikatas), kurių argumentais (::Jungties_operandas) gali būti apribojimo elementai arba pačios jungtys. Prisimenant, kad jungties operacijos, kaip taisyklė, būna iš Būlio logikos srities („ir“, „arba“, „ne“...), yra logiška supaprastinti schemą ir naudoti du argumentus.

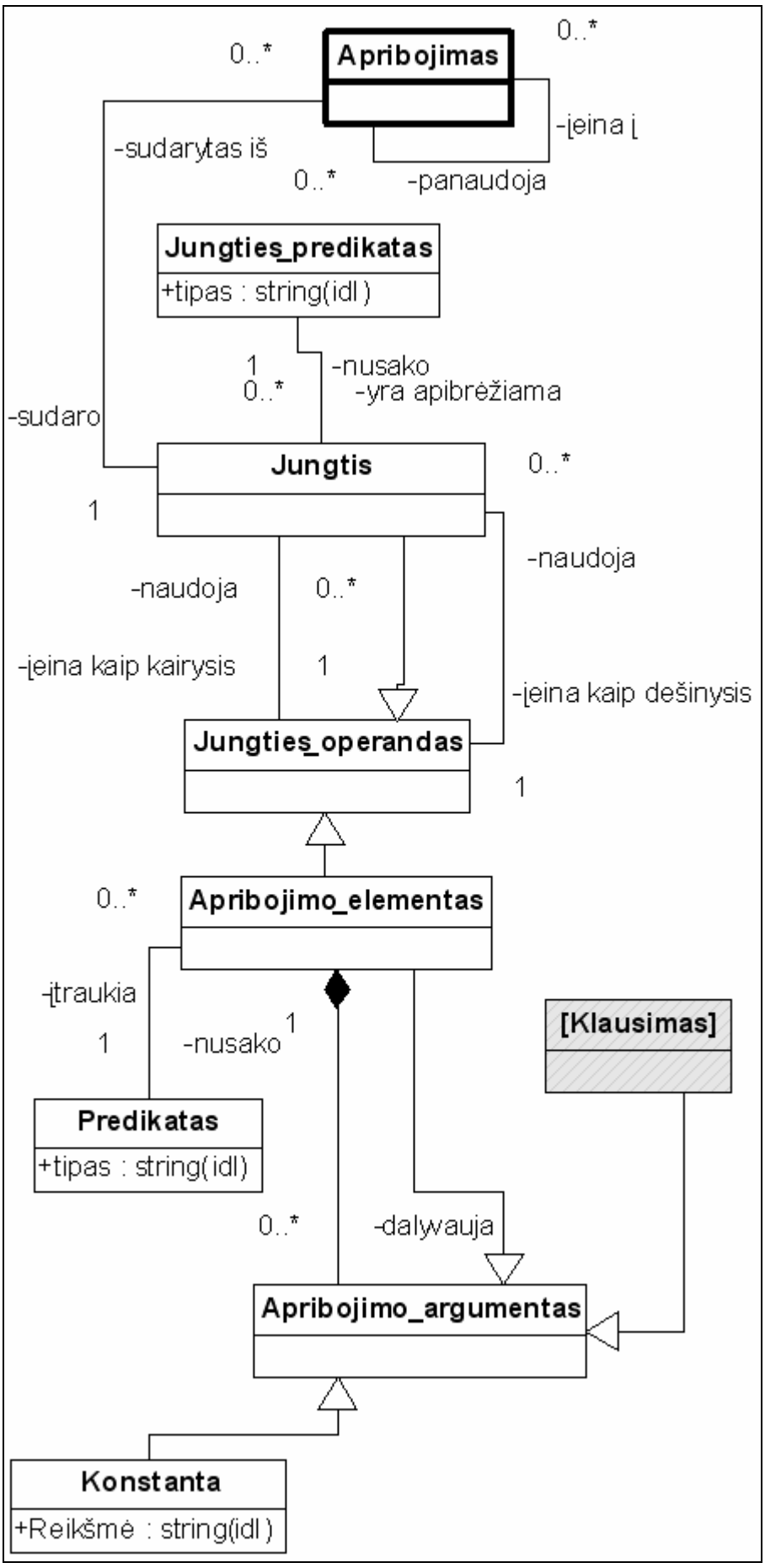
Pats apribojimas (::Apribojimas) tėra nuoroda į aukščiausio lygio jungtį, kurios argumentai hierarchiškai sujungia kitas jungtis ir elementus.

Savaime suprantama, aukščiausio lygio (taip pat ir inversinė „ne“) jungtis gali būti primityvi, sudaryta tik iš vieno apribojimo elemento, apgaubto jungties argumente. Tokiu atveju antrąją nuorodą galima palikti tuščią (angl. *null*).

Kelis apribojimus galima apjungti paprastu konjunkciniu sąryšiu, taip sumažinant pasikartojimą.

Žvelgiant į struktūrinę schemą gali kilti klausimas, kodėl apribojimo elementui nesuteikti galimybių įtraukti jungčių predikatus ir supaprastinti visą struktūrą pašalinant ::Jungtis bei ::Jungties_operandas. Atsakymas slypi greitaveikoje: apribojimo elementų funkcijos gali būti įvairios, reikalauti neriboto kiekio argumentų, o jungtys yra specifiškai loginės ir reikalauja daugiausia dviejų argumentų, todėl jų atveju galima naudoti supaprastintą algoritmą.

Interpretavimo algoritmas lenkiškąja notacija užrašytam apribojimui yra labai paprastas ir remiasi faktu, kad lenkiškosios notacijos išraiškoje bet kuriuo laiko momentu bent viena funkcija turi parengtus naudojimui argumentus: „Rasti pirmąją pasitaikiusią funkciją, po kurios seka reikiamas kiekis determinuotų argumentų, įvykdyti funkciją, rezultatą patalpinti vietoj jos ir argumentų. Kartoti tol, kol išraiškoje beliks vienintelis narys - konstanta“.



12 pav. Apribojimo duomenų struktūros

- Pasekmės

Konstruojant apribojimą iš atominių elementų, naudojant neribotus galimų funkcijų sąrašą ir logines sąsajas pasiekiamas pakankamas universalumas, bet beveik nepasunkėja vykdymo algoritmas

3.2.3.6. „Projekcija“

- *Problema*

  Klausimynus sudarantys įvairių tipų klausimams sunku parinkti egzempliorių reikšmių saugojimo ((de)serializavimo, angl. *(de)serialization*) RDBVS formatą, pasižymintį:
  - Universalumu;
  - Pakankama greitaveika.

- *Kontekstas*

  Klausimynų projektuotojas turi sukurti RDB realizaciją pasirinktoje RDBMS.

- *Sprendimas*

  Priimant, jog kiekviename klausimyne individualus klausimas gali būti įtraukiamas tik vieną kartą (kriterijus: UID), galima klausimynų egzempliorius serializuoti supaprastintu formatu, taip padidinant greitaveiką.

  Nepriklausomai nuo klausimyno sudėtingumo ir klausimų hierarchijos gylio, klausimyno egzemplioriaus duomenis galima įsivaizduoti kaip dvimatį masyvą M, indeksuojamą pagal klausimo UID ir įvestos klausimo reikšmės numerį:

  M[*klausimo_UID***][***reikšmės_indeksas***]**

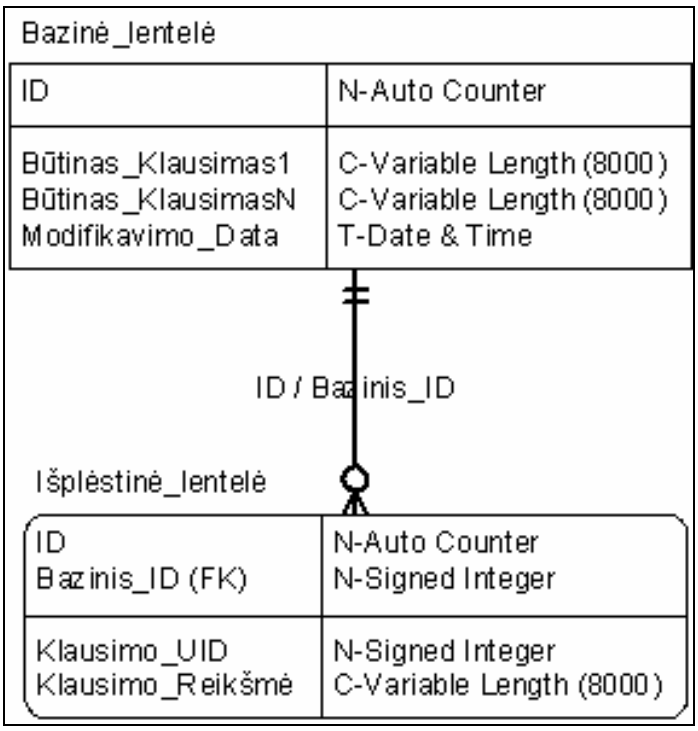
Homogeniškumą dar padidina teiginys, kad visos klausimų reikšmės – nepriklausomai nuo klausimo domeno tipo ir kitų faktorių, - atmintyje saugomos tekstinės eilutės forma. Iš to išplaukia, kad masyvo M elementai yra išimtinai tekstinės eilutės tipo.

Akivaizdžiausias problemos sprendimas – visus masyvo elementus (pagal pirmąjį indeksą) patalpinti į vieną lentelę su nuo klausimyno dydžio priklausančio laukų kiekiu, - įveda nemažai apribojimų:

- Pakeitus klausimyno struktūrą (pridėjus / pašalinus klausimų), tenka SQL DDL komandomis keisti lentelės struktūrą, kas kenkia universalumo kriterijui;
- Dažnai RDBVS turi apribojimų lentelės laukų kiekiui ir greitaveikai, kai laukų kiekis yra didelis.

Kur kas lankstesnis sprendimas – naudoti hibridinę dviejų lentelių schemą vienam klausimynui atvaizduoti (13 pav.):

- Bazinė lentelė – talpina pačius pagrindinius klausimyno egzemplioriaus duomenis: automatiškai generuojamą UID, modifikavimo datą, visų klausimyno klausimų, pažymėtų kaip „būtini“, reikšmes;
- Išplėstinė lentelė – talpina visų įprastų klausimyno egzemplioriaus klausimų reikšmes. Klausimų reikšmių masyvas įrašomas kita dimensija, nei aukščiau aprašytame akivaizdžiausiame sprendime.



13 pav. Klausimyno reprezentacija RDBMS

Toks paskirstymas yra logiškas keliomis prasmėmis:

- Tik kelių klausimų („būtinų“) reikšmės turi būti serializuojamos viename tekstiniame lauke (pvz., apjungiant kabliataškiais; tikslinė denormalizacija) – visi kiti klausimai su keliomis įvestomis reikšmėmis gali užimti kelis įrašus išplėstinėje lentelėje;
- Bazinė lentelė gali turėti sisteminių laukų, susijusių su daugybiniu resursų naudojimu (angl. *concurrency*), įvairiomis egzemplioriaus gyvavimo ciklo datomis, pilnumo indikatoriais, redagavusių vartotojų ID ir t.t. Nereikia pamiršti, kad bazinė lentelė privalo turėti bent vieną sisteminį lauką, nes kitu atveju, klausimyne nesant būtinų klausimų, bazinėje lentelėje lieka vien tik automatiškai pildomas ID laukas – o daugelis DBVS tokiu atveju neleidžia atlikti *SQL DML INSERT* sakinių.

Greitaveika užtikrinama, nes:

- Visur naudojami skaitiniai unikalūs identifikatoriai, todėl indeksai užima nedaug vietos ir greitai skaitomi;
- Atsakymų į klausimus reikšmėms saugoti naudojami kintamo ilgio tekstiniai laukai – taupantys DB vietą ir taip sumažinantys orientacijos (angl. *seeking*) laiko kaštus;

- Atsakymų į klausimus duomenys saugoti nedideliame kiekyje lentelių ir nesudaro DB lygio hierarchijos, todėl SQL DML operacijos yra trumpos ir efektyviai optimizuojamos paties RDBVS priemonėmis.

- *Pasekmės*

Klausimynų egzempliorių serializavimas individualiose lentelių porose sumažina vidutinį tikėtiną įrašų skaičių lentelėse, bet visgi gali sumažinti greitaveiką ypač dideliuose duomenų rinkiniuose.

3.2.3.7.„Vienišas horizontas“

- *Problema*

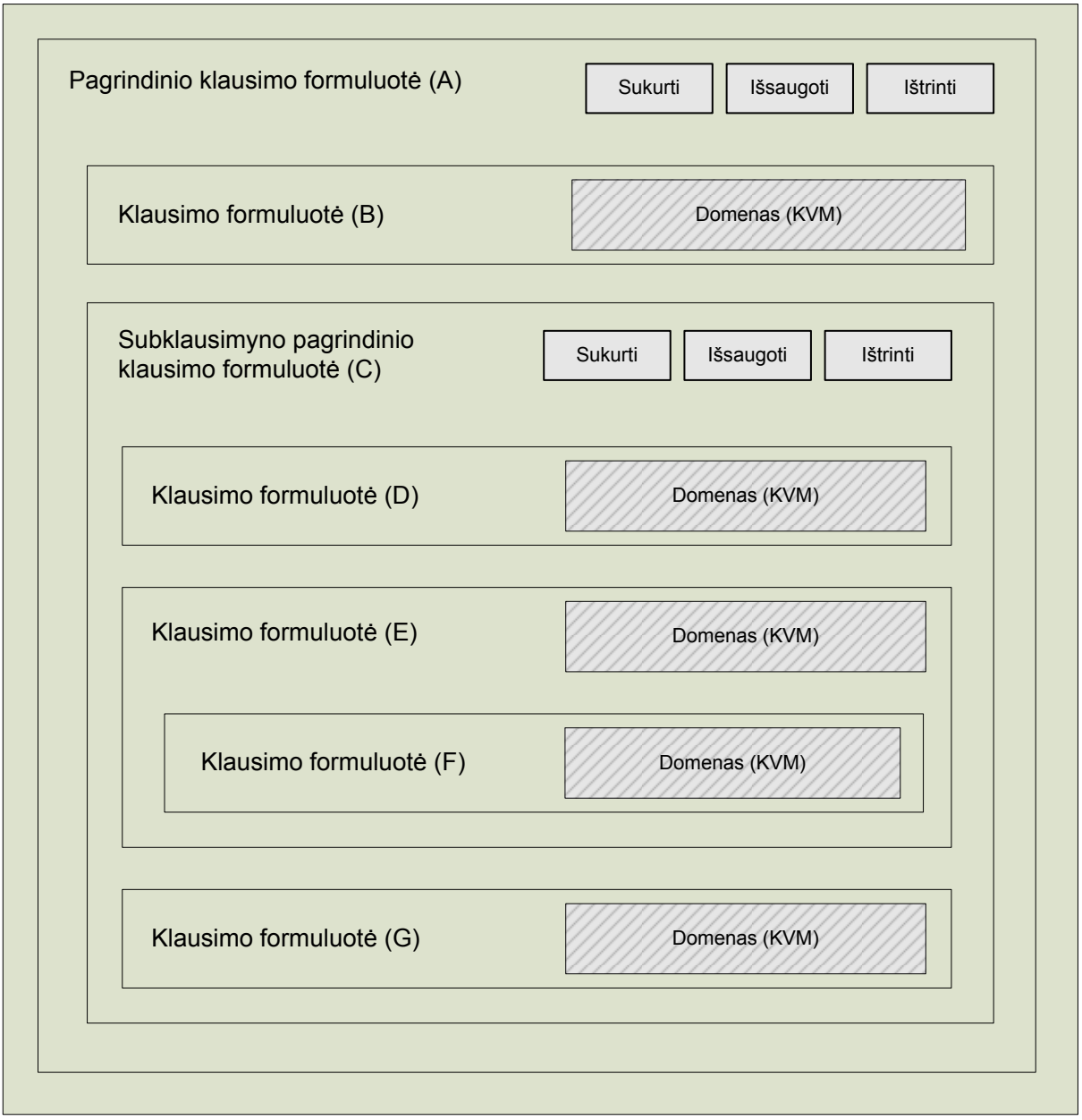
Yra poreikis klausimyno KOM pateikti vartotojui suprantama forma.

- *Kontekstas*

Klausimynų IS projektuotojas konstruoja KOM atitikmenį grafinėje vartotojo sąsajoje. Vaizdumas labiau pageidaujamas nei glaustumas ar navigavimo lengvumas.

- *Sprendimas*

Pats natūraliausias (ir lengviausiai realizuojamas) variantas – tiesioginis KOM hierarchijos pateikimas grafine forma. Kadangi KOM – medis, kurio lapai – klausimai, o šakos – ryšiai tarp klausimų, pasitelkiama klasikinė medžio atvaizdavimo forma (naudojama daugelyje OS *treeview* pavadinimu) (14 pav.).



14 pav. Klausimyno pateikimo struktūra „Vienišo horizonto“ atveju

Hierarchijos gylio vizualizacija pasiekiama naudojant atitraukimą nuo kairiojo lango krašto (angl. *indent*). Vaikiniai klausimai atitraukiami toliau, nei tėviniai. Taip pat vaikiniai klausimai vizualiai apgaubiami tėvinio klausimo kontūru, taip pabrėžiant nuosavybę.

Sudėtiniai klausimų domenai (žr. 2.1.2), prijungiantys prie esamojo medžio vaikinio klausimyno egzemplioriaus medį (per klausimą C, žr. 14 pav.), įtakoja hierarchiją ir grafiškai yra atvaizduojami kaip vaikiniai klausimai, tačiau prideda ir valdančiųjų komandų (dažniausiai – mygtukų ar kontekstinio meniu pavidalu; sisteminis meniu nėra optimalus dėl savo globalumo).

Pagrindinis klausimynas nuoseklumo dėlei turėtų taip pat būti aprūpinamas tokiomis pat valdančiosiomis komandomis kaip ir vaikiniai klausimynai, tačiau privalo būti valdomas ir per sisteminį meniu (priešingai nei efemeriški vaikiniai klausimynai, pagrindinis klausimynas visuomet aktyvus).

(Vaikinio) klausimyno valdančiosios komandos apima pagrindinius veiksmus, įtakojančius tik tą klausimyno egzempliorių, kuriam buvo sukurtos. Komanda „Sukurti“ turėtų leisti pasirinkti tarp naujo klausimyno egzemplioriaus sukūrimo bei pateikimo pildymui ir egzistuojančio reikiamo tipo klausimyno parinkimo iš sąrašo / paieškos medžio (žr. XXX).

- *Pasekmės*

KOM atvaizduojamas labai intuityviai bei suprantamai, tačiau paprasčiausia forma kartu yra ir sunkiausia naudoti. Neigiamos pasekmės:

- Beveik visuomet didelė klausimų dalis vartotojui nematoma vienu metu dėl didelės vertikalios apimties;
- Dideliuose klausimynuose (pvz., produktų valdymo – 1000 klausimų) navigacija tampa labai neefektyvi, nekalbant jau apie greičio netekimą dėl didžiulio kiekio grafinių OS primityvų, kuriuos reikia vienu metu sutalpinti į konteinerį ir rodyti dalimis;
- Esant giliai hierarchijai (pvz., produktų valdyme – 10 lygių), vis didėjantis atitraukimas nuo kairiojo lango krašto gali paslėpti dalį klausimų ar jų zonų; vartotojas būtų priverstas naudoti horizontalias slankjuostes.

  Būtina pasverti privalumus ir trūkumus.

3.2.3.8.„Savivalda“

- *Problema*

Yra poreikis klausimyno KOM pateikti vartotojui suprantama forma.

