VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ STUDIJŲ PROGRAMA

Trasavimo įgyvendinimas kiberfizinių–socialinių sistemų reikalavimų inžinerijoje

Traceability Realization in Requirements Engineering of Cyber–Physical–Social Systems

Magistro baigiamasis darbas

Atliko:	Irmantas Ivanauskas	(parašas)
Darbo vadovas:	dr. Audronė Lupeikienė	(parašas)
Darbo recenzentas:	dr. Vytautas Čyras	(parašas)

Vilnius – 2022

TURINYS

SANTRAUKA 2

SUMMARY 3

ĮVADAS 4

1. REIKALAVIMŲ TRASAVIMO YPATUMŲ ANALIZĖ KIBERFIZINĖSE–SOCIALINĖSE SISTEMOSE 8
   1.1. Kibernetinių–fizinių–socialinių sistemų tipai 8
      1.1.1. Kibernetinės–fizinės sistemos 8
      1.1.2. Kibernetinės–fizinės sistemos su žmogumi 9
      1.1.3. Kibernetinės–socialinės sistemos 9
      1.1.4. Kibernetinės–fizinės–socialinės sistemos 10
   1.2. Reikalavimų trasavimas 12
      1.2.1. Reikalavimų inžinerija 12
      1.2.2. Reikalavimų trasavimas 12
      1.2.3. Reikalavimų trasavimo ryšiai 14
      1.2.4. Trasavimo praktikos 16
      1.2.5. Reikalavimų trasavimo svarba 17
      1.2.6. Reikalavimų trasavimo problemos kibernetinėse–fizinėse–socialinėse sistemose 17
   1.3. Reikalavimų trasavimo metodai 19
      1.3.1. Trasavimo matricos 19
      1.3.2. Trasavimo sąrašai 20
      1.3.3. Trasavimo medžiai 20
      1.3.4. Kokybės funkcijų sklaida 21
      1.3.5. Trasavimas pagal identifikatorių 22
      1.3.6. Trasavimas pagal atributus 22
      1.3.7. Trasavimo metodai kibernetinėms–fizinėms–socialinėms sistemoms 23
   1.4. Skyriaus išvados 24

2. KOKYBĖS FUNKCIJŲ SKLAIDOS TAIKYMAS TRASAVIMUI ĮGYVENDINTI 26
   2.1. Trasuojami objektai 26
      2.1.1. Sistemos kūrimo fazės 26
   2.2. Trasavimo modelis 29
   2.3. Reikalavimai trasavimo modeliui 29
      2.3.1. Trasavimo objektai ir ryšiai 29
   2.4. Kokybės funkcijų sklaida 31
   2.5. KFS pranašumai prieš kitus trasavimo metodus 36
   2.6. KFS trūkumai ir ateities tyrimai 39
   2.7. Apibendrinimas 39

3. KFS PANAUDOJIMAS KURIANT IŠMANAUS MIESTO SISTEMĄ 40

4. REZULTATAI IR IŠVADOS 45
   4.1. Rezultatai 45
   4.2. Išvados 45

Santrauka

Hibridinės sistemos, sudarytos ir techninės, programinės įrangos bei sistemas jungiančio socialinio aspekto, vadinamos kibernetinėmis–fizinėmis–socialinėmis sistemomis. Tokios sistemos laikomos naujos kartos sistemomis. Kadangi tai naujos kartos sistemos, jų kūrimas nėra iki galo ištirtas.

Kuriant kibernetines–fizines–socialines sistemas, susiduriama su įvairiais sunkumais. Šiame darbe išskirta ir plačiau nagrinėta trasavimo problema. Kadangi šios naujos kartos sistemos susideda iš daugybės komponentų, sekti ryšius tarp jų pasidaro sudėtinga. Taip pat tokiose sistemose realizacija yra vykdoma ir autonominio komponento, ir visos sistemos kontekste.

Atlikus literatūros analizę, paaiškėjo, jog dabartiniai trasavimo būdai nesprendžia kai kurių sunkumų. Išanalizavus trasavimo matricas, sąrašus, medžius, trasavimą pagal identifikatorių ir atributus, nustatyta, kad nei vienas iš šių metodų netinka iškeltoms problemoms spręsti.

Darbe ieškota alternatyvaus metodo trasuoti reikalavimus kiberfizinėse–socialinėse sistemose. Iškeltoms problemoms spręsti buvo pasitelktas kokybės funkcijų sklaidos kokybės namas. Nors ši metodika nėra specifiškai skirta reikalavimams trasuoti, modifikuotas kokybės namas gali būti naudojamas šiuo tikslu. Darbe buvo pateiktos modifikacijos kokybės namui.

Metodas buvo patikrintas aprašant kibrnetinės–fizinės–socialinės sistemos trasavimo ryšius. Kaip atvejo pavyzdys pasirinkta išmaniojo miesto sistema, aprašyti ryšiai tarp skirtingų artefaktų bei nupiešti kokybės namai, vizualizuojantys ryšius tarp skirtingų reikalavimų lygmenų.