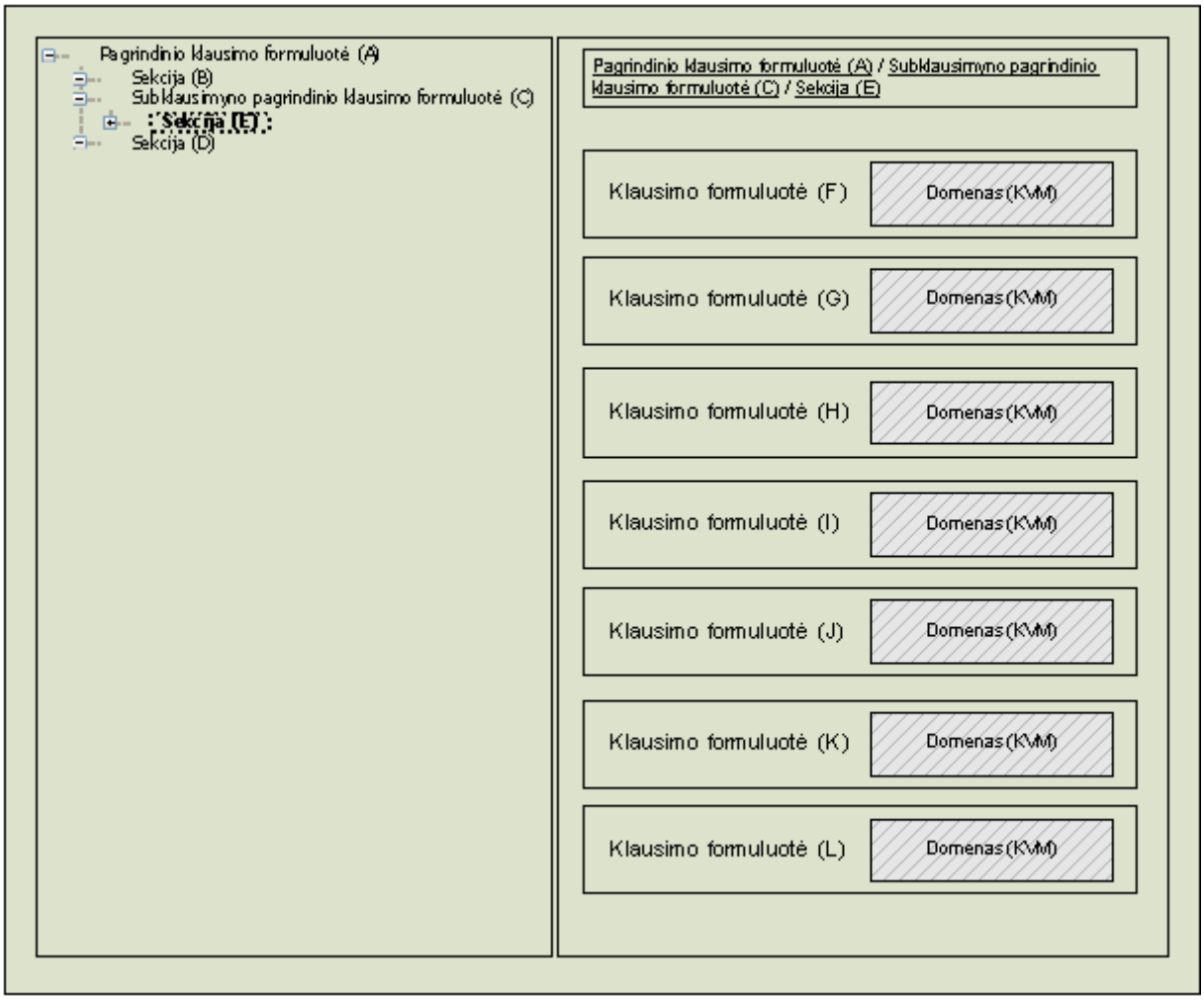
- *Kontekstas*

Klausimynų IS projektuotojas konstruoja KOM atitikmenį grafinėje vartotojo sąsajoje. Siekiama balanso tarp vaizdumas ir glaustumo / navigavimo lengvumo.

- *Sprendimas*

Naudojantis „skaldyk ir valdyk“ principu tarpusavyje sujungtų klausimų aibė suskaidoma savotiškais miniklausimynais – sekcijomis. Sekcija apgaubia logiškai susietų klausimų aibę. Sekcija taksonomiškai yra klausimas, taigi, į sekciją gali įeiti norimas kiekis kitų sekcijų. Palyginimui – katalogo ir failo sąvokos OS failų sistemoje ganėtinai tiksliai atitinka sekcijos ir klausimo sąvokas (įskaitant faktą, kad nemažai OS katalogą realizuoja specialaus turinio failu).

Kita svarbus žingsnis – atskirti sekcijų ir paprastų klausimų vizualizaciją. Tai padaroma sukuriant dvi atskiras lango zonas (15 pav.). Pirmoji talpina hierarchinį sekcijų atvaizdį (nerodant nė vieno paprasto klausimo), antroji – esamu momentu sufokusuotos sekcijos vaikinius klausimus (praleidžiant ten galbūt įsipynusias vaikines sekcijas).



15 pav. *Klausimyno pateikimo struktūra „Savivaldos“ atveju*

Sufokusavus sekciją, kurią reprezentuojantis KOM klausimas turi sudėtinį domeną, dešiniojoje zonoje gali būti parodoma panaši grafinių primityvų kolekcija, kaip „Žingsnis po žingsnio“ (žr. 3.2.3.9) atveju.

Analogiškai „Žingsnis po žingsnio“ atvejui, navigacijos juosta įgalina lengvai pereiti į bet kurį sufokusuotos sekcijos tėvą hierarchijoje.

- *Pasekmės*

Šis KOM atvaizdavimo metodas neturi stipriai pasireiškiančių trūkumų, yra ganėtinai optimalus:

- Navigacija lengvesnė dėl aiškiai išreikšto ir glausto sekcijų hierarchijos medžio, veikiančio kaip bibliotekos indeksas;
- Vienu metu matomas pakankamas klausimų kiekis, kad susidaryti aiškesnį vaizdą iš konteksto;

- Laikantis siūlymo apriboti į sekciją įeinančių klausimų maksimalų kiekį (paprastai - 5-30), dažnai net nereikia naudoti slankjuostės dešinėje zonoje;
- Sąlyginai nedidelis vienu metu rodomų klausimų kiekis gali palengvinti grafinės posistemės ar KOM generatoriaus apkrovą ir sumažinti vėlinimą (angl. *latency*);
- Užimama ekrano erdvė nėra tokia maža, kad tiktų mažaekraniams įrenginiams, bet vidutinio dydžio ekranus turintys įrenginiai (planšetiniai (angl. tablet), nešiojamieji (angl. laptop) kompiuteriai) puikiai tinka. Bet kuriuo atveju, tik horizontali dimensija yra kritinė; vertikaliąją galima labai sumažinti naudojant slankjuostes.

3.2.3.9. „Žingsnis po žingsnio“

- *Problema*

  Yra poreikis klausimyno KOM pateikti vartotojui suprantama forma.

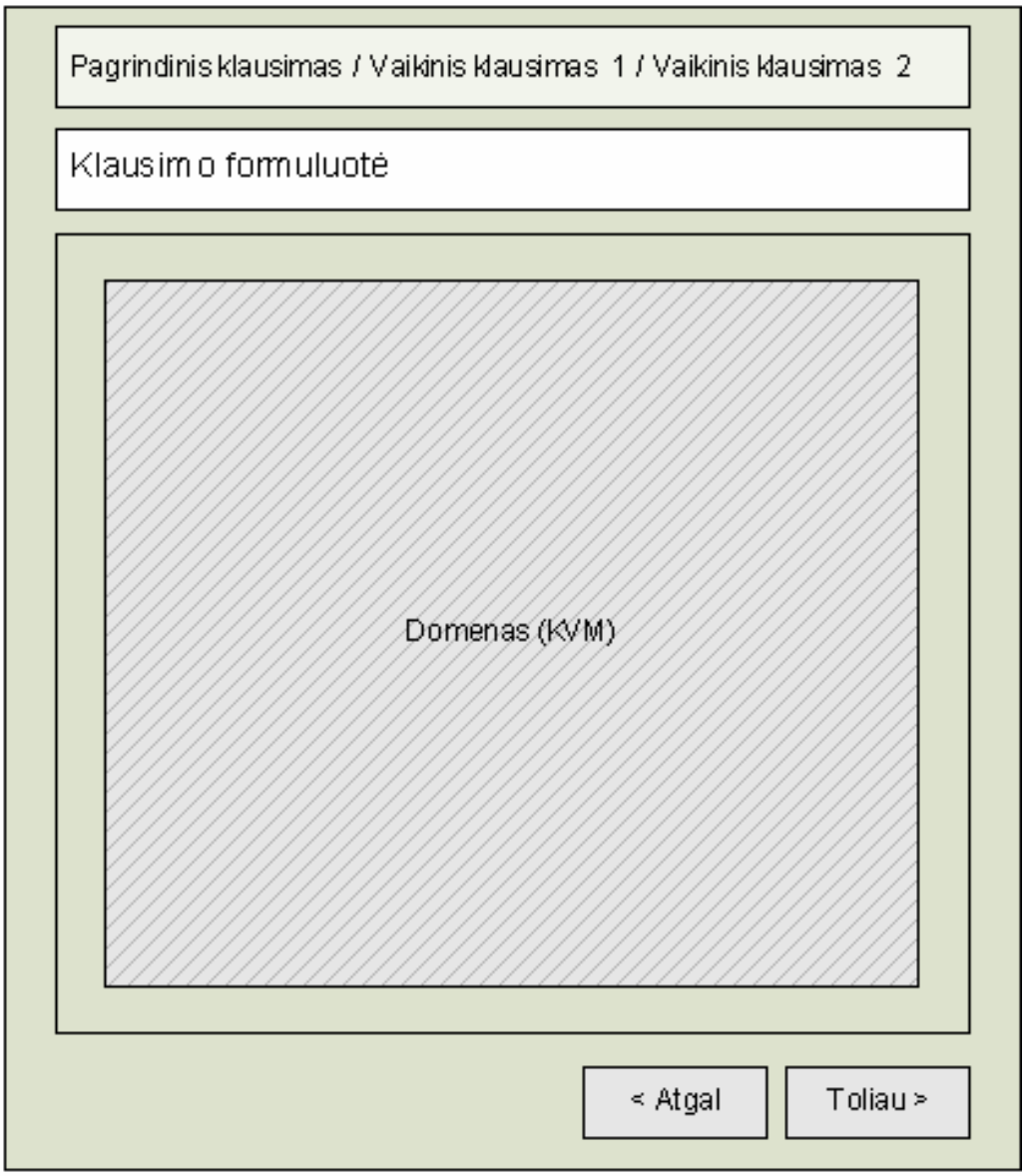
- *Kontekstas*

  Klausimynų IS projektuotojas konstruoja KOM atitikmenį grafinėje vartotojo sąsajoje. Tikslinė platforma – mažaekranis prietaisas (delninis kompiuteris, protingasis mobilusis telefonas (angl. *smartphone*)).

- *Sprendimas*

  Esant labai apribotai ekrano erdvei – apie 100 – 200 rastrinių taškų (angl. *pixels*), nepraktiška mėginti atvaizduoti keletą klausimų hierarchijos lygių ar apskritai keletą klausimų, kadangi navigacija slankjuostėmis bus labai sudėtinga ir neigiamai nuteikianti.

  Optimalus variantas – fokuso (o ne elementų vaizdavimo drobės (angl. *canvas*)) naudojimas. Fokuso elementu laikomas vienas klausimas, kuris konkrečiu metu gali būti parodomas ekrane – tačiau tik vienas vienu metu (16 pav.):



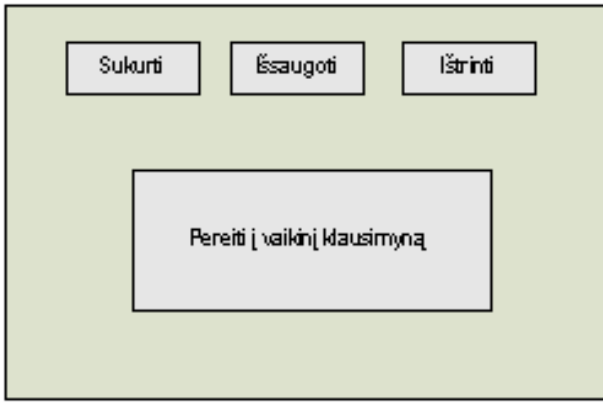
16 pav. Klausimyno pateikimo struktūra „Žingsnis po žingsnio“ atveju

Hierarchijos suvokimą pagerina neprivaloma navigacijos juosta, nurodanti kelią iki sufokusuoto klausimo. Pageidautina kelio elementuose įdiegti hipernuorodas (angl. *hyperlinks*), leidžiančias naviguoti dideliais žingsniais.

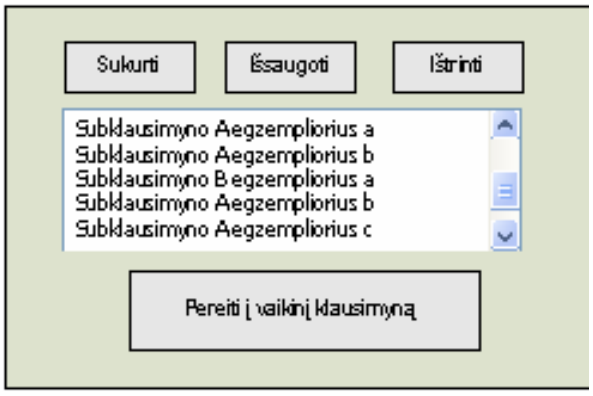
Klausimo formuluotė užima kitą eilutę; taip taupoma vieta, be to, tekstui nebetelpant, navigacijos juosta ir pavadinimas gali padidinti savo aukštį ir tapti daugiaeilutėmis (angl. *multiline*).

Visą likusią ekrano erdvę užima domenas. Kadangi paprastai domenų vertikali dimensija būna nedidelė, puslapiavimo problemos – retos. Egzistuoja du atvejai, reikalaujantys papildomo dėmesio:

- Diskretusis domenas – „Gausybės ragas“ (žr. 3.2.3.12) taikymas dažnai sukelia didelio variantų kiekio atsiradimą, taip padidinat vertikalios dimensijos matmenį ir sukeliant problemą mažaekraniuose įrenginiuose. Minėtas „Gausybės ragas“ siūlo ir išeitį - kompaktišką versiją, aprašytą „Žynyje“ (žr. 3.2.3.10).
- Sudėtiniai domenai – įvesdami papildomus hierarchijos lygius ir klausimus potencialiai maksimizuoja domeno vertikalios dimensijos matmenį ir labai netinka mažaekraniams įrenginiams. Vėlgi patartina fokusą skaidyti mažoms diskrečioms dalims (17 pav. – sudėtinio paprastojo, 18 pav. – sudėtinio kompozicinio pavyzdžiai):



17 pav. Sudėtinio paprastojo domeno atvaizdis „Žingsnis po žingsnio“ atveju



18 pav. Sudėtinio kompozicinio domeno atvaizdis „Žingsnis po žingsnio“ atveju

Navigacija paprastai vykdoma dviem mygtukais „Atgal“ ir „Toliau“ (16 pav.), kurie keičia fokusą pereidami KOM medį paieška į gylį (angl *depth-first search*). Tokiu būdu įmanoma fokusuoti visus paprastus klausimus.

Sudėtinių domenų atveju perėjimą tarp tėvinio klausimo ir pirmojo vaikinio klausimyno klausimo atlieka mygtukas „Pereiti į vaikinį klausimyną“

(17, 18 pav.). Atvirkštinę funkciją atlieka mygtukas „Atgal“ – tuomet, kai fokusuojamas pagrindinis vaikinio klausimyno klausimas, nes būtent šiame klausime minėtasis mygtukas neturi paskirties.

Papildomos navigacijos galimybės gali būti patalpinamos aukščiau minėtose hipernuorodose.

- *Pasekmės*

Siūloma KOM atvaizdavimo forma yra labai glausta ir nereikli vaizduoklio dydžiui bei grafinės posistemės galiai, tačiau paini naviguoti (dėl bendro vaizdo trūkumo; visuomet matomas tik sufokusuotas klausimas).

3.2.3.10. „Žynys“

- *Problema*

Dėl įvairių priežasčių (žr. “Gausybės ragas” motyvaciją, 3.2.3.12 bei “Eiliškumas”, 3.2.3.19) vartotojui nepatogu naudotis klasikine diskrečiojo domeno pateikimo forma: vertikaliu sąrašu; gali būti įtakojamas rezultatų teisingumas.

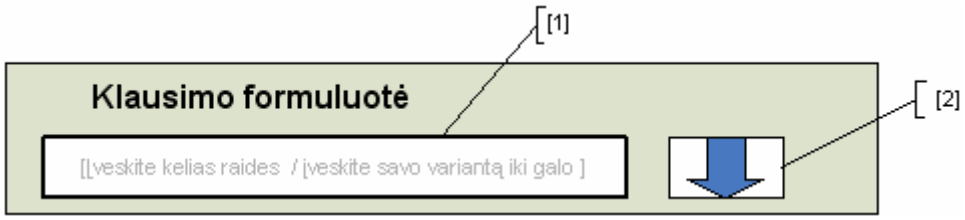
- *Kontekstas*

Mėginama pagerinti atvaizdavimo kokybę ir naudojamumą klausimų lygyje, ypač taupant vertikalią ekrano vietą (kas labai svarbu „Upės“ atveju, žr. 0).

- *Sprendimas*

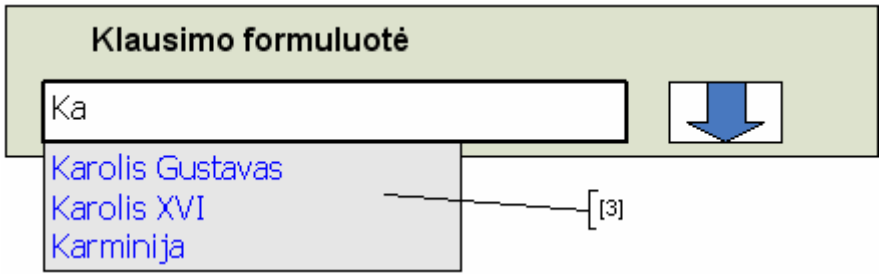
Padėtį normalizuoja kokybiškai naujo GVS elemento, galinčio visiškai pakeisti klasikinį plokščią sąrašą, įvedimas.

19 pav. Pateikta bendra schema, kur 1 simbolizuoja paprastą teksto įvedimo eilutę (angl. *textbox*), o 2 – mygtuką, perjungiantį klausimo domeną į klasikinį režimą, jeigu vartotojas nori vienu metu matyti visus galimus pasirinkimus arba paprasčiausiai nemėgsta klaviatūra pagrįsto įvedimo stiliaus.



19 pav. Bendra Žynio schema startinėje būklėje

Įvedimas pradedamas kaip elementariojo skaliarinio teksto tipo domeno atveju – pildant teksto laukelį. Įvedant vis daugiau raidžių, aktyvuojamas akceleratoriaus sąrašas, imantis reikšmes iš domeno aprašo ir pateikiantis tas, kurios neprieštarauja jau įvestoms (20 pav.). Vartotojas gali bet kuriuo momentu rodyklių klavišais arba pele pasirinkti vieną iš siūlomų variantų.

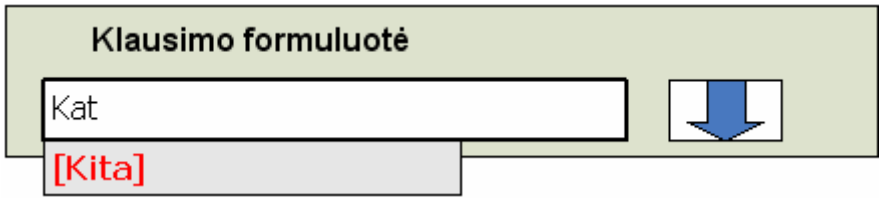


20 pav. Reikšmių akceleratorius

Vartotojui įvedus varianto fragmentą, nebeatitinkantį nė vieno iš esančių domene, galima dvejopa reakcija:

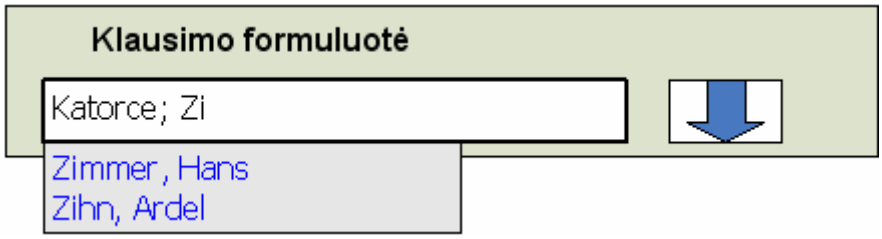
Egzistuojant variantui "Kita", visa tekstinė įvestis automatiškai nukreipiama į šį variantą (21 pav.);

"Kita" neegzistuojant, spalviniu teksto pokyčiu vartotojas informuojamas apie neleistiną įvedimą.



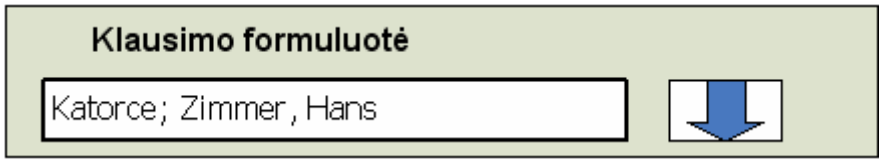
21 pav. "Kita" reikšmės įvedimas

Keli variantai gali būti įvedami, atskiriant juos kabliataškiais (22 pav.). Po kiekvieno kabliataškio akceleratorius veikia taip, kaip aprašyta bendru atveju.



22 pav. Kelių variantų parinkimas

23 pav. matomas užpildytas Žynio elementas. Bet kuriuo laiko momentu 2 mygtuko (19 pav.) paspaudimas atjungia Žynį ir sugrąžina klasikinį atvaizdavimo stilių.



23 pav. Užpildytas Žynio elementas

- *Pasekmės*

Klausimai nebeatrodo gremėzdiški ir atbaidantys; sutaupoma daug vertikalios erdvės, todėl vartotojui tenka kur kas mažiau naudotis slankjuostėmis. Klaviatūra pagrįstą įvedimą mėgstantys vartotojai dirba greičiau, o pelės mėgėjai visada gali pereiti į įprastinį režimą.

3.2.3.11. „Žodžių magija“

Klausimo formuluotė gali stipriai įtakoti atsakymų tikslumą. Gana dažnai net, atrodytų, gerai skambantys klausimai, juos pateikus kita forma (bet su identiška semantika) atsakomi nevienodai, kas gali labai sumažinti surinktų duomenų patikimumą ar, produktų valdymo sistemų atveju, sukelti brangių faktinių klaidų.

Egzistuoja tam tikros atominės taisyklės, padedančios išvengti dažnai pasitaikančių formuluotės trūkumų:

- Diskretusis domenas turi turėti elementą „Kita“. Šitaip užtikrinama, kad tikslaus pasirinkimo neturintis vartotojas vistiek galės išreikšti savo nuomonę. Veikiančioje klausimynų IS aptiktas didelis „Kita“ populiarumas signalizuoja, kad pernelyg mažas diskrečiojo domeno elementų kiekis;
- Pageidautina klausimus, neturinčius detalaus galimų atsakymų sąrašo ir naudojančius teksto tipo elementariuosius domenus, aprūpinti diskrečiojo tipo domenais su išsamia galimų atsakymų aibe [Ver+06] – arba papildyti pačią klausimo formuluotę keliais gerai parinktais pavyzdiniais atsakymo variantais (angl. *aided recall*). Vartotojai, nematantys aibės ar pavyzdžių yra linkę neužregistruoti bent dalies sau tinkančių atsakymo variantų;
- „Jūs“ turi būti vartojamas saikingai ir nedviprasmiškai [Bra+04]. Vartotojas gali turėti problemų atskirdamas tikrąjį objektą: jį (kaip asmenį) ar juos (jį ir kitus namiškius / šeimos narius);
- Klausimai apie vartotojo pažiūras turi būti kiek įmanoma trumpesni [Gan06]. Klausimai apie vartotojo elgseną – ilgi (tai padeda prisiminti detales, užmirštus aspektus);
- Laiko / objekto prasmėmis abstraktūs klausimai turi būti sukonkretinami [Gan06]. Pavyzdžiui, klasikinis klausimas „Ką dažniausiai geriate?“ neapriboja nei gėrimo tipo, nei laikotarpio, todėl iš esmės yra bevertis, nes skirtingos asmenybės jį traktuos įvairiai, o atsakymai juk lyginami / grupuojami pagal tą pačią skalę. Čia reikėtų nepamiršti žiniasklaidos formulės 5K: „Kas? Ką? Kur? Kada? Kaip?“;
- Kokio nors veiksmo dažnumo nustatymo klausimų formuluotės turi nurodyti kiek įmanoma mažesnį (nuo apklausiamo veiksmo įprasto dažnumo priklausantį [Bra+04]; pvz., beprasmiška klausti, kiek mašinų nusiperka per dieną) laikotarpį. Žmonės, gavę tokį klausimą su ilgu laikotarpiu (pvz., „Kiek kartų per mėnesį valotės dantis?“) apskaičiuoja dažnumą trumpame „natūraliame“ laikotarpyje, ir paprasčiausiai padaugina, taip gaudami dažnumą klausime nurodytame ilgame laikotarpyje. Taip gali atsirasti paprastų aritmetinių ar ekstrapoliacinių klaidų, o rezultato tikslumas visai nepadidėja;
- Būtina pasiekti balansą tarp specifinių / techninių bei abstrakčių / bendrųjų terminų [Cas+01]. Techniniai (tarptautiniai) terminai, nors konkretūs ir nedviprasmiški, gali vartotojui būti nežinomi ir / ar prastai suprantami. Kita vertus, kasdieniniai terminai masinės kultūros kontekste dažnai tampa dviprasmiškais (kai kurių ypač populiarių terminų dauguma žmonių

nebesugebėtų teisingai apibrėžti, pvz., „liberalizmas“, „modelis“), todėl klausimyno rezultatai gaunami ne to paties objekto kontekste, nėra sulygintini;

- Galimus atsakymų variantus iliustruojant atitinkamais paveikslėliais galima žymiai sumažinti atsitiktinių klaidų skaičių klausimynuose. Tam tikri atsakymų variantai apskritai sunkiai išreiškiami teksto forma, pvz. audinių raštai produktų valdymo sistemose;
- Klausimų su skaliariniais domenais formuluotėje neturėtų būti minimi matavimo vienetai; jų išsamų sąrašą (su autokonvertavimo galimybe) patartina pateikti greta domeno. Vartotojai gali skaičiuoti kitais matavimo vienetais, nei nurodyta klausime; būtinybė konvertuoti įveda aritmetinės klaidos riziką, nekalbant apie vartotojo suirzimą.

3.2.3.12. „Gausybės ragas“

Dažnai įsivaizduojama, kad trumpas diskrečiojo domeno galimų atsakymo variantų sąrašas mažina vartotojo nuovargį, didina susidomėjimą. Taip pat dažnai ši nuomonė yra klaidinga.

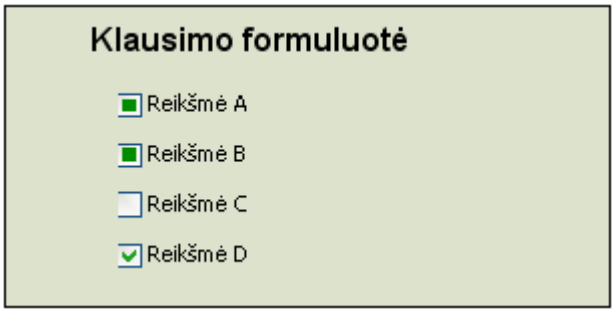
Svarbus faktorius rezultatų teisingumui pasiekti – tinkamas vartotojo nuomonės atspindėjimas, neprarandant detalių ir „nesuapvalinant“ jų iki semantiškai artimiausio egzistuojančio atsakymo varianto (suliejimas; angl. *aliasing*) [Cas+01]. Faktoriui išpildyti būtina pateikti tiek galimų atsakymo variantų, kad būtų padengti visi atvejai, išskyrus gal tik tikrai egzotinius. Pernelyg ilgas galimų atsakymų sąrašas rezultatų kokybės prasme yra kur kas mažesnė blogybė už pernelyg trumpą.

Egzistuoja keli būdai neigiamų ilgų sąrašų efektų pašalinimui:

- Pernelyg ilgas diskrečiojo domeno elementų sąrašas galėtų būti suskaidomas pagal deramą taksonomijos tipą ir pateikiamas keliais klausimais, susietais egzistenciniais ryšiais [Bra+04] (žr. 2.1.4). Aktyvavus konkrečią kategoriją, turi būti parodomas vaikinis klausimas su kategorijos turiniu kaip savo diskrečiojo domeno elementais;
- Vizuali optimizacija sąrašui pagal „Žynį“ (žr. 3.2.3.10);
- Klausimo domenui pasižymint didesniu už vienetą kardinalumu patartina įvesti trečią varianto būseną („parinktas“ – „neparinktas“ papildyti „neperžiūrėtas“), kuri simbolizuos variantą, kuriam vartotojas dar neskyrė dėmesio – ir variantas pradžioje neturės nei teigiamos, nei neigiamos reikšmės. Modifikacija, nors ir gali sukelti vartotojo susierzinimą, priverčia sąžiningai peržiūrėti visus ilgo

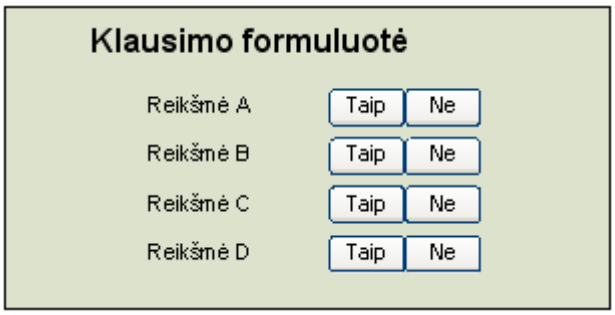
sąrašo variantus [Tou06], kai paprastai būtų peržiūrimi tik keli viršutiniai. Grafiškai vietoj populiariosios varnelės (angl. *checkbox*) reikia parūpinti trečiosios būsenos atvaizdavimą:

- trijų būsenų varnele (angl. *tristate checkbox*) (24 pav. – reikšmės A ir B dar vartotojo neaktyvuotos; C ir D – atsakytos atitinkamai teigiamai ir neigiamai);



24 pav. Ilgas variantų sąrašas su trijų būsenų varnelėmis

- mygtukais „pritariu“ / „nepritariu“ (25 pav.).



25 pav. Ilgas variantų sąrašas su mygtukais

Vienos ar kelių nurodytų alternatyvų taikymas sumažina rezultatų paklaidą.

3.2.3.13. „Tolydi nuomonė“

Asmenų nuomonės nustatymo klausimuose pastebimas pernelyg didelis diskretumo lygis, t.y. egzistuojančių galimų atsakymo variantų neadekvatumas esamai asmens nuomonei. Net ir naudojant aptakius daugybinius variantus (pvz., „labai“, „vidutiniškai“, „mažai“) vartotojai dažnai turi pasirinkti vieną jų viso labo kaip kompromiso rezultatą. Taip nuomonės „apkarpomos“, neleistinai supaprastinamos. Be visa ko, tie patys žodiniai galimi atsakymo

variantai įvairiems žmonėms sužadina nevienodo stiprumo asociacijas, taip išbalansuojant globalią vertinimo skalę.

Tokiais atvejais gerų rezultatų pasiekiama pasitelkiant „analoginį klausimą“ – t.y. klausimą, kaip atsakymą priimantį skaitinį įvertį slankiojo kablelio elementariojo skaliarinio domeno pagrindu (grafiškai tai puikiai atvaizduojama tempikliu (angl. *slider*)). Alternatyviai galima naudoti ir diskretųjį domeną su elementais, atstojančiais smulkius skaitmeninius įverčius, paprastai nuo 1 iki 10 (rėžiai – susitarimo reikalas).

Žmonės kur kas lengviau ir tiksliau išreiškia savo nuomonę skaitinio įverčio forma nei klasikine [Gan06] (aprašyta problemos apibūdinime). Užtenka nurodyti viso labo du rėžių žodinius įverčius, kaip taisyklė – kardinaliai priešingos semantikos. Patartina nenaudoti subjektyvių terminų (pvz., „labai“, „prastai“), o detalizuoti smulkiau, nepaliekant erdvės nereikalingoms interpretacijoms.

3.2.3.14. „Hibridinė nuomonė“

Kartais skaitinis atsakymo pobūdis (žr. „Tolydi nuomonė“, 3.2.3.13) būna pernelyg primityvus ir nelankstus siekiamam tikslui. Atsakymai į kai kuriuos klausimus sunkiai išreiškiami vien tik skaitiniu įverčiu.

Sudėtingesnė sprendimo forma apima klausimo išskaidymą į du, rodomus vienas po kito. Jie neprivalo būti susieti priežastiniais ryšiais, gali būti rodomi vienu metu.

- Pirmasis klausimas formuluojamas kaip įprasta, bet turi diskretinį domeną su tekstiniais variantų apibūdinimais (panašiai kaip „Tolydi nuomonė“ problemos konstatavime, tik apibūdinimai labiau specifiški);
- Antrasis klausimas išaiškina vartotojo įsitikinimo, išreikšto pirmo klausimo atsakyme, absoliutinį stiprumą. Priima tik skaitinius atsakymus (taip, kaip „Tolydi nuomonė“ sprendime);

Kartu šie klausimai įgalina nuomonę išreikšti lanksčiau nei „Tolydi nuomonė“, apimti didesnį variantų spektrą, neprarandant galimybės analoginiu būdu parinkti tikslią diskrečiojo varianto įtaką.

3.2.3.15. „Nuomonės prizmė“

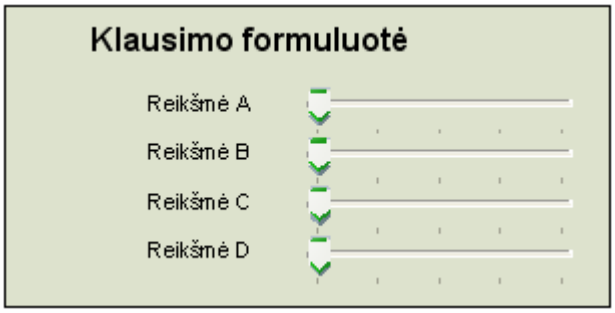
Tam tikrais sudėtingais atvejais užduodamas klausimas rezonuoja su vartotojo pažiūromis taip, kad į jį negalima atsakyti vienareikšmiškai, pasirenkant diskretų variantą ir /

ar analoginį įtakos laipsnį (žr. „Tolydi nuomonė“, 3.2.3.13 ir „Hibridinė nuomonė“, 3.2.3.14).Numatant tokią problemą dideliame vartotojų rate, būtina pateikti būdą jai apeiti.

Norimų savybių suteikia daugiaklausiminis konstruktas, leidžiantis parinkti kelis atsakymus norimais svoriniais koeficientais.

Struktūra gali būti dvejopa:

- Tėvinis klausimas su diskrečiuoju domenu bei egzistenciniais ryšiais susietais vaikiniais tolydžiais (žr. 3.2.3.13) klausimais. Pasirinkus kažkurį tėvinio klausimo atsakymo variantą, aktyvuojamas atitinkamas vaikinis klausimas (pvz., pasirinkus variantą „Labai mėgstu“, vaikinis klausimas galėtų vadintis „Kiek labai mėgstate?“), kuriame reikia nurodyti analoginį teiginio stiprį;
- Tėvinis klausimas neturi domeno; iš karto rodoma tiek vaikinių klausimų (su atitinkamais pavadinimais), kiek buvo tėvinio klausimo galimų atsakymo variantų (26 pav.).



26 pav. Daugybinis tolydinis atsakymų parinkimas

- Kiekvienas vaikinis klausimas pradžiai nustatomas ties minimalia reikšme; vartotojas gali jas didinti kiek norėdamas.

Pirmasis variantas taupo vertikalią puslapio vietą bei mažina vizualią maišatį, tačiau reikalauja daugiau veiksmų ir gali būti gluminantis be papildomų paaiškinimų. Antrasis variantas labai paprastas naudoti, bet gremėzdiškas.

Sprendimas įgalina užduoti miglotosios logikos stiliaus klausimus, neturinčius vienmačių atsakymų, leidžia vartotojui pasirinkti ne vien tik priešingus (angl. *mutually exclusive*) atsakymų variantus.

3.2.3.16. „Nejauki tyla“

Įvairių tipų klausimynuose esantys klausimai, įvedami klausimynų IS projektuotojų, turi tendenciją išlaikyti kūrėjų asmeninių bei vertybinių požiūrių fragmentų. Klausimynų IS

vartotojai gali nepritarti tokiems fragmentams, šitaip pajusdami (nesąmoningą) priešiškumą klausimų sudarytojams ir neracionaliai iškreipdami atsakymus į „įžeidžiančius“ klausimus.

Problema apima daug įvairių atvejų:

- Marketingo klausimyne: „Ką manote apie mūsų tobulą produktą XX?“ (tobulybės prezumpcija; vartotojas gali įvesti blogą atsiliepimą „iš principo“);
- Visuomenės nuomonės tyrimo klausimyne: „Ar „Mažeikių Naftą“ išparceliavę konservatoriai turi būti teisiami?“ (politinių pažiūrų įžeidimas; nepagrįstas kaltinimas, galintis sulaukti neigiamo atsakymo vien dėl formuluotės);
- Produkto valdymo klausimyne: „Kokia kėdės kaina?“; galimi atsakymo variantai: „runkeliui neįkandama“, „> 500 lt.“, „> 100 lt.“ (visuomeninės padėties priminimas);

Klausimų formuluotės turi būti peržiūrimos [Ver+06]; pastebėti asmeniškumai ir semantiškai / pragmatiškai miglotos vietos – pašalinti. Ypač svarbu aptikti ir pakoreguoti klausimus, vien savo formuluote įtakojančius vartotojo apsisprendimą.

Paprastai žodžiai, kuriuos pašalinus nepakeičia klausimo esmės, neturi būti paliekami, pvz. „tobulas“ probleminiame pavyzdyje (yra išimčių, žr. „Žodžių magiją“, 3.2.3.11). Slengas, pretenzingi veiksmažodžiai turi būti pašalinti / supaprastinti (žr. kitus probleminius pavyzdžius).

3.2.3.17. „Mono arba stereo“

Dėsninga naudoti vienpolinius (angl. *unipolar*; ieškantys vartotojo nuomonės apie vieną objektą) klausimus, tačiau pastebima, kad, uždavus kitą klausimą apie objekto analogą / konkurentą, pirmojo „taip“ bei antrojo „ne“ tikimybių suma nesudaro 100%.

Netgi perfrazuojant klausimus dvipoliniu (angl. *bipolar*, pateikiantis vartotojui pasirinkti iš dviejų kardinaliai priešingų objektų) variantu, pastebima rezultatų variacija kaitaliojant vieną objektą poroje.

Nepatyrusiems klausimynų projektuotojams dažnai neaiški abiejų tipų klausimų naudojimo specifika, trūkumai ir privalumai.

Vienpoliai klausimai [Bra+04] taikytini tais atvejais, kai aptariamas objektas yra pragmatiškai primityvus, neturi vidinės struktūros ar didelio alternatyvų rato. Pavyzdžiui, žodžio laisvė ar XY dantų pasta. Priešingu atveju iškyla rizika, kad variantą „ne / nemėgstu / neparemiu“ pasirinks tiek išties priešingai nusiteikę, tiek ir nuomonės laikinai neturintys / abejingi / nekantrūs / dvilypės nuomonės vartotojai. Iš esmės neigiamas atsakymas sukelia reikšmių suliejimą. Turint tai omenyje, galima vienpolius klausimus taikyti ir kompleksiniams

objektams, norint sužinoti ar vartotojas juos mėgsta – sąmoningai atmetant dalį „... ar nemėgsta“, kurios tiksliai išmatuoti nepavyks.