Darbe padarytos išvados, jog trasuojant reikalavimus kokybės namų pagalba, užtikrinama daugiau informacijos nei tradiciniais būdais ir ji yra atvaizduojama patogiau.

Summary

Hybrid systems, consisting of hardware, software and the social aspect, are called cyber–physical–social systems. Such systems are considered to be a new generation of systems. As these are new generation systems, their development has not been fully explored.

The development of cyber–physical–social systems faces various challenges. In this work, traceability problem is pointed out and discussed in more detail. Because these next–generation systems are made up of many components, it becomes difficult to keep track of the connections between them. Also in such systems, the implementation is performed in the context of both the autonomous component and the system as a whole.

An analysis of the literature has shown, current tracing methods do not solve the problems. Analysis of tracing matrices, lists, trees, tracing by identifier and attributes revealed that none of these methods are suitable for solving the problems raised.

An alternative method for tracing requirements in cyberphysical–social systems was sought. A quality house for the dissemination of quality functions was used to address the issues raised. Although this methodology is not specifically designed to track requirements, a modified quality house can be used for this purpose. The work presented modifications for a quality house.

The method was tested by describing the tracing relationships of a cyber–physical–social system. The smart city system was chosen as an example, the connections between different artifacts are described, and quality houses are drawn, visualizing the connections between different levels of requirements.

The paper concludes that tracing the requirements with the help of quality houses provides more information than traditional methods and it is displayed more conveniently.

Įvadas

Tyrimo objektas

Reikalavimų trasuojamumu vadiname galimybę nusakyti ir sekti reikalavimo gyvavimą nuo reikalavimo suformavimo iki įgyvendinimo sistemoje [GF94].

Naudojant tinkamas technikas, reikalavimų trasuojamumas suteikia daug naudos sistemos kūrėjams. Trasuojamumas leidžia pagerinti projekto valdymą, padarydamas jį lengviau prognozuojamu. Naudojantis trasavimo ryšiais, galima greitai pastebėti, kiek artefaktų bus paveikta atliekant pakeitimus, todėl lengviau įvertinti rizikas bei pakeitimo kainą. Kadangi reikalavimai būna sekami iki pat įgyvendinimo, projekto vadovas gali lengviau nustatyti, kokia projekto dalis jau yra baigta.

Kibernetinėse–fizinėse-socialinėse sistemose reikalavimų ir projektinių artefaktų kiekis yra didelis, jų aibė dinamiškai kinta, o įgyvendinimas yra susijęs per socialinio ir autonominio veikimo lygmenis. Tokiose sistemose trasavimo uždavinys tampa sudėtingas ir sunkiai sprendžiamas. Šiame darbe šis uždavinys yra apibrėžiamas kaip trasavimo problema, kuri ir bus nagrinėjama.

Temos aktualumas

Hibridinės sistemos, sudarytos iš techninės ir programinės įrangos, vadinamos kibernetinėmis–fizinėmis sistemomis. Prie jų prijungus socialinį aspektą, gauname kibernetines–fizines–socialines sistemas. Per pastaruosius dešimt metų tokių sistemų naudojimas, o taip pat sudėtingumas, ženkliai išaugo [Hor14]. Tobulėjančios technologijos (pavyzdžiui, 5G ryšys) sąlygos, kad ir toliau bus kuriamos vis didesnės ir sudėtingesnės sistemos. Kiberfizinės–socialinės sistemos charakterizuojamos šiomis savybėmis:

1. Turi didelį sudedamųjų dalių skaičių, kuris gali dinamiškai kisti;
2. Komponentų atributai nėra iš anksto nustatyti;
3. Sąveikų skaičius tarp komponentų gali būti labai didelis;
4. Komponentai gali veikti nepriklausomai;
5. Sistemos veikimas yra tikimybinis;
6. Sistema intensyviai sąveikauja su aplinka;
7. Operacijos tokioje sistemoje yra jautrios kontekstui.

Kibernetinių–fizinių–socialinių sistemų atsiradimas pakeitė santykį tarp žmonių, kompiuterių ir fizinės aplinkos [ZYL+16]. Nuolatinis poreikis didesniems duomenims, platesniam sistemų bendradarbiavimui bei lankstumui rodo, kad atsiranda naujos kartos informacinės sistemos [PZJ+16].

Kaip tokių sistemų pavyzdį būtų galima pateikti save vairuojančių mašinų būrį. Toks važiuojantis mašinų būrys ne tik turi laikytis kelių eismo taisyklių, bet ir laikytis optimaliais atstumais viena nuo kitos bei dalintis papildoma informacija (pavyzdžiui, apie ant kelio nuvirtusį medį ar pakelėje stovintį laukinį gyvūną) [Dre18]. Toks mašinų būrys pagerina ne tik važiavimo saugumą. Važiuojant tokiame automobilių būryje taip pat taupomi degalai – taigi, tausojama aplinka. Be to, koordinuotas automobilių judėjimas galėtų išspręsti ir kamščių keliuose problemą [JLW+15].