Pakeičiant vienpolį klausimą (pvz., „Ar remiate mokamą mokslą?“) dviem vienpoliais klausimais su skirtinga apklausiama reakcija, bet vienodu objektu („Ar remiate mokamą mokslą?“, „Ar nepalaikote mokamo mokslo?“) dar labiau pagilina 100% problemą. Kuo siauresne semantika pasižymi apklausiamos reakcijos – tuo mažiau juos pasirenkančių vartotojų [Tou06]. Patenkinamas sprendimas pasiekiamas atmetant primityvių (todėl, atrodytų, didinančių tikslumą) galimų atsakymų variantų „taip / ne“ naudojimą ir į jų vietą perkeliant priešingąsias reakcijas („Koks Jūsų požiūris į mokamą mokslą?“ – „Palaikau“ / „Nepalaikau“). Atsiranda suliejimo problema, tačiau jos efektai kur kas silpnesni nei 100% problemos.

Vienpolių klausimų dilemą dar galima spręsti / pagerinti naudojant „Tolydi nuomonė“ (žr. 3.2.3.13), taip sušvelninant 100% problemą.

Dvipoliai klausimai struktūriškai yra palyginamieji, todėl nukenčia nuo konkrečios objektų kombinacijos ypatybių nevienodumo. Lieka nežinomas tiek proporcinis vieno objekto pranašumas prieš kitą, tiek ir absoliutus abiejų objektų įvertis. Be to, prarandama galimybė pastebėti koreliacijas tarp atskirose porose lyginamų objektų.

Tačiau nereikia pamiršti, kad dvipoliai klausimai gali būti naudojami tranzityvumo dėsnio taikymui lyginant dideles objektų aibes, bet neapkraunant vartotojo vienu klausimu su didžiule variantų aibe (kas sukelia problemų, žr. „Žodžių magiją“, 3.2.3.11). Kombinuojant su „Nuomonės prizmė“ (žr. 3.2.3.15) galima iki minimumo sumažinti suliejimo efektą ir gana tiksliai sužinoti proporcinius bei absoliučius įverčius.

3.2.3.18. „Paralelizmas“

Pradedantieji klausimynų projektuotojai dažnai vietos taupymo dėlei ar tiesiog dėl nežinojimo klausimams suteikia dvimačius (angl. *double-barreled*; apima du susijusius, bet nevienodiems objektams priklausančius teiginius) atsakymų variantus. Retais atvejais klausimuose, turinčiuose diskrečiuosius domenus su daugiau nei dviem variantais pasitaiko netgi pusantrinių (angl. *one-and-a-half barreled*; vienas iš vidurinių variantų yra dvimatis) variantų. Būtina išaiškinti problemos mechaniką ir jos išvengimo bei korekcijų būdus.

Aprašyti atvejai atsiranda netikėtai ir natūraliai, mėginant plačiau paaiškinti variantus. Neigiamas aspektas – kritiškai ryškios suliejimo paklaidos (abu atvejai), reikšmių prasmių pakeitimas (pusantrinis atvejis).

Perdėtas dvimačio varianto pavyzdys (dėl politinio nekorektiškumo jam taikytina ir „Nejauki tyla“, žr. 3.2.3.16): „Būsimuose prezidento rinkimuose pasirinksite:“ – „Petraitį ir elitą“, „Jonaitį ir liaudį“. Atsakymo variantai sulieja dviejų objektų – prezidento asmuo ir socialinė orientacija, - galimas reikšmes. Asmuo, kurio įsitikinimų vektorius neatitinka nė vieno iš esamų variantų, įvesdamas patirs net 50% suliejimo paklaidą.

Realus pusantrinio varianto pavyzdys: „Kokia Jūsų nuomonė apie šį stalą?“ – „Puiki“, „Vidutiniška“, „Neutrali, reikėtų didesnio katalogo“, „Prasta“, „Nekokia“. Nuo vidurinio varianto (kuris detalizuojamas geresniam termino „neutralus“ supratimui) įvedamas antras su klausimu nesusijęs objektas: marketingo kokybė; taip antriniu kontekstu „užteršiami“ visi likę variantai. Suglumintas vartotojas variantą „Prasta“ gali pasirinkti tiek galvodamas apie stalą, tiek ir apie katalogo dydį.

Kaip matyti iš pavyzdžių, keliamačių variantų reikia vengti [Bra+04] jau projektavimo stadijoje, labai atsargiai pasirenkant variantų detalizacijų formuluotes, kad neįvesti papildomų objektų. Paaiškinimus patartina talpinti iššokančiuose komentaruose (angl. *tooltip*); taip lengviau atskirti paaiškinimą nuo integralios varianto formuluotės dalies.

3.2.3.19. „Eiliškumas“

Priklausomai nuo rikiavimo tvarkos tiek klausimų, tiek galimų atsakymų variantų lygyje didelėse užpildytų klausimynų aibėse gaunami statistiškai skirtingi duomenys.

Problema sprendžiama identifikuojant šiuos dėsningus atvejus [Tho+06]:

- Pagrindinio ir išvestinio klausimų dilema: taikant šį populiarų klausimų formavimo būdą (pagrindinis klausimas nustato bendrą vartotojo polinkį; atitinkamas išvestinis klausimas apima tik dalį pagrindinio klausimo objekto) būtina išvestinį klausimą pateikti už pagrindinio. Priešingu atveju atsiranda koreliacija: priklausomai nuo išvestinio klausimo sukeliamų asociacijų, pasikeičia žemiau einančio pagrindinio klausimo įvertis;
- Ilgas (vizualiai: netelpantis į puslapį) diskrečiojo domeno elementų sąrašas: dėl kontrasto nebuvimo, viduryje esantys variantai aktyvuojami rečiau nei viršutiniai / apatiniai. Reikėtų, kad sąrašas kiekvieną kartą būti pateikiamas atsitiktine tvarka; jos taikymas didelėje vartotojų statistinėje aibėje neutralizuoja šį defektą;
- Daugybiniai vienpoliniai klausimai su tuo pačiu objektu („Mono arba stereo“ sąlyga, žr. 3.2.3.17): jei keli tokie klausimai yra eilėje (pvz., A, B, C) ir vartotojo nuomonė atitinka A variantą, padidėja tikimybė teigiamai (ar bent

labiau teigiamai tolydaus klausimo atveju, žr. 3.2.3.13) atsakyti ir į eilėje žemiau esančius klausimus B bei C. Šį rezultatus iškraipantį efektą galima minimizuoti pirmoje vietoje iškeliant tikėtinai mažiausiai populiarią reikšmę.

3.2.3.20. „Viskas arba nieko“

Pasirenkamuosiuose klausimuose su diskrečiaisiais domenais, susidedančiais iš daug variantų, pastebima tendencija žymėti daugiau (arba mažiau, jeigu prašoma išvardinti socialiai neigiamus veiksnius) atsakymų nei reikėtų. Tai sukeliama vartotojo nenoro pasirodyti blogesniam už kitus, žymėti neproporcingai (jo nuomone) mažą kiekį atsakymo variantų didelėje (numanomai - išsamioje) jų aibėje.

Simptomai pastebimi tokiuose klausimuose kaip: „Kokias sporto šakas kultivuojate?“, kaip galimus atsakymo variantus pateikiant sporto šakų pavadinimus.

Disbalansas sumažinamas išskaidant klausimą į du mažesnius [Tou06]. Pirmasis paklausia, ar vartotojas apskritai turi bent vieną jam tinkantį atsakymo variantą (pavyzdyje: „Ar apskritai užsiimate sportu?“, galimi atsakymo variantai: „taip“, „ne“), ir pateikiant nuo atsakymo „taip“ priklausantį (egzistencinis ryšys; žr. 2.1.4) klausimą, prašantį išvardinti tinkamus variantus (pavyzdyje: „Jeigu užsiimate – kokiomis sporto šakomis?“), nustatant atsakymo kardinalumą didesnį ar lygų vienetui.

3.2.3.21. „Vidutinybė“

Klausimynuose (ne produktų valdymo tipo) dažnai pasitaiko atvejų, kai norima išgauti vartotojo nuomonę tam tikrais socialiai nepriimtinais ir / ar labai asmeniniais klausimais. Nors klausimynų IS iš esmės nereikalauja jokios „žmogaus – žmogaus“ sąveikos, klasikinė aksioma „asmuo (nesąmoningai) nori, kad aplinkiniai apie jį galvotų gerai“ gali įtakoti atsakymo teisingumą, kas statistiškai iškreipia klausimyno korektiškumą. IS vartotojas nenori, kad kas nors žinotų – ir dar negrįžtamai užregistruotų tai IS duomenų bankuose, - jog vartotojas pasielgė neetiškai ar net nusikalstamai [Gal+06] [Bra+04]. Sunku prisipažinti net ir nepažįstamam žmogui, atsakymus įvedančiam nešiojamame kompiuteryje statistikos tikslais.

Problema gali būti iliustruojama pavyzdžiu: „Ar vogėte iš savo darbovietės?“.

Patenkinamas sprendimas gali būti pasiekiamas naikinant nusiteikimą, jog „socialiai nusidėjęs“ vartotojas yra žemiau daugumos, išskirtinis neigiama prasme. Turi būti akcentuojamas (nors ir faktams esant priešingiems) klausimo objekto paplitimas, neišskirtinumas visuomeninės vidutinybės kontekste [Ver+06]. Galima išskirti penkias

alternatyvias formuluotės pertvarkymo versijas, taikytinas priklausomai nuo situacijos ar konteksto [Bra+04] [Gal+06]:

- „Kaip visi“: potencialiai neigiamas socialinis klausimo turinys pateikiamas kaip visuotinai priimtinas. Pagal aukščiau pateiktą pavyzdį transformacijos išeiga yra: „Kaip žinia, šiais laikais daug žmonių vagia iš savo darboviečių. Kaip elgiatės Jūs?“;
- „Kaip kiti“: nešvelninant formuluotės, klausime panaikinamas asmeniškumo faktorius, traktuojant vartotoją kaip eilinį, abstraktų žmonijos narį. Pavyzdžio transformacija: „Ar pažįstate žmonių, ką nors pavogusių iš savo darboviečių?“, galimus atsakymus pateikiant: „taip“, „taip, save“, „ne“;
- „Reliatyvumas“: pakeisti klausimo akcentą, „paskandinant“ norimą sužinoti faktą dar labiau socialiai nepriimtinų alternatyvų aibėje, taip sukeliant vartotojui dirbtinį palengvėjimo jausmą. Pavyzdį reikėtų suabstraktinti iki „Kokius veiksmus Jums yra tekę atlikti?“, pateikiant galimus atsakymus: „nužudyti tėvus“, „susprogdinti dangoraižį“, „vogti iš darbovietės“;
- „Visi klysta“: dirbtinai sumažinti klausimo aktualumą įvedant atsainumą, akcentuojant praeitį. Pavyzdys transformuojamas į: „Ar kada nors, nors vieną kartą, Jums teko vogti iš darbovietės?“;
- „Kaltas avansu“: išvaduoti vartotoją nuo tiesioginio prisipažinimo avansu jį laikant „kaltu“ – ir suteikti galimybę tą prezumpciją paneigti. Pavyzdys galėtų būti pakeistas taip: „Kiek kartų vogėte iš darbovietės?“; galimi atsakymo variantai: „nė vieno“, „vieną“, „daugiau nei vieną“;
- „Atbulinė psichologija“: vartotojui „tarp kitko“ pateikiamos priežastys (nebūtinai objektyvios ar įrodytos: „baltas melas“), dėl kurių klausime figūruojantis veiksmas tampa mažiau socialiai nepriimtinas. Šitaip kompensuojamas bandos instinktas. Pavyzdžio modifikacija gali skambėti taip: „Apklausomis įrodyta, kad darbovietės daiktų pasisavinimas sustiprina ryšį tarp darbuotojų, be to – darbovietės turtas dažniausiai būna apdraustas. Kokia Jūsų patirtis šia tema?“.

Įvertinti (ir galbūt algoritmiškai kompensuoti, įvedant patikimumo koeficientą) atsakymų tikrumą socialinio priimtinumo kontekste padeda savotiškų metaklausimų [Cas+01] (su skaliariniais domenais; diskretieji nelabai patogūs koeficientams skaičiuoti) pridėjimas klausimyno gale, pvz.: „Kokiu lygiu klausimai privertė Jus pasijausti nepatogiai?“.

Šablono pritaikymo rezultatai – mažiau iškreiptas klausimyno teisingumas.

4. Praktinis klausimynų IS kūrimas taikant projektavimo šablonų kalbą

Projektavimo šablonų kalba laikoma užbaigta tik jei gali būti sėkmingai taikoma praktikoje ir srities ekspertai pripažįsta jos aktualumą bei teisingumą. Dėl konfliktuojančių standartų ir vieningos projektavimo šablonų kalbų bendruomenės nebuvimo daugelis kalbų naudojamos tokios, kokios yra. Praktiškumo įrodymas dažnai būna vienintelis vertės kriterijus.

4.1. Diegimas praktikoje

Šiame darbe pateikta projektavimo šablonų kalba naudojama UAB „Baltijos IT ekspertų grupė“ naujų darbuotojų apmokymui. Konkrečiai – programuosiančių klausimynų sferoje supažindinimui su dalykine sritimi ir patirtimi pagrįstais projektiniais IS bei turinio sprendimais.

Projektavimo šablonų kalba buvo palaipsniui kuriama nuo 2004 metų rugpjūčio mėnesio. Naudodamas įgytą patirtį autorius užsienio klientams sukūrė dvi skirtingų charakteristikų klausimynų IS – darbe žymimos sutartiniais pavadinimais „Mažylis“ ir „Storulis“, - kartu realizuojančias nemažą dalį kalboje esančių projektavimo šablonų.

4.2. Projektavimo šablonų panaudojimo lygis

Kadangi pateikti projektavimo šablonai yra ganėtinai aukšto lygio, o kalba – modulinė ir leidžianti pasirinkti norimą realizacijos gylį, pavyzdinėse klausimynų IS realizuojami ne visi projektavimo šablonai (6 lent.). Ypač išsiskiria psichologiniai, kurių panaudojimo lygį sunku tiksliai nustatyti, nes teisingos klausimų formuluotės gali būti tiek klausimynų projektuotojo asmeninės patirties, tiek ir projektavimo šablonų taikymo rezultatas.

6 lentelė. Projektavimo šablonų realizavimas pavyzdinėse klausimynų IS

Projektavimo šablonas	**Sukurtos IS**	
	„Mažylis“	**„Storulis“**
Medžiai be lapų	+	+
Pilkasis kardinolas	+	+
Magnus Rex	+	+
Klaustuko galia	+	+

Projektavimo šablonas	Sukurtos IS	
	„Mažylis“	**„Storulis“**
Kardomoji priemonė	+	-
Projekcija	+	+
Vienišas horizontas	+	-
Savivalda	-	+
Žingsnis po žingsnio	-	+/-
Žodžių magija	-	+
Gausybės ragas	+/-	+/-
Tolydi nuomonė	+	-
Hibridinė nuomonė	-	-
Nuomonės prizmė	-	-
Nejauki tyla	-	+
Mono arba stereo	+	+
Paralelizmas	+	+
Eiliškumas	-	+
Viskas arba nieko	+/-	-
Vidutinybė	+	+

Kaip matyti, didžioji dalis projektavimo šablonų realizuojama bent dalinai; toliau aptariama pavyzdinių IS specifika ir detalizuojama 6 lentelė (žr. 4.3).

4.3. Pavyzdinės klausimynų IS

Kaip pavyzdžiai pateikiamos klausimynų IS yra ganėtinai skirtingos realizacijos prasme, tačiau panaudoja svarbiausius programinius / vartotojo sąsajos projektavimo šablonus. Dėl architektūros ir platformos skirtumų projektavimo šablonai dažniausiai realizuojami su išlygomis ir nevienodai.

Bendras pavyzdinių klausimynų IS tarpusavio palyginimas pateikiamas 7 lent.:

7 lentelė. Pavyzdinių klausimynų IS savybių tarpusavio palyginimas

Kriterijus	**Sukurtos IS**	
	„Mažylis“	**„Storulis“**
„Kliento-serverio“ architektūros potipis	Mišrus klientas	Storas klientas (*thick client*)
Veikimo terpė	Žiniatinklis (*Gecko*)	*Windows NT* OS (*Windows Forms*)
Realizacijos platforma	• *Mozilla Firefox* • *AJAX* • *XML* • *Javascript* • *PHP 5* • *MySQL 5*	• *.NET 2.0: C#* • *Visual C++ 7* • *COM* • *SQL Server 2005*
Programavimo metodika	• Procedūrinė • Objektinė (nežymiai)	Objektinė
Lygis	*Web 2.0*	*Enterprise*
Kūrimo įrankiai	• *Macromedia Dreamweaver MX 2004* • *UltraEdit* • *Venkman* • *Web Developer*	• *Microsoft Visual Studio 2005 Enterprise* • *SQL Server Management Studio*
Ryšio protokolas	*HTTP*	*TCP/IP*
Veikimo greitis	Mažas (daugiausia interpretuojamas kodas)	Didelis (*JIT*)
Mobilumas	Didelis (žiniatinklis, *Mozilla Firefox* naršyklė)	Mažas (*.NET* / *Windows*)
Resursų poreikis	Didelis (greitam interpretavimui pasiekti)	Vidutiniškas (lėtoka grafinė sąsaja, *managed* architektūra)

Kaip matyti, aplinka ir netgi kūrimo principai abiem atvejais yra ganėtinai nepanašūs, todėl projektavimo šablonų realizavimas įgauna daugiau svorio.

4.3.1. „Mažylis“

„Mažylis“ – koncepcinė naujos kartos IS, akcentuojanti *Web 2.0* paradigmos mobilumą ir patogumą naudoti. Nors ši IS veikia žiniatinklio naršyklės aplinkoje, bet vartotojo sąsajos galimybėmis ir greičiu prilygsta klasikinėms programoms. Dėl standartinių naršyklės *Mozilla Firefox* galimybių naudojimo nereikalauja jokių papildomų įskiepių, taigi, yra parengta naudojimui praktiškai bet kuriame kompiuteryje.

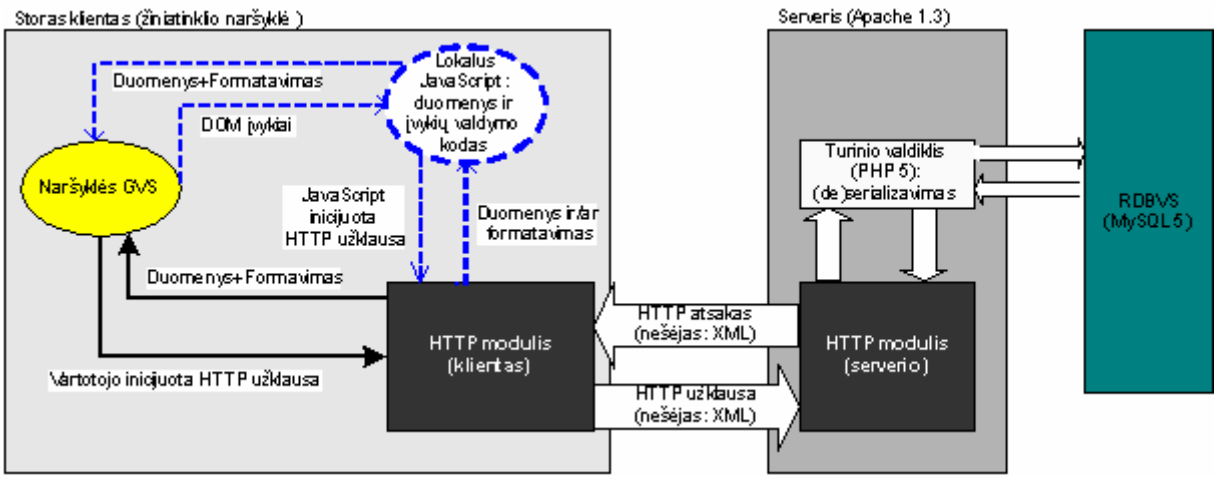
Vieninteliai „Mažylio“ trūkumai – populiariausios, bet techniškai atsilikusios naršyklės *Microsoft Internet Explorer* nepalaikymas, sąlyginai lėtas vykdymo greitis (kadangi *Javascript* kodas yra interpretuojamas), sunkus ir lėtas kodo rašymas (manipuliavimas dviem programavimo kalbomis kliente ir serveryje, objektinės ir procedūrinės metodologijų priešprieša, *Javascript IDE* nebuvimas).

„Storo kliento“ žiniatinklio naršyklėje realizacijos ypatumai aprašomi prie darbo pridėtame (žr. priedą „Straipsnis“, 10.1) straipsnyje, skelbtame konferencijoje „Informacinės Technologijos 2006“.

4.3.1.1. “Medžiai be lapų” realizacija

Nors ir pagrįsta žiniatinklio technologijomis, aptariama klausimynų IS yra sudaryta kaip „storo kliento“ aplikacija (27 pav.):

- Javascript interpretatorius naršyklėje laiko masyvus, sudarančius KOM;
- Grafinė vartotojo sąsaja remiasi dinamine puslapį sudarančių elementų manipuliacija be jokių puslapio perkrovimų. Naudojamas įvykių (angl. *events*) modelis sąveikai tarp Prezentacijos ir Valdiklio atitinka KVM ir ryšius su KOM.
- Statinė klausimynų aprašo dalis (klausimai, ryšiai tarp jų...) saugomi meta DB (priedas „“Mažylio“ klausimynų IS KDB schema“, žr. 10.2). Ji „pagal pareikalavimą“ įkraunama į Javascript interpretatoriaus atmintį kaip KOM dalis;
- Dinaminė klausimynų egzempliorių informacija (įvestos reikšmės) yra saugoma standartizuotos formos DB, labai primenančioje priede „“Storulio“ RDB fragmentas“ (žr. 10.4) pateiktą fragmentą.



27 pav. „Mažylio“ architektūra

Visos kartu sudėtinės IS dalys atitinka MVC paradigmą.

4.3.1.2.“Pilkasis kardinolas” ir “Magnus Rex” realizacijos

Nors *Javascript* oficialiai palaiko *OOP*, faktiškai viską supaprastina iki masyvų su tekstinių eilučių indeksais naudojimo objektams atvaizduoti ir paprastų funkcijų prijungimo prie objektų kaip kintamųjų (atitinka *C++ function pointer*). Polimorfizmą galima realizuoti papildomo kodu. Visas *Javascript* pseudo *OOP* palaikymas [Žil06] interpretavimo metu užima papildomai procesoriaus laiko ir neduoda įprastinio *OOP* privalumų, todėl (beveik) nenaudojamas – išvengiant papildomo interpretacijos lygio, iškart taikomas hierarchinių duomenų struktūrų aprašymas masyvais su asociatyviniais indeksais.

Žemiau pateikiamas duomenų struktūrų, atitinkančių „Pilkasis kardinolas“ (žr. 3.2.3.2) ir „Magnus Rex“ (žr. 3.2.3.3) rekomendacijas.

```
var GEData = new Array();
//    *name*
//    *values*:
//          (ARR[property_id] -> OBJ:
//                *values*
//                      (ARR -> OBJ:
//                            *lval*
//                            *rval* (o)
//                      )
//    )
//    *treeData*:
//    (ARR[property_id; hierarchical] -> OBJ:
//                *level*
//                *count*
//                *id*
//                *cardState*
//                *childLevel*
//                *isUpdated*
```

```
//                *children*:
//                      {ARR -> OBJ: {treeData_element}}
//                *deferredChildren*
//                *noPhysicalRef*
//                *dataErrorActive*
//    )
//  *treeDataRef*:
//          (ARR: (OBJ:[treeData]))
//    *SERStateArray*:
//          (ARR: boolean)
//    *SERValArray*:
//          (ARR: boolean)
//    *children*:
//          (ARR -> {GEData_element})
var valArray = new Array()
// (OBJ:lval|rval)
var SERValArray = new Array();
var SERStateArray = new Array();
var SERMasterDepArray = new Array();
var SERSlaveDepArray = new Array();
```

4.3.1.3.“Klaustuko galia” realizacija

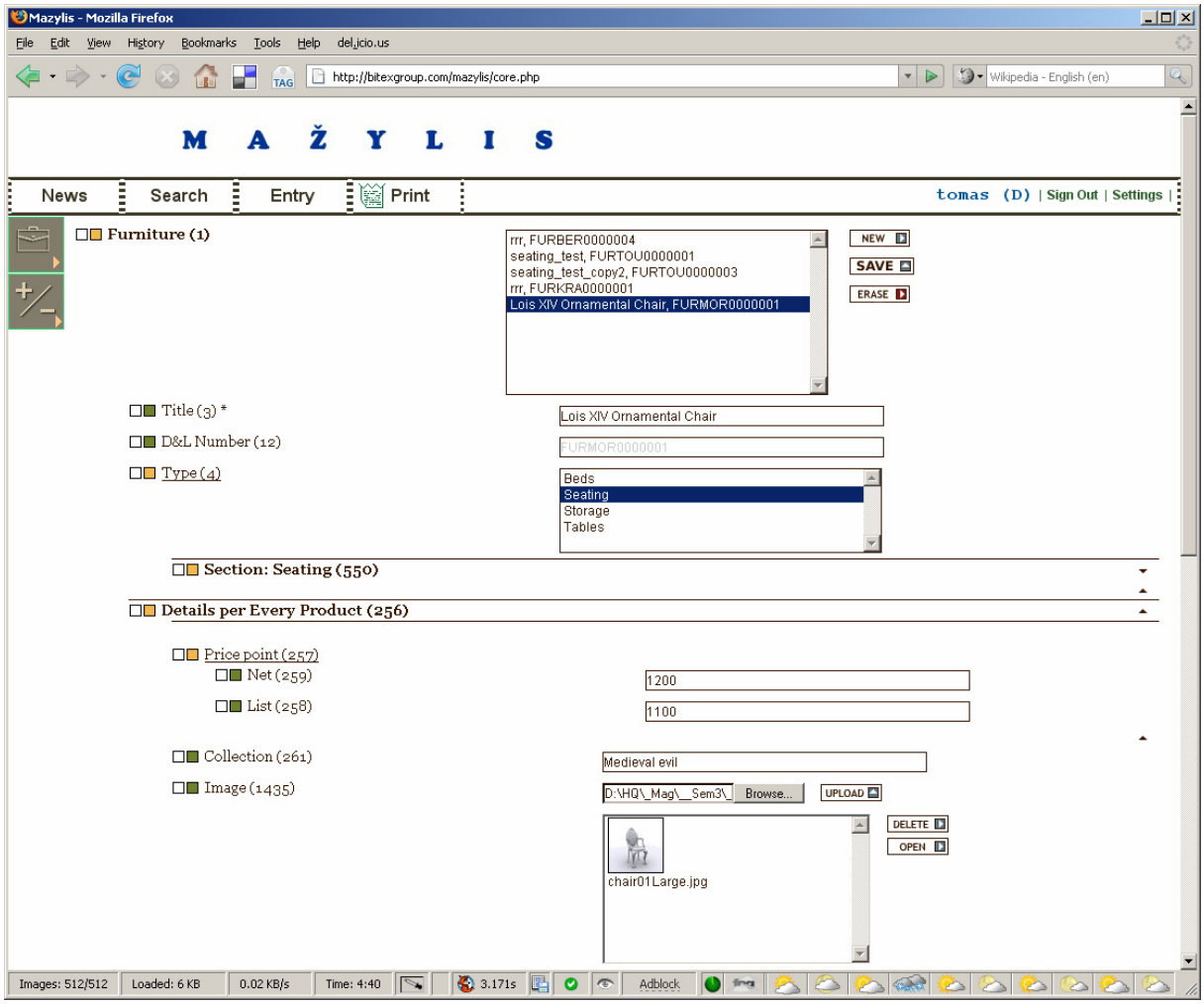
Individualių klausimų statinė struktūra (pagal „Klaustuko galia“, žr. 3.2.3.4) pateikiama žemiau:

```
var metaProps = new Array();
//    *id*
//    *name*
//    *enableCondition/NULL* - in CC format
//    *type/NULL*
//    *isMandatory/NULL*
//    *minOccurs/NULL*
//    *maxOccurs/NULL*
//    *defaultChildID*
//    *skipChildrenRendering*
//    *hasDomain/NULL*
//    *domainTypeName/NULL* - STRING, INTEGER, FLOAT, DATETIME
//    *domainFriendlyTypeName*
//    *domainFormatExpr/NULL*
//    *domainFormatExpl/NULL* - requires domainFormatExpr
//    *domainUnitMainName*
//    *domainUnitAlternativeArray/EMPTY*: OBJ:displayName|conversionRate
//    *domainRangeArray/EMPTY*
//    *domainHasAssocRoots*
//    *domainValueArray/EMPTY*: OBJ:id|content|associatedRootID
//    *domainValueIDRefArray/EMPTY*: ID -> OBJ (domainValue)
//    *associatedEntity/NULL*
//    *associationType/NULL*
//    *children/EMPTY*: Array of ID (integer)

var itemData = new Array();
// OBJ:id|value|type
```

4.3.1.4.“Vienišas horizontas” realizacija

Pateikimas vartotojui realizuojamas pagal „Vienišas horizontas“ (žr. 3.2.3.7) (28 pav.):



28 pav. „Mažylyje“ atidarytas klausimynas

4.3.1.5.“Kardomoji priemonė” realizacija

Egzistenciniai apribojimai (žr. 2.1.4) klausimams pagal „Kardomoji priemonė“ (žr. 3.2.3.5) yra suformuojami serveryje *PHP* kalba iš KDB saugomų elementų ir naudojami kaip tekstinė eilutė interpretatoriuje. KDB schema (priedas „“Mažylio“ klausimynų IS KDB schema“, žr. 10.2).

4.3.2. „Storulis“

„Storulis“ – klasikinė *enterprise* lygio IS *.NET* platformoje. Sukurta naudoti Windows aplinkoje, pasižymi ypač didelių klausimynų palaikymu ir paieškos galimybėmis. Nenaudoja tarpinio (angl. *middleware*) serverio tarp kliento ir duomenų bazės.

Nors egzistenciniai apribojimai ir nerealizuoti, sistema palaiko jų struktūrą, tačiau kitokia nei „Kardomoji priemonė“ (žr. 3.2.3.5) forma (apribojimai iš karto saugomi vientisa tekstine eilute, kuri vykdymo metu įterpiama į iš anksto parengtą kodo apvalkalą ir dinamiškai sukompiliuojama naudojant *reflection*).

4.3.2.1. “Medžiai be lapų” realizacija

Ši klausimynų IS projektavimo šablone nurodytus komponentus realizuoja taip:

- KVM sutapatinamas su klausimų / domenų atvaizdavimo klasėmis, realizuojančiomis nurodytus interfeisus ir abstrahuojančiomis grafinę dalį nuo komandų sąsajos. Pavyzdžiui, interfeiso komanda clearValues() panaikina įvestus atsakymus į konkretų klausimą – nepriklausomai nuo klausimo tipo ir jo atvaizdavimo formos;
- KOM realizuojamas statine klase, kurios atributai – masyvai, indeksuojantys klausimynus aprašančių klasių (priedas “Storulio” klausimynų IS metaklasės“, žr. 10.3) objektus. Savaime suprantama, objektų viduje egzistuoja nuorodos į susijusius objektus, bet kartais tiesioginis kreipinys per vienmatį masyvą yra ekonomiškesnis;
- KDB (metaDB) nedaug skiriasi nuo „Mažylio“ atvejo, ją galima rasti priede „“Mažylio“ klausimynų IS KDB schema“, žr. 10.2;
- RDB klausimynų egzempliorių reikšmes saugo suporintose lentelėse, kurių fragmentą galima pamatyti priede „“Storulio“ RDB fragmentas“ (žr. 10.4).

4.3.2.2. “Pilkasis kardinolas”, “Magnus Rex”, “Klaustuko galia” realizacijos

Priešingai nei „Mažylis“, „Storulis“ yra tikra OO metodologijos IS.

Projektavimo šablonai „Pilkasis kardinolas“ (žr. 3.2.3.2), „Magnus rex“ (žr. 3.2.3.3) realizuojami atitinkamomis klasėmis (priedas “Storulio” klausimynų IS metaklasės“, žr. 10.3).

Taikomas nesusietas objektų sekimas statinėje klasėje *MetaDataContainer*, kurioje C# kolekcijos talpina aukščiau minėtų klasių objektus, indeksuojamus pagal dažniausiai naudojamą ir atpažįstamą jų savybę:

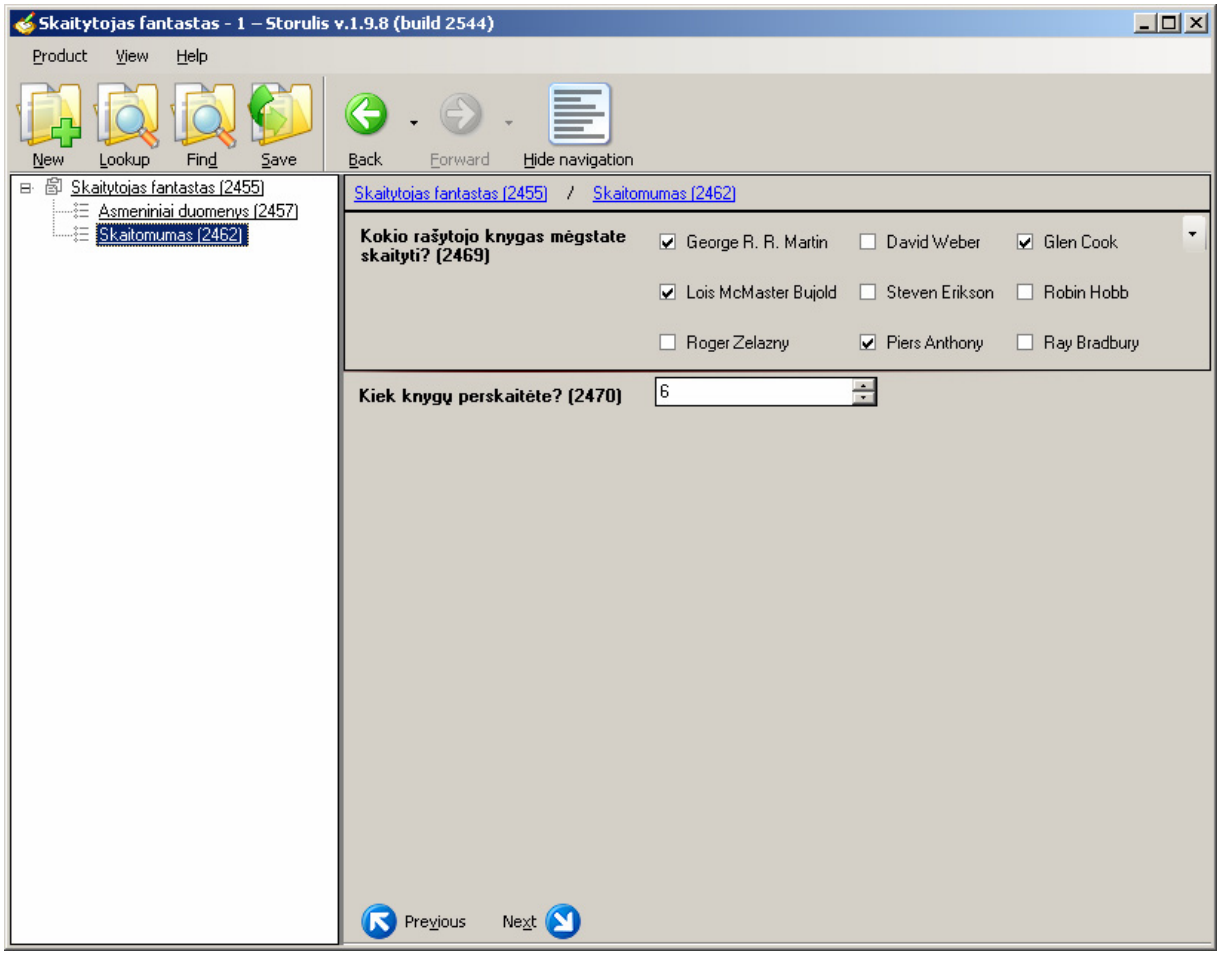
```
      [Serializable()]
          public class MetaDataContainer
          {
              public Dictionary<string, EntityDefinition> entities = new
Dictionary<string,EntityDefinition>();
              public      Dictionary<string,      EntityCategoryDefinition>
entityCategories = new Dictionary<string, EntityCategoryDefinition>();
              public Dictionary<Int32, PropertyDefinition> properties = new
Dictionary<int,PropertyDefinition>();
              public       Dictionary<Int32,       PropertyDomainDefinition>
propertyDomains = new Dictionary<int,PropertyDomainDefinition>();
              public Dictionary<Int32, PropertyDomainScalarTypeDefinition>
propertyScalarTypes                          =                          new
Dictionary<int,PropertyDomainScalarTypeDefinition>();
              public     Dictionary<string,     MappingTableAliasDefinition>
tableAliases = new Dictionary<string,MappingTableAliasDefinition>();
              public        Dictionary<Int32,        PropertyUnitDefinition>
propertyUnits = new Dictionary<int,PropertyUnitDefinition>();
              public   Dictionary<Int32,   PropertyLinkConditionDefinition>
linkConditions = new Dictionary<int,PropertyLinkConditionDefinition>();
              //Added to ease implementation of saving and cleaning
database up
              public List<PropertyLinkDefinition> propertyLinks = new
List<PropertyLinkDefinition>();
              public    Dictionary<Int32,    PropertyDomainValueDefinition>
propertyDomainValues = new Dictionary<int,PropertyDomainValueDefinition>();
              public Dictionary<Int32, PropertyDomainValueRangeDefinition>
propertyDomainRanges                         =                          new
Dictionary<int,PropertyDomainValueRangeDefinition>();
              public List<AssociatedEntityConfig> associatedEntities = new
List<AssociatedEntityConfig>();
              public Dictionary<string, EntityDefinition> tempEntities =
new Dictionary<string,EntityDefinition>();
```

4.3.2.3. “Projekcija” realizacija

Šis projektavimo šablonas visiškai realizuojamas „Storulyje“, suskaidant didelius klausimynus į kelis mažesnius ir apjungiant juos sudėtiniais domenais į vieną visumą. Priede „“Storulio“ RDB fragmentas“ (žr. 10.4) galima pamatyti mažą dalį visų į sistemą įvestų klausimynų egzempliorius saugančių lentelių *Microsoft SQL Server 2005 RDBVS*.

4.3.2.4. “Savivalda” realizacija

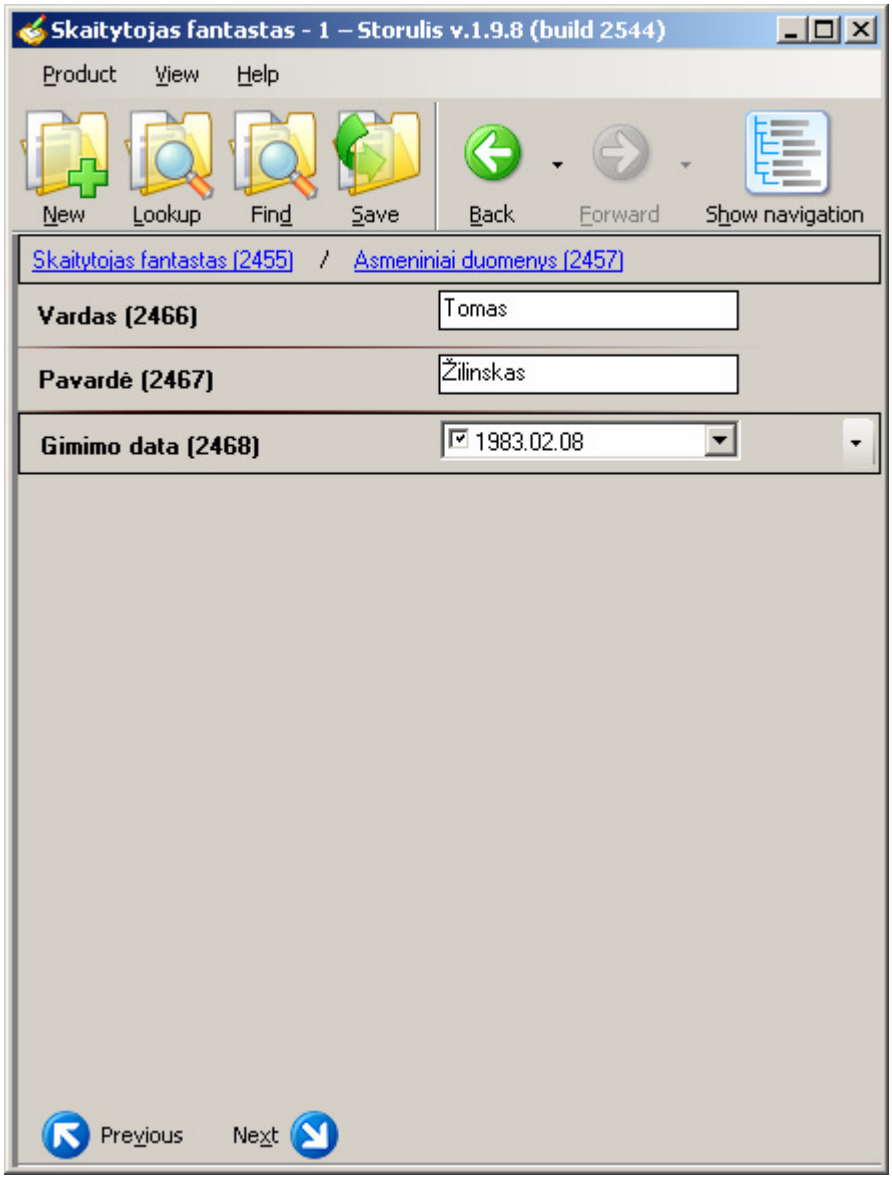
Pateikimas vartotojui gali būti dvejopas: pagal „Savivalda“ (žr. 3.2.3.8) (29 pav.):



29 pav. „Storulyje“ atidarytas klausimynas

4.3.2.5.“Žingsnis po žingsnio” realizacija

KOM gali būti dalinai pateikiamas pagal „Žingsnis po žingsnio“ (žr. 3.2.3.9) (30 pav.) vizualųjį modelį, vienu metu pateikiant keletą vieno hierarchinio lygio klausimų, bet nerodant „Savivalda“ būdingo medžio kairėje lango pusėje:



30 pav. „Storulyje“ minimaliai rodomas klausimynas

5. Sukurtos projektavimo šablonų kalbos kokybės tyrimas

Būtina atlikti darbo rezultato kokybės tyrimą pagal pasirinktus universalius (žr. 2.10) ir vietinius (žr. 2.11) skirtingų tipų kokybės kriterijus.

5.1. Tyrimas pagal universalius kriterijus

Universalūs kriterijai reikalauja parinkti kelias projektavimo šablonų kalbas – alternatyvas, - lyginamajai analizei. Šiuo atveju naudojamos dvi projektavimo šablonų kalbos, paimtos iš pasaulinio lygio specializuotos konferencijos *PLoP – Pattern Languages of Programming*, - dviejų leidimų. Alternatyvų charakteristikos ganėtinai nepanašios ir turėtų

sudaryti tinkamą kontekstą lyginimui su darbo rezultatu (8 lent.), kurioje pateikiami euristiniai autoriaus požiūriu paremti skaitiniai įvertčiai (skalėje 1-10, pilkumo atspalviais):

8 lentelė. Projektavimo šablonų kalbų palyginimas (universalūs kriterijai)

Kriterijus \ Kalba	***(Sukurta kalba)***	**[Esk99]**	**[Tow98]**
Projektavimo šablonų bei pačios kalbos vaizdumas	**6** ☒ psichologiniai šablonai sunkiai vizualizuojami	**8** ☑ Konceptai aiškinami *UML* struktūrinėmis bei sekų diagramomis; ☑ Sprendimai iliustruojami *Java* kodu	**1** ☒ nors tema – grafinė vartotojo sąsaja, visiskai nenaudojamos diagramos
Turinio inovatyvumas	**8** ☑ originali metaduomenimis valdomos klausimynų IS architektūra; ☑ klausimynų vizualizacijos pateikimas; ☑ klausimynų turinio psichologinis / socialinis pritaikymas; ☒ psichologinių dėsnių pagrindimas tyrimais paimtas iš literatūros	**10** ☑ aprašoma ypač lanksti komponentų architektūra; ☑ peržengiamos statinių interfeisų ribos; ☑ komponentai sąveikauja netiesiogiai, per magistralę	**1** ☒ klasikinių grafinės OS vartotojo sąsajos principų vaizdo perpiešimui perteikimas projektavimo šablonų kalbos forma
Kalbos struktūros sudėtingumas	**7** ☑ skirtingi šablonų tipai ir formatai; ☑ įvairūs ryšiai; ☑ agregacijos neprivalomumas; ☑ ribotas šablonų punktų kiekis	**2** ☒ primityvus šablonų siejimas ☒ išimtinis amorfinio šablonų formato *Portland* naudojimas	**4** ☒ veiksmažodiniai ryšiai tarp šablonų; ☑ išsamus šablonų formatas

Kaip matyti lentelėje, sukurta projektavimo šablonų kalba nedaug atsilieka nuo pasaulinio masto konferencijai pristatomų pagal du svarbiausius kriterijus (vienu atveju netgi neproporcingai lenkia) ir pirmauja pagal trečiąjį kriterijų. Pirmojo kriterijaus deficitas gali būti nesunkiai ištaisomas papildomomis pastangomis.

5.2. Tyrimas pagal vietinius kriterijus

Darbo rezultato būklė parinktų vietinių kriterijų aspektais yra:

- *Dalykinės srities (arba pasirinkto jos poaibio) realizavimo programinėje įrangoje ekspertinių žinių pilnumas* – **8:**
  - ☑ pateikiama klausimynų IS (meta)duomenų struktūrų aprašymai;
  - ☑ (meta)duomenų serializavimas reliacinėje DB;
  - ☑ duomenų struktūrų pateikimas vartotojui;
  - ☒ nesusisteminti algoritmai (kadangi jie labai įvairūs ir priklauso nuo konkretaus užsakovo įgeidžių);
  - ☒ nerealizuotos kai kurios retai naudojamos klausimynų savybės: procedūriniai ryšiai tarp klausimų / klausimų performulavimas (žr. 2.1.4).
- *Vertybinė koncentracija* – **10:**
  - ☑ projektavimo šablonų temos apsiriboja klausimynų IS konstravimu ir sukonstruotos universalios klausimynų IS metaduomenų pildymo optimizavimu.
- *Pagrįstumas* – **10:**
  - ☑ iš 10 programinių / grafinės vartotojo sąsajos šablonų bent viena pavyzdinė IS realizuoja 9 (daugelį – abi, b9damos labai nevienodos techninės architektūros prasme), o vienintelis likęs yra susietas gerinimo ryšiu, taigi, neesminis.
- *Pradinių reikalavimų optimalumas* – **10:**
  - ☑ Projektavimo šablonų sprendimuose manipuliuojama tik tipų, nurodytų kriterijaus apibrėžime, žiniomis.

Kaip matyti, pagal vietinius kriterijus darbo rezultatas – labai geras.

6. Išvados

1. Nustatyta, kad optimaliam naujo klausimynų projektuotojo darbui reikalingas ekspertines žinias optimaliausia pateikti projektavimo šablonų kalbos (metodikos) forma, naudojant *Coplien* ir *Portland* projektavimo šablonų formatus bei įvairius ryšius tarp jų.
2. Šiame darbe pateikiama klausimynų projektavimo šablonų kalba apima tiek IS, tiek ir jos turinio kūrimą, padeda pasirinkti laiko patikrintus sprendimus iš dvidešimt vieno projektavimo šablono trijose kategorijose.
3. Programinių / architektūrinių projektavimo šablonų, esančių kalboje, visuma įgalina klausimynų projektuotoją sukurti universalią, išplečiamą klausimynų IS, pasižyminčią savybėmis, būtinomis sudėtingiems klausimynams atvaizduoti.
4. Projektavimo šablonų kalboje pateikti vartotojo sąsajos projektavimo šablonai nusako optimalius žmogaus-mašinos sąveikos būdus klausimynų kontekste.
5. Įtraukti psichologiniai projektavimo šablonai leidžia pagerinti klausimynų sandarą ir tiksliau suformuluoti pačius klausimus, taip sumažinant neatitikimus tarp apklausiamojo tikrosios nuomonės, ir to, kas užfiksuojama.
6. Pasiūlyta projektavimo šablonų kalba leidžia naujam klausimynų projektuotojui mokytis palaipsniui – skaitant visų trijų tipų projektavimo šablonus ir einant įvairiais ryšiais tarp jų iki reikiamo detalumo lygio. Taip sumažinamas pradinis žinių slenkstis, ir naujas darbuotojas gali anksčiau prisidėti prie sistemų plėtojimo.
7. Dvi komercinės klausimynų IS jau yra sukurtos pagal aprašytos klausimynų projektavimo šablonų kalbos, pritaikytos praktikoje (žr. įdiegimo aktą) ekspertines žinias. Tai įrodo, kad reikalingų ekspertinių žinių pateikimas projektavimo šablonų kalbos (metodikos) pavidalu pasiteisino: klausimynų projektuotojai lengvai atsirenka konkrečiai problemai tinkančius šablonus ir noriai naudoja šablonų pavadinimus vidinėje komunikacijoje.

7. Tolesnės tyrimų kryptys

- Klausimynų projektavimo šablonų rinkinį išplėsti naujais projektavimo šablonais. Tai turėtų būti lengva dėl diskrečios, modulinės pasiūlytos projektavimo šablonų kalbos konstrukcijos.
- Standartizuoti ir pateikti projektavimo šablonų forma algoritminę klausimynų IS dalį (tradiciškai labai priklausomą nuo kliento norų).
- Prijungti projektavimo šablonus, aprašančius metaduomenų saugojimą (KDB, žr. priedą „“Mažylio“ klausimynų IS KDB schema“, 10.2).
- Klausimynų projektavimo šablonų kalbą pateikti šios srities specialistų peržiūrai (angl. *peer review*).
- Pasiūlytą klausimynų projektavimo metodiką išbandyti su kitokios sudėties ir didesnėmis komandomis.
- Pakankamai išvysčius klausimynų projektavimo šablonų kalbą, oficialiai ją paskelbti *PLoP* (*Pattern Languages of Programming*) konferencijoje ir / ar globaliuose internetiniuose projektavimo šablonų kalbų saugyklose (angl. *repositories of patterns* / *pattern languages, pvz.,* [PLT06]).

8. Terminų ir santrumpų žodynas

- **IS** (**I**nformacinė **S**istema; angl. ***I****nformation* ***S****ystem*) - žmogaus-mašinos sistema, kurią sudaro žmonės, techninės ir programinės priemonės, darbo instrukcijos;
- **XML** („*e****X****tended* ***M****arkup* ***L****anguage*“, „Išplėsta žymėjimo kalba“) – supaprastintas universalios dokumentų kūrimo kalbos *SGML* variantas, įgalinantis aprašyti įvairių pavidalų duomenis struktūrizuota forma; įgalina specializuotų pokalbių kūrimą;
- **DOM** („***D****ocument* ***O****bject* ***M****odel*“, „Dokumento objektų modelis“) – struktūrizuotų dokumentų pateikimo objektiškai orientuotu modeliu būdas. Dažniausiai dokumento struktūra pateikiama kaip aibė jos elementus atitinkančių objektų, sujungtų į medžio struktūrą.
- **OS** („***O****perating* ***S****ystem*“, „Operacinė sistema“) – programinės įrangos kompleksas, kurio kontekste veikia vartotojo programos, besinaudojančios *OS* teikiamomis paslaugomis;

- **HTTP** („***H**yper**T**ext **T**ransfer **P**rotocol*“, „Hiperteksto persiuntimo protokolas“) – pagrindinis žiniatinklio funkcionalumą užtikrinantis protokolas, naudojamas žiniatinklio duomenų persiuntimui;
- **URL** („***U**niform **R**esource **L**ocator***“**, „Vieningas resursų identifikatorius“) **–** standartinis įvairaus tipo informacinių resursų internete adreso formatas;
- **OOP** („***O**bject – **O**riented **P**rogramming*“, „Objektiškai orientuotas programavimas“) – progresyvus požiūris į projektavimą ir programavimą, grupuojant duomenis ir veiksmus pagal jų loginį bendrumą;
- **REGEXP** („***Reg**ular **Exp**ression*“, „Reguliari išraiška“) – simbolių eilutė, kuri, pagal tam tikrą sintaksę, aprašo arba atitinka tam tikrą simbolių eilučių aibę. Plačiai naudojama teksto paieškoje, keitime ir filtravime;
- **Applet** („įskiepis“) – nesavarankiška programa, dažniausiai veikianti naršyklės ar joje įdiegto konteinerio kontekste;
- **Javascript** – „Netscape“ sukurta interpretuojama objektiškai orientuota kalba, dažniausiai veikianti naršyklėse. Standartizuota *ECMA-262* standartu;
- **Predikatas** – funkcija ar operacija, grąžinanti Būlio logikos reikšmę („taip“ arba „ne“);
- **„Polish notation“** („Lenkiškoji notacija“) – būdas vienareikšmiam sudėtingų loginių sąlygų užrašymui, nenaudojant skliaustų (operacija iškeliama prieš operandus);
- **AJAX** („***A**synchronous **J**avascript **A**nd **X**ML*“, „Asinchroninis Javascript ir XML“) – terminas, apibūdinantis žiniatinklio programavimo stilių, kuriame naudojama *HTTP* ryšys, *DOM*, *Javascript* (kaip kliento pusės kalba), *XML* (duomenų perdavimo formatui), *XMLHttpRequest* objektas (ryšiui su serveriu). *AJAX* įgalina sukurti greitaveikes žiniatinklio programas, sąsaja primenančias įprastines;
- **.NET** – firmos Microsoft globali platforma, skirta įvairių tipų taikomųjų programų kūrimui ir vykdymui. Veikimo principu primena Sun Java, bet išsiskiria dideliu palaikomų programavimo kalbų kiekiu (oficialiai C#, C++, Visual Basic, JScript, bet pritaikyta ir kitos, pvz., Borland Delphi, Fortran). Vykdymui naudoja tarpinį kodą, bet prieš vykdymą sukompiliuoja į atitinkamo procesoriaus architektūrą.

9. Literatūra

- [Gam+95] Design Patterns: Elements of Reusable Object-Oriented Software/ Gamma E., Helm R., Johnson R., Vlissides J. - Boston: Addison-Wesley, 1995. - 395 p.
- [Shal+04] Shalloway A., Trott J. R. Design Patterns Explained: A New Perspective on Object-Oriented Design, Second Edition. - Boston: Addison-Wesley, 2004. - 480 p.
- [Bus+96] Pattern-Oriented Software Architecture: A System of Patterns/ Buschmann F., Meunier R., Rohnert H. ir kt. - New York: John Wiley & Sons, 1996. - 476 p.
- [Cop98] Coplien J. O. Software Design Patterns: Common Questions and Answers// The Patterns Handbook: Techniques, Strategies, and Applications / ats. red. L. Rising - New York: Cambridge University Press, 1998. - p. 311-320.
- [Lau98] Lauder A., Kent S. Precise Visual Specification of Design Patterns// ECOOP'98 -- Object-Oriented Programming: Lecture Notes in Computer Science / ats. red. E. Jul. – Springer-Verlag, 1998, T.1445. - p. 114-134.
- [Ale+79] Alexander C., Ishikawa S., Silverstein M. The Timeless Way of Building. - New York: Oxford University Press, 1979. - 524 p.
- [Ale+77] A Pattern Language/ Alexander C., Ishikawa S., Silverstein M. ir kt. – New York: Oxford University Press, 1977. – 1216 p.
- [PLT06] Pattern Language Titles [interaktyvus] [žiūrėta 2006-09-20]. Prieiga per internetą: < http://c2.com/ppr/titles.html >;
- [Ada+95] Fault-Tolerant Telecommunication System Patterns/ Adams M., Coplien J. O., Gamoke R.// Pattern Languages of Program Design: Proceedings of the Second International Conference on Pattern Languages of Programming / ats. red. Coplien J. O., Kerth N., Vlissidis J. – Monticello, Illinois: Addison-Wesley, 1995. – 482 p.
- [Tow98] Display Maintenance/ Towell, D.// Pattern Languages of Program Design: Proceedings of the Fourth International Conference on pattern Languages of Programming / ats. red. Harrison N., Foote B., Rohnert H. - Monticello, Illinois: Addison-Wesley, 1999. – 513 p.
- [Esk99] Component Interaction Patterns/ Eskelin P.// Pattern Languages of Program Design: Proceedings of the Fifth International Conference on Pattern Languages of Programming / ats. red. Manolescu D., Voelter M., Noble J. – Monticello, Illinois: Addison-Wesley, 2001. – 429 p.
- [Bra+04] Bradburn N., Sudman S., Wansink B. Asking Questions: The Definitive Guide to Questionnaire Design for Market Research, Political Polls, and Social and

Health Questionnaires, Revised Edition. - New York: John Wiley & Sons, 2004. - 425 p.

- [Cas+01] Case S. M., Swanson D. B. Constructing Written Test Questions for the Basic and Clinical Sciences. - New York: National Board of Medical Examiners, 2001. - 131 p.
- [Bie+03] Biemer P. P., Lyberg L. E. Introduction to Survey Quality. - New York: John Wiley & Sons, 2003. - 397 p.
- [Tou06] Tourangeau R. The Importance of Nonresponse for Survey Design // Joint Statistical Meetings 2006. - Seattle, 2006, p. 248-261.
- [Ker+06] Kerwin J., Wang A., Campbell S., Shipp S. A Comparison of Strategies for Reducing Item Nonresponse in a Web Survey // Joint Statistical Meetings 2006. - Seattle, 2006, p. 308-317.
- [Tho+06] Thomas R. K., Greenfield S., Bremer J. Response Order Effects in International Online Surveys // General Online Research 2006: The 8th international conference and congress fair. – Ravensberger Spinnerei, Bielefeld, 2006, p. 86-94.
- [Ver+06] Verheyen C., Schuebel C., Moser K. The impact of persuasion strategies on the response rate in online surveys: Incentives, foot-in-the-door-technique or when 'even a penny will help' // General Online Research 2006: The 8th international conference and congress fair. – Ravensberger Spinnerei, Bielefeld, 2006, p. 112-117.
- [Gan06] Ganassali S. Web surveys questionnaire design and quality of responses // SMABS-EAM Conference. – Budapest, 2006, p. 187-198.
- [Gal+06] Galesic M., Bosnjak M. Response Biases in Online Surveys // General Online Research 2006: The 8th international conference and congress fair. – Ravensberger Spinnerei, Bielefeld, 2006, p. 167-181.
- [Žil06] Žilinskas, T. AJAX technikos panaudojimo žiniatinklio informacinėse sistemose galimybių analizė // Informacinės technologijos 2006: 11-iosios tarpuniversitetinės doktorantų ir magistrantų konferencijos pranešimų medžiaga, II dalis [Kaunas, 2006 m. balandžio 28 d.]. - Vilnius, 2006, p. 64-69.

10. Priedai

10.1. Straipsnis

AJAX TECHNIKOS PANAUDOJIMO ŽINIATINKLIO INFORMACINĖSE SISTEMOSE GALIMYBIŲ ANALIZĖ

Tomas Žilinskas

Kauno technologijos universitetas, Informatikos fakultetas

Klasikinės žiniatinklio informacinės sistemos yra statiškos, menkai struktūrizuotos ir sąlyginai nepatogios naudoti.

Straipsnyje analizuojama *AJAX* technikos įtaka, galimybės ir potencialūs panaudojimo būdai kuriant šiuolaikines dinamiškas žiniatinklio informacines sistemas. Aprašyti patirtimi pagrįsti rekomenduotini panaudojimo scenarijai.

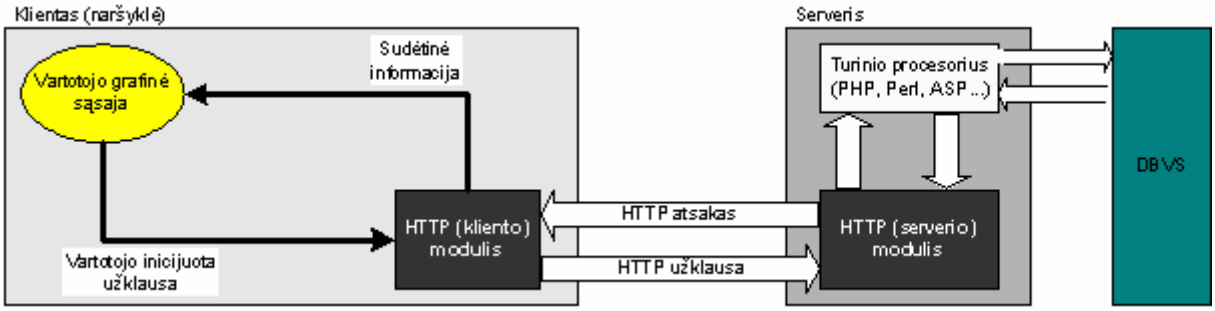
1. Įvadas

Žiniatinkliu pagrįstų informacinių sistemų bazinė architektūra ir veikimas iš vartotojo pusės faktiškai nesikeičia nuo 1990 metais išrastos žiniatinklio naršyklės pristatymo. Serverio pusės technologijos nuolat tobulėjo, tačiau prezentacijos lygmenį buvo apėmusi inovacinė stagnacija.

Pagaliau, pirmą kartą nuo naršyklės išradimo, atsirado ir plinta universali technika *AJAX*, siūlanti galimybę radikaliai pakeisti pasenusią žiniatinklio informacinių sistemų vartotojo sąsajos architektūrą ir prieinamumą vartotojui. Šiame darbe nagrinėjamos *AJAX* perspektyvos, santykis su egzistuojančiomis konvencijomis bei praktikoje atrasti optimalūs panaudojimo scenarijai.

2. Problema

Klasikinė žiniatinklio informacinės sistemos architektūra yra pagrįsta „kliento - serverio“ paradigma (1 pav.):



1 pav. Klasikinė naršyklės – serverio architektūra

Pagrindiniai šios architektūros trūkumai:

- Iš vartotojo pusės: *HTTP* užklausos, vykdomos išimtinai vartotojo inicijacija (egzistuoja minimalių išimčių) savo prigimtimi yra **sinchroninės**, todėl, kol užklausa negrąžino atsako (naujų puslapio duomenų), vartotojas negali atlikti jokių veiksmų. Atsižvelgiant į nepastovią interneto ryšio kokybę, *HTTP* užklausa ir didelės apimties atsakas gali ilgai užtrukti, taip vartotojui sukeliant nepatogumų ir diskomfortą.
- Efektyvumo: labai dažnai pasitaiko atvejų, kai *HTTP* užklausa sąlygoja atsiuntimą naujos didelės apimties puslapio versijos, kuri labai nedaug skiriasi nuo buvusios naršyklės atmintyje prieš užklausos nusiuntimą. Elementarus pavyzdys: pakeitus vieną sudėtingos ekrano formos lauko reikšmę ir nusiuntus visą formą patikrinimui, serverio atsake paprastai bus ta pati forma – kai užtektų patvirtinimo: „pavyko / klaida“.
- Architektūrinis: iš principo *HTML* formatas **apjungia** Prezentaciją bei Duomenis (pagal *MVC* paradigmą). Dažniausiai bet kokie duomenys yra talpinami kartu su informacija, „kur“ ir „kaip“ jie turėtų būti atvaizduoti naršyklės lange („kur“ – *(X)HTML* žymės, „kaip“ – *CSS* formatavimo komandos). Pasikartojantiems vieno tipo bei semantikos duomenims pozicinė / formatavimo informacija dažnai dubliuojama. Prezentacijos bei Duomenų apjungimas padidina painiavą, mažina struktūrizavimo lygį, bereikalingai didina duomenų srautus, šitaip prisidėdamas ir prie efektyvumo problemos.

Daug kartų mėginta apeiti minėtus apribojimus, tačiau visais atvejais pagrindinis bruožas buvo papildoma į naršyklę integruota programinė įranga. Įvairių rūšių įskiepiai (*„applets“*) suteikdavo naršyklei naujų savybių, pakeisdavo informacijos srauto eigą, tačiau nė vienas netapo visuotinai pripažintu sprendimu 1 pav. pavaizduotos architektūros trūkumams.

Žymiausi tobulinimo bandymai:

- *Sun Java*: įskiepiai, galintys naudoti daugelį *Java* kalbos savybių, tačiau apriboti saugumo nustatymų. Versijų nesuderinamumas gali sukelti įvairiausių problemų.

- *Macromedia Flash*: įskiepiai, pasižymintys įspūdinga grafine posisteme, tačiau neypatingu duomenų valdymo palaikymu *ActionScript* kalba. Pasitaiko versijų konfliktų.

Dėl vartotojų inercijos, nenoro įdiegti bet kokią papildomą PĮ, versijų nesuderinamumo tarp konkrečių įskiepių ir palaikomosios PĮ, bei standartų nebuvimo (visi įskiepiai – atskirų firmų gaminiai), nei vienas sprendimas nepasiekė absoliučios (ar bent dominuojančios) skverbties vartotojų srityje, tuo labiau – IS kūrėjų.

Kai kurie ne įskiepiais pagrįsti sprendimai (*IFRAME* foninis krovimas, *ECMAScript* kodo dinaminis generavimas) buvo gana geri, tačiau palaikomi tik vienoje ar keliose naršyklėse, be to – „*hack*“ lygio (t.y. efektas pasiekiamas priemonėmis, skirtoms kitiems tikslams).

3. *AJAX* esmė

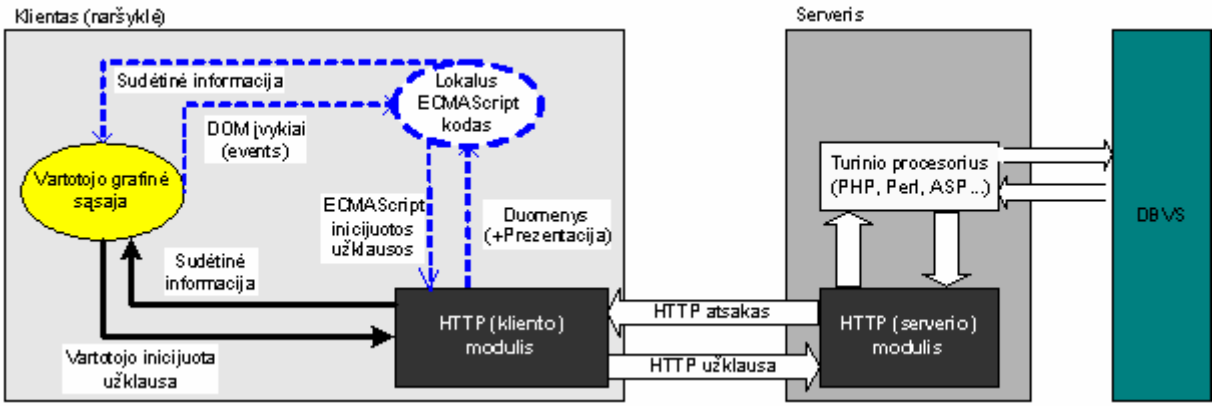
AJAX - pažangus *JavaScript* su *XML* (*„Asynchronous JavaScript And XML“*, arba *„Advanced JavaScript And XML“*), – tai technika (išpopuliarėjusi 2005 metais [7]), kuri, su tikslu pakeisti klasikinę žiniatinklio sistemų architektūrą, numato sinchroninį naudojimą šių technologijų:

- *ECMAScript* standarto programavimo kalba, veikianti naršyklėje, plačiau žinoma kaip *JScript* (*Microsoft* PĮ) arba *JavaScript* (visi kiti gamintojai);
- Naršyklės puslapio manipuliavimo *DOM API* (plačiau žinomas kaip *DHTML* dalis);
- Foninio interneto ryšio naršyklėje programinis komponentas *XMLHTTPRequest* (dar vadinamas *XHR*). Jis įgalina programiškai inicijuoti *HTTP* užklausas iš *ECMAScript* kodo;
- Metakalba *XML* – naudojama *HTTP* užklausoms/atsakams bei puslapio turiniui (*XHTML* – *XML* pokalbė);

Suderintai naudojant minėtas technologijas (kurios, imant pavieniui, jau seniai taikomos IS kūrime, tik nevienodais lygiais), įmanoma pasiekti kokybiškai tobulesnę žiniatinklio informacinių sistemų architektūrą.

Klasikinė sudėtinga žiniatinklio IS yra realizuojama plono kliento (*„thin client“*) architektūra, kuri yra globalios „klientas-serveris“ architektūros porūšis. Tokiu atveju didžioji skaičiavimų dalis atliekama serveryje, kas turi privalumų ir trūkumų.

AJAX suteikia galimybę architektūrą pakeisti į storo kliento (*„thick client“*, dar žinomas kaip „sudėtinga žiniatinklio aplikacija“ – *RIA*, *„Rich Internet Application“*) porūšį, kuriame didžioji skaičiavimų dalis atliekama naršyklėje, *ECMAScript* kalbos kode (2 pav.; brūkšninėmis linijomis pavaizduoti *AJAX* įvesti pokyčiai):



2 pav. Naršyklės-serverio architektūra su XHR išplėtimu

AJAX pagrindu jau veikia nemažai įtakingų žiniatinklio svetainių (*AJAX* pionierės *Google* paieška (*„Suggestions“*), paštas (*„GMail‘*), *Amazon.com A9* paieška, *Netflix*), tačiau firmų IS srityje pritaikymas dar nežymus.

Nors *XHR* komponentas – esminė *AJAX* dalis, - nėra standartizuotas, visos naršyklės jį realizuoja beveik vienodai – tiek interfeiso, tiek veikimo prasmėmis. *W3C* organizacija rengia *„DOM Level 3 – Load And Save Specification“* pasiūlymą, kuris oficialiai nustatys *XHR* standartą.

4. Architektūros pokyčiai

Nauja sąranga išlaiko visus plono kliento architektūros privalumus ir neturi jos trūkumų (1 lent.):

1 lentelė. Skirtingo storio klientų palyginimas

Plono kliento charakteristikos	**Pokyčiai, įvedus storą klientą su *AJAX***
• **Privalumai**	
Kodo pokyčiai iškart propaguojami klientams	Nežymūs - kadangi verslo logikos didžioji dalis patalpinama *ECMAScript* kode, kuris kiekvieną kartą atsiunčiamas iš serverio (žr. 4.1 punktą)
Greitai pasiekiami duomenys	Nežymūs - struktūrizuoti duomenys gali būti (foniniu režimu) atsiunčiami į naršyklę ir saugomi *ECMAScript* atmintyje viso darbo su IS metu (žr. 4.6 punktą)
Centralizuotas valdymas	Nežymūs - naršyklės *ECMAScript* kodas gali bet kada programiškai užmegzti ryšį su serveriu ir gauti

Plono kliento charakteristikos	Pokyčiai, įvedus storą klientą su *AJAX*
	būsenos / elgesio nuorodas – neįtakodamas vartotojo sąsajos
Nereikia jokios papildomos PĮ	Nežymūs – *AJAX* komponentus paremia didžioji dalis šiuolaikinių naršyklių, taigi, *AJAX* veikia nulinių reikalavimų (*„zero footprint“*) principu
• **Trūkumai**	
Serverių aparatūra turi būti labai galinga, brangi	Žymūs – skaičiavimai atliekami klientų kompiuteriuose, todėl serveriai gali būti paprastesni ir pigesni
Dideli pasikartojančių duomenų srautai tarp kliento ir serverio – lėtas puslapių atvaizdavimas	Žymūs – siunčiami tik tie duomenys, kurie reikalingi
Beveik kiekvienas vartotojo sąsajos pokytis reikalauja pilno atnaujinimo (*„round-trip“*) iš serverio	Žymūs – *DOM* manupuliacijos iš *ECMAScript* kodo įgalina keisti sąsajos elementus be serverio įsikišimo

4.1 Skaičiavimų krūvio perkėlimas į klientą (ECMAScript)

Istoriškai *ECMAScript* kalba buvo laikoma nepatikima, siauro pritaikymo (vykdoma praktiškai vien tik naršyklių kontekste), todėl - neabejotinai netinkama bent kiek rimtesniems skaičiavimams, išskyrus (galbūt) grafinių efektų generavimą ar formų validavimą. Pagrindiniai trūkumai ir jų sprendimai pateikiami 2 lentelėje:

2 lentelė. ECMAScript trūkumai ir jų sprendimai

Trūkumas IS kūrimo kontekste	Šiuolaikinis sprendimas
Standartizavimo trūkumas (skirtingos naršyklės realizuoja nevienodą sintaksę ir veikimą)	Laikantis *ECMAScript* standarto, garantuojamas veikimas visose šiuolaikinėse naršyklėse [9]
Trūksta OO galimybių	*ECMAScript* yra objektinė iš prigimties; visos jos esybės, įskaitant funkcijas, yra objektai. Egzistuoja

Trūkumas IS kūrimo kontekste	Šiuolaikinis sprendimas
	savotiškos prototipinės klasės. Paveldimumas, enkapsuliacija ir polimorfizmas gali būti nesunkiai realizuoti programiškai
Labai lėtas vykdymas (interpretavimas)	Egzistuoja daug greičio optimizavimo būdų (žr. 4.3 punktą)

Praktika rodo, jog sudėtingos žiniatinklio IS verslo logiką galima perkelti į kliento pusę, kur ji bus vykdoma išimtinai *ECMAScript* kontekste. Savaime suprantama, dėl interpretacinės prigimties *ECMAScript* negali pasiekti klasikinių programų greičio.

Perkėlus verslo logiką į klientą, serverį galima traktuoti kaip duomenų šaltinį, ir atitinkamai optimizuoti: organizuoti dažnai reikalingų duomenų buferius (*„cache“*) su pertekliškumu. Labai tampriai su DBVS susijusias paprogrames (pvz., *CRUD* operacijoms vykdyti) patartina palikti serverio kode.

4.2 Duomenų perdavimas

Beveik visose šiuolaikinėse naršyklėse (*IE 5+, Mozilla 1+, Opera 8+*) esantis *XHR* komponentas leidžia foniniu režimu (netrikdant vartotojo ir nestabdant jo veiksmų) siųsti duomenis norimu formatu. Atsiųsti duomenys gali būti apdorojami / konvertuojami *ECMAScript* kalba bei patalpinami puslapiui skirtoje *ECMAScript* atmintyje, kur gali būti laikomi kiek norima ilgai, šitaip išvengiant pakartotinio jų siuntimo kiekvienos užklausos į serverį metu.

Praktikoje patartina vartotojui suteikti galimybę atsisiųsti visus statinius duomenis vos tik pradėjus darbą su IS, tačiau tai gali užtrukti, todėl vartotojui turėtų būti suteikiama galimybė siuntimo procesą perkelti į foną ir leisti atlikti bent įžanginius veiksmus su IS grafine sąsaja.

Foninis duomenų siuntimo procesas beveik neapkrauna naršyklės, tačiau duomenis atsiuntus ir juos apdorojant prieš patalpinimą į *ECMAScript* atmintį, skaičiavimai sulėtina / sustingdo vartotojo veiksmus sąsajoje, todėl reikia stengtis tuos veiksmus padaryti kiek įmanoma paprastus (pvz. duomenis siųsti formatu, kuo labiau panašiu į saugojimo formatą).

4.3 *ECMAScript* kodo optimizavimas

ECMAScript kalba yra interpretuojama, ne kompiliuojama, todėl kritiškai svarbu kodą optimizuoti, kad būtų pasiektas vartotojui priimtinas IS veikimo greitis. Pagrindinės strategijos [11]:

- Pakartotinis skaičiavimo rezultatų panaudojimas, saugant juos *ECMAScript* lokalioje atmintyje;
- Daugybinių nuorodų (obj1.obj2[,narys‘].savybe1.savybe2) vengimas;

- Išraiškų keitimas: „if“ sąlygų sutvarkymas pagal tikėtinumą, ciklų gerinimas (*„unwinding“*);
- Ribotas OO – įvedamas abstraktumas reikalauja skaičiavimo laiko sąnaudų;

4.4 *ECMAScript - DOM* optimizavimas

Pagrindinis veiksnys, mažinantis skaičiavimo greitį naršyklėje – interakcija su puslapio *DOM*. Dažniausiai algoritmų rezultatas turi būti vizualus, o IS duomenų apdorojimo atveju – keisti didelę puslapio turinio dalį. Programinė sąsaja, per kurią galima keisti puslapio sudėtį, yra ganėtinai išsami, padaro prieinamą detalią informaciją apie puslapio elementus, tačiau dėl to komandos vykdomos ypač lėtai (lyginant su įprastiniu *ECMAScript* kodu). Todėl auksinė taisyklė šiuo atveju – *ECMAScript* kode naudoti kiek įmanoma mažiau *DOM* komandų, o ypač - jautriose vietose (*„bottlenecks“*).

Ganėtinai plačiai paplitusi (kiek taip apskritai galima teigti naujai technologijai) praktika saugoti būsenos informaciją puslapyje esančių *XHTML* elementų nestandartiniuose (*W3C* prasme) atributuose yra absoliučiai žalinga. Tokia informacija dažnai būna reikalinga daugelyje net ir pačių elementariausių verslo logikos algoritmų, ir jos pakartotinis gavimas per lėtas *DOM* komandas sukelia pakankamai didelį greičio praradimą, kad algoritmų vykdymas naršyklėje taptų nepraktiškas.

Pasiteisinęs sprendimas – puslapio elementams suteikti kompozicinius ID (per *XHTML* atributą „id“), o duomenis apie elementus saugoti *ECMAScript* atminties globaliuose kintamuosiuose – dažniausiai masyvuose, nurodant susieto *XHTML* elemento ID. Vykdant algoritmus *DOM* komandos bus naudojamos tik tuomet, kai keisis *XHTML* elemente matoma informacija.

Kraštutiniu atveju neužtenka saugoti nuorodas į *XHTML* elementą, nes norint jį keisti, reikia *DOM* komanda *getElementByID* jį surasti, kas nebūtinai atitinka O(n) sudėtingumą. Siekiant maksimizuoti greitį, patartina *XHTML* elemento sukūrimo metu *ECMAScript* kintamajame išsaugoti ne tik ID, bet ir nuorodą į elemento objektą, gaunamą aukščiau minėta *DOM* komanda.

4.5 Saugumas

Kritikai pastebi, jog *ECMAScript* kodas, naršyklėje viso labo interpretuojamas, yra prieinamas kiekvienam bent kiek išmanančiam tinklapio vartotojui. Tačiau egzistuoja pagalbinės programos, padarančios kodą beveik neįskaitomu, ar net jį suspaudžiančios (naudinga tiek saugumo, tiek greičio prasmėmis) [10]. Derėtų priminti, jog ir klasikinių programų kodą galima ganėtinai nesunkiai atkurti iš dvejetainių failų. Taigi, *ECMAScript* naudojimas verslo logikai nesukelia didesnės rizikos, nei alternatyvos.

Norint išvengti vartotojo prisijungimo perėmimo (*„hijacking“* / *„spoofing“*), patartina būseną saugoti sesijoje, naudojantis sausainiukais (*„cookies“*) – šios *PHP* ir analogiškų turinio apdorojimo

kalbų savybės galioja ir foniniams *XHR* komponentu užmegztiems ryšio kanalams. Iš to išplaukia, kad vartotojo autentikacija ir, ypač, autorizacija turi būti realizuotos serveryje, kuris kontroliuos firmos duomenų prieinamumą pagal vartotojo teisių lygį. Potencialus įsilaužėlis, net ir perpratęs *ECMAScript* kode esančios verslo logikos veikimą ir jį modifikavęs, negalės gauti iš serverio duomenų, neturėdamas prisijungimo prie IS.

4.6 Duomenų kanalai

Duomenys, perduodami *XHR* inicijuotu *HTTP* kanalu, gali atitikti keletą formatų:

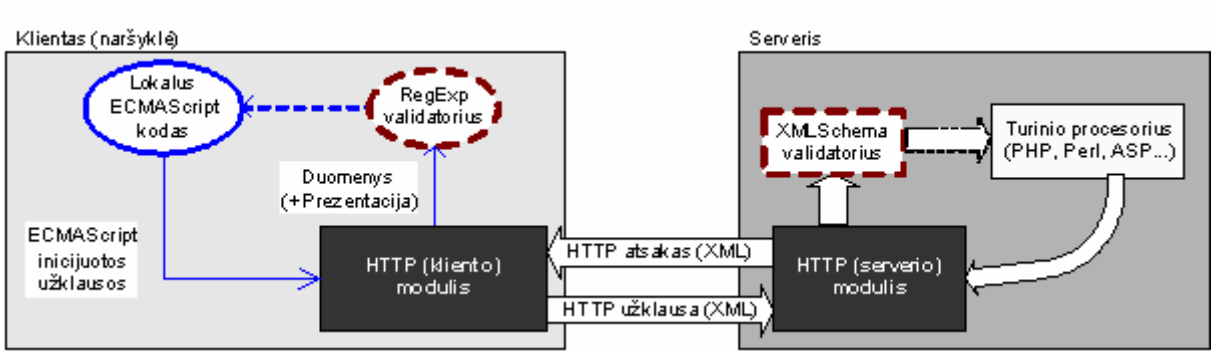
- *XML*: universaliausias formatas, leidžiantis naudoti norimą pokalbę. Įgalina *XMLSchema* tikrinimus (žr. žemiau), be to, *ECMAScript* gali *XML* duomenis apdoroti standartinėmis *DOM* funkcijomis;
- *JSON*: įgalina tekstu perduoti serializuotus *ECMAScript* duomenis. Serializavimo/deserializavimo bibliotekos yra sukurtos daugeliui populiarių programavimo kalbų, įskaitant *PHP*;
- *Unikalus* tekstu pagrįstas formatas: leidžia duomenis užkoduoti norimu būdu.

Nors tekstinis formatas yra lanksčiausias, *JSON* – užima nedaug vietos, tačiau *XML* suteikia daugiausia galimybių:

- Kreiptis į žiniatinklio paslaugas (*„web services“*) per *SOAP* užklausas, kurios parašytos *XML* pokalbe;
- Supaprastintai kreiptis į specializuotas žiniatinklio paslaugas *REST* formatu, kuris yra *XML* pokalbė;
- Automatiškai tikrinti į serverį atėjusių *XML* užklausų struktūros teisingumą naudojantis *XMLSchema* validatoriumi – šitaip pagerinamas serverio kodo stabilumas (blogai suformuotos užklausos atmetamos) bei saugumas (piktybinės specialiai suformuotos užklausos atmetamos, kaip neatitinkančios *XML* pokalbės formato);
- Perduoti *XHTML* dokumentą, kurį naršyklėje galima modifikuoti *ECMAScript DOM* komandomis arba iš karto įterpti į norimą IS puslapio vietą – taip paprastai suteikiant sistemai TVS galimybių (galima iš serverio atsiųsti formos šabloną arba iškart suformatuotus duomenis).

Naršyklėje iš *XHR* kanalo ateinantys struktūrizuoti duomenys (įskaitant *XML*) gali būti automatizuotai tikrinami *RegExp* priemonėmis pagal norimo sudėtingumo taisykles (t.y. kliente *RegExp* gali atlikti tą patį, kaip ir *XMLSchema* - serveryje).

Įdiegus tikrinimą kliente ir serveryje, galima pasiekti, jog ryšio linijos arba vienos pusės programinės klaidos neįtakos kitos pusės veikimo, šitaip efektyviai užtikrinant duomenų saugą bei saugumą nuo piktybiškai klaidingų užklausų (3 pav.; brūkšninėmis linijomis pavaizduoti pokyčiai).



3 pav. XML kanalas su dvipusiu integralumo tikrinimu (fragmentas)

Dėl asinchroninės *XHR* kanalo prigimties reikia papildomų priemonių įgyvendinant grįžtamąjį ryšį (*„callback“*): *ECMAScript* funkcijos(-ų) iškvietimą, kai *XHR* kanalas gauna duomenis iš serverio. Gaunamų duomenų formate (patogiausia - *XML*) reikia numatyti vietą grįžtamosios funkcijos(-ų) pavadinimui(-ams) ir parametrams. Patį funkcijos(-ų) atsiradimą gaunamuose duomenyse galima realizuoti dvejopai:

- Užklausos duomenų formate (patogiausia - *XML*) numatyti vietą funkcijos(-ų) pavadinimui(-ams) ir parametrams; visa tai bus automatiškai nukopijuojama į atsaką. Šis būdas patogus, kai tos pačios užklausos rezultatus kartais apdoroja skirtingos funkcijos;
- Serverio paprogramė, apdorojanti konkrečią užklausą, pati prideda grįžtamosios funkcijos vardą ir parametrus. Šis būdas patogus statinių užklausų atvejais, be to, sumažina užklausos dydį.

5. Vartotojo požiūrio pokyčiai

AJAX pritaikymas IS vartotoją gali gluminti [5]: per eilę metų įprantama, jog, kažką atlikus žiniatinklio IS grafinėje sąsajoje, visas puslapis lėtai persikrauna. *AJAX* naudojančios žiniatinklio IS, vartotojo sąsajos požiūriu besielgiančios kaip paprastos *Windows* / *Unix* programos, turėtų kompensuoti vartotojų inerciją [6], vizualiais signalais (subtiliai pasikeitęs fonas, laikinai invertuota teksto spalva) nurodydamos pokyčių vietą.

Globaliai *AJAX*, įgalinanti žiniatinklio IS supanašėjimą su tradicinėmis, yra naujos koncepcinės bangos – *Web 2.0*, kurioje akcentuojama idėja „žiniatinklis – tai platforma“ – dalis.

6. Išvados

- *AJAX* technikos taikymas įgalina sukurti kokybiškai naujas, dinamiškas žiniatinklio informacines sistemas, sumažinant lėšas serverių fermoms ir padidinant interaktyvumą;

- Kad *AJAX* pagrindu sukurta IS prilygtų įprastinėms savo funkcionalumu ir greičiu, būtina laikytis specifinių projektavimo ir programavimo praktikų;
- AJAX palaikymui naršyklėje nereikia jokios papildomos programinės įrangos.

Literatūra

[1] Aslesson R., Schutta N. T. Foundations of Ajax. New York, APRESS, 2006: 274 p.

[2] Crane D., Pascarello E., James D. Ajax In Action. New York, Manning Publications Co, 2006: 650 p.

[3] Dell'Aringa T. Bring your website to live with AJAX and DHTML. Amsterdam, Dynamic Zones International, 2006: 168 p.

[4] Zakas N. C., McPeak J., Fawcett J. Professional Ajax (Programmer to Programmer). New York, Wrox, 2006: 432 p.

[5] Galitz W. O. The Essential Guide to User Interface Design. New York, Wiley Computer Publishing, 2002: 760 p.

[6] Kuniavsky M. Observing the User Experience: A Practitioner's Guide to User Research. New York, Morgan Kaufmann Publishers, 2003: 576 p.

[7] Garrett J. J. Ajax: A New Approach to Web Applications. http://adaptivepath.com/publications/essays/archives/000385.php (2006.03.02)

[8] Garrett J. J. Why Ajax Matters Now. http://www.ok-cancel.com/archives/article/2005/09/why-ajax-matters-now.html (2006.03.04)

[9] Browser News: Statistics. http://www.upsdell.com/BrowserNews/stat.htm (2006.03.05)

[10] Web Code Compression. http://sniptools.com/tipstricks/web-code-compression (2006.03.20)

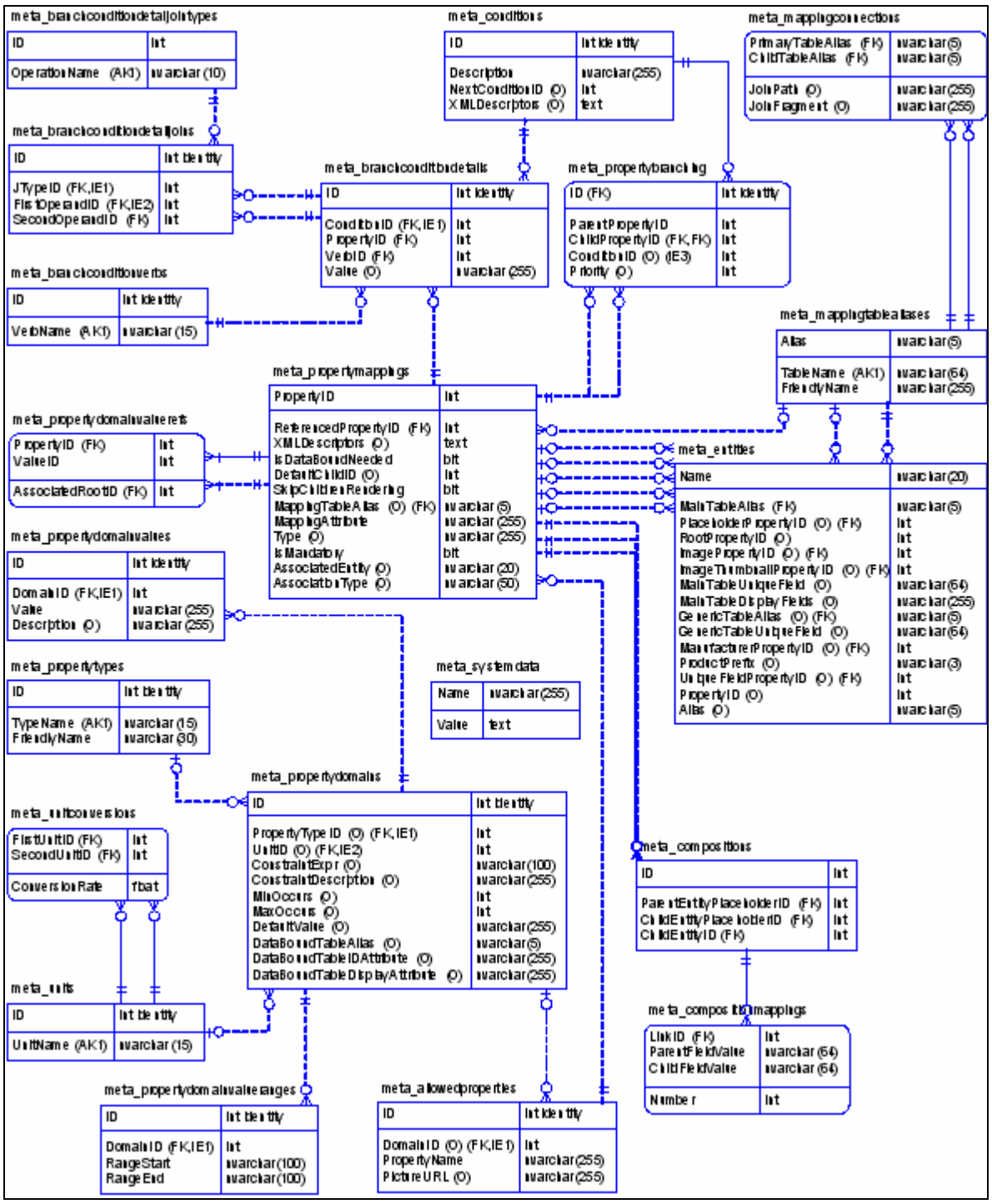
[11] JavaScript Optimization. http://home.earthlink.net/~kendrasg/info/js_opt (2006.03.25)

ANALYSIS OF POSSIBILITIES FOR APPLICABILITY OF AJAX TECHNIQUE IN WEB-BASED INFORMATION SYSTEMS

Classic web-based information systems are static, poorly structured and relatively awkward to use.
This paper analyses the influence, possibilities and potential ways of using AJAX in development of cutting-edge web-based information systems. Experience-based best practices are described.

10.2. "Mažylio" klausimynų IS KDB schema

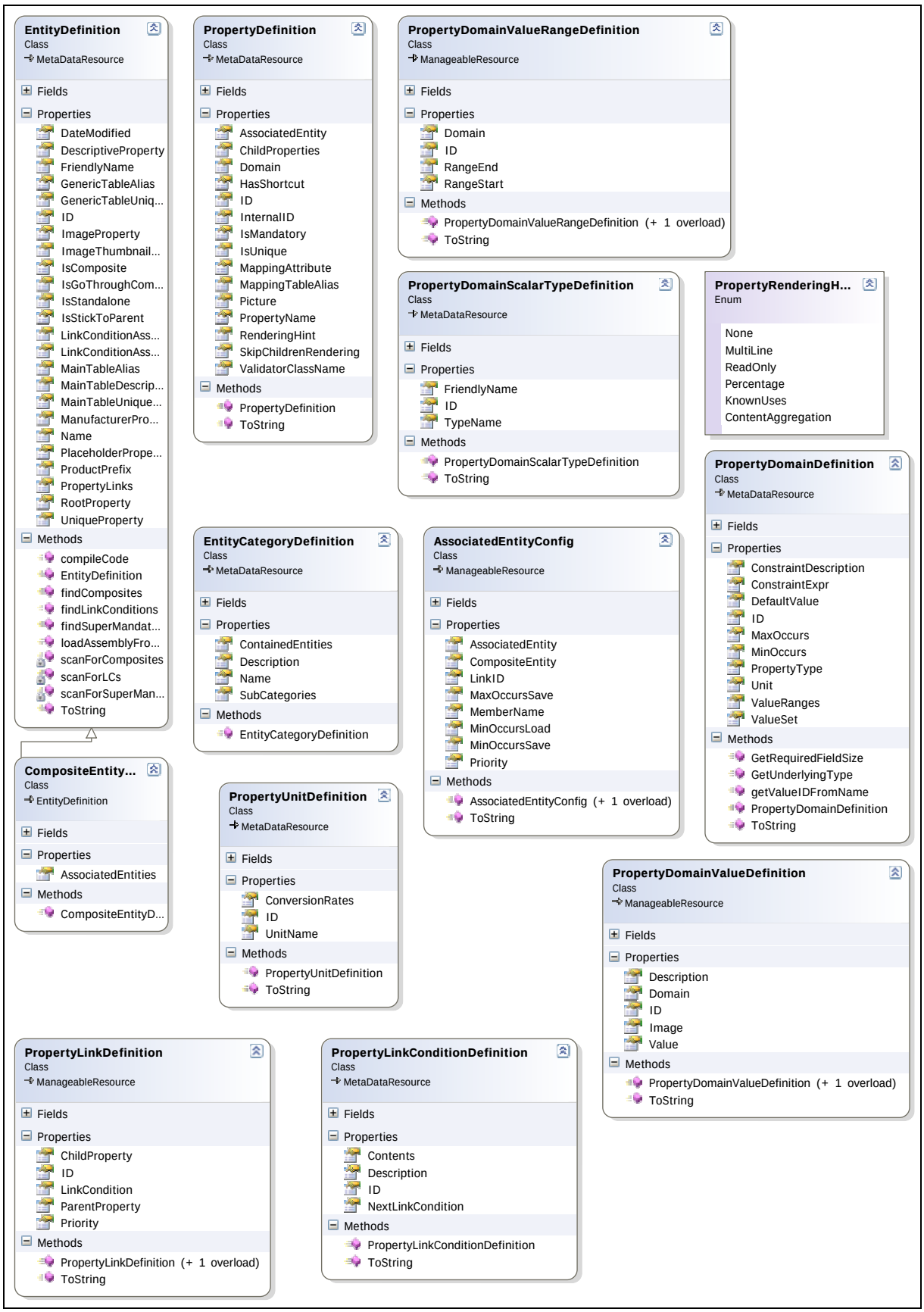
31 pav. pateikiama „Mažylis" IS KDB (t.y. metalygmens klausimų, klausimynų ir apribojimų aprašymo, žr. „Medžiai be lapų", 3.2.3.1).



31 pav. „Mažylio“ KDB schema

10.3. “Storulio” klausimynų IS metaklasės

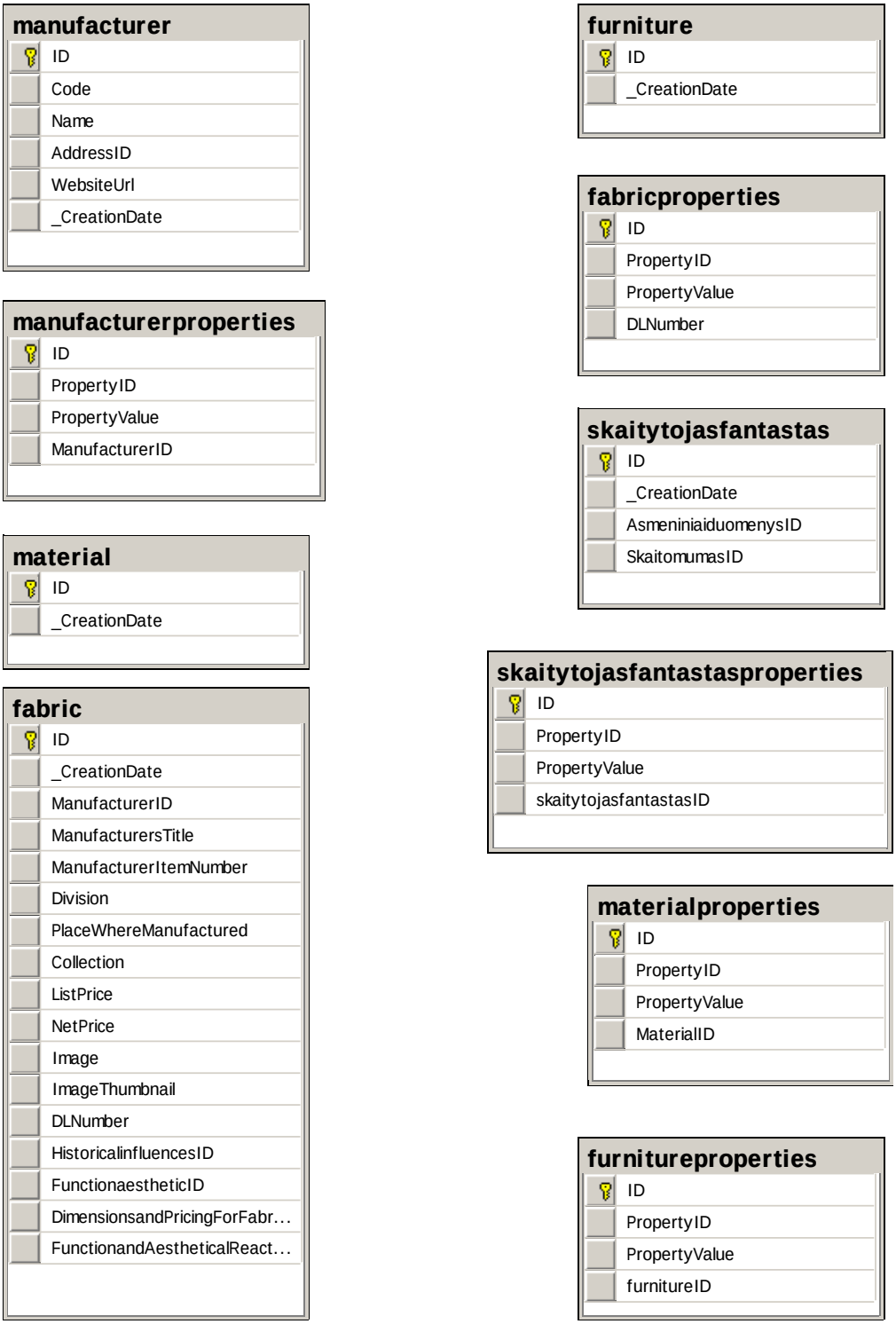
32 pav. pateikiamos “Storulis” IS metaklasės (t.y. klausimynų ir klausimų statinės struktūros klasės). Diagrama sugeneruota *Microsoft Visual Studio 2005 Class Diagram* įrankiu, neparodo ryšių tarp klasių.



32 pav. „Storulio“ metaklasių diagrama (Microsoft Visual Studio 2005)

10.4. “Storulio” RDB fragmentas

33 pav. pateikiama dalis “Storulio” klausimynų IS klausimynų egzempliorių DB (RDB, žr. „Projekcija“, 3.2.3.6). Į fragmentą įeina kelių skirtingų klausimynų lentelės iš produktų valdymo ir komercinių apklausų sferų.



33 pav. „Storulio“ RDB (fragmento) diagrama (Microsoft SQL Server 2005)

10.5. Įdiegimo aktas