Kadangi tai naujos kartos sudėtingos sistemos, jų tyrimas yra aktualus uždavinys.

Nagrinėjama probleminė sritis buvo pasirinkta, nes sėkmingas reikalavimų trasavimas projekte gali praversti valdant projektą, užtikrinant procesų skaidrumą, tikrinant ir tvirtinant reikalavimus bei, vėliau, palaikant sistemą [KS09].

Temos naujumas

Kiberfizinių–socialinių sistemų kūrimas reikalauja pilno trasavimo, sujungiančio sistemai keliamus uždavinius, reikalavimus, projektavimą bei realizaciją. Sunku valdyti trasavimą, kuomet modeliai stokoja sinchronizacijos, trūksta sąryšio tarp tekstinių reikalavimų, suinteresuotų šalių iškeltų tikslų sistemai bei pačios sistemos komponentų, turimos mažos galimybės plėsti sistemą. Šios problemos nėra tinkamai išnagrinėtos.

Reikalavimų trasavimo problemai spręsti yra sukurta įvairių įrankių ir metodikų, tačiau jos turi minusų. Lentelė, kurioje nurodomi ryšiai tarp sistemos reikalavimų ir įgyvendinimo, yra vadinama trasavimo matrica. Tačiau toks metodas neatitinka kiberfizinių–socialinių sistemų inžinerijos poreikių. Taip sekant reikalavimus, susidaro labai daug jungčių tarp atskirų sistemos elementų, todėl sekti reikalavimus pasidaro labai sudėtinga ir brangu [CCM03].

Rankiniu būdu sekami reikalavimai taip pat turi trūkumą, kad juos reikia nuolat atnaujinti, o atnaujinus vieną ryšį reikia atnaujinti ir kitus, kurie yra susiję su šiuo reikalavimu. Dėl šios priežasties, dažnai rankiniu būdu sekami reikalavimai tampa nebesuderinti su esamais reikalavimais ir jų realizacija. Taip pat tokiame trasavime atsiranda klaidų [KS09].

Trasavimo automatizavimui kuriami karkasai. Vienas iš jų yra TORUS. Vis dėlto, nors šis karkasas buvo kurtas automatizuoti reikalavimų trasavimą, tačiau pritaikyti jo realaus pasaulio sistemomis su kintančia struktūra ir pilnai padengti sistemos reikalavimų trasavimą, nepavyko [DSH16].

Tikslas, uždaviniai ir laukiami rezultatai

Tikslas

Ištirti kiberfizinių–socialinių sistemų ir jų kūrimo specifiką ir tuo remiantis pasiūlyti, kaip spręsti reikalavimų trasavimo uždavinį.

Uždaviniai

1. Išnagrinėjus kiberfizinių–socialinių sistemų ir jų kūrimo proceso ypatumus, apibrėžti trasavimo problemą ir charakterizuoti jos sudėtingumą.
2. Išanalizuoti mokslinėje literatūroje pasiūlytus trasavimo problemos sprendimo būdus.
3. Pasiūlyti metodą tikslų, reikalavimų, projektinių ir realizacinių artefaktų trasavimo problemai išspręsti kuriant kiberfizines–socialines sistemas.
4. Patikrinti pasiūlytą metodą, pritaikant jį konkrečiai kiberfizinių–socialinių sistemų klasei.
5. Pateikti metodines ir technologines rekomendacijas metodui įgyvendinti.

Laukiami rezultatai

1. Nustatyti kiberfizinių–socialinių sistemų trasavimo uždavinio ypatumai ir esminės charakteristikos.
2. Nustatyta aibė kiberfizinių–socialinių sistemų reikalavimų, projektavimo ir realizacinio lygmens elementų, kurie gali būti jungiami trasomis.
3. Nustatyta, kaip spręsti trasavimo problemą kibernetinėse–fizinėse–socialinėse sistemose, kurios sudarytos iš heterogeninių dalių.

Darbo struktūra

Šį darbą sudaro trys numeruojami skyriai. Pirmajame skyriuje atliekama literatūros šaltinių analizė. Analizuojama, kas yra kibernetinės–fizinės–socialinės sistemos bei kokios jų ypatybės, taip pat aiškinamasi reikalavimų trasavimo ypatumai bei metodai. Skyriaus pabaigoje atliekamas skirtingų trasavimo metodų palyginimas.

Antrajame skyriuje pateikiami metodiniai siūlymai, kaip pasitelkiant kokybės funkcijų sklaidos metodą būtų galima įgyvendinti trasavimą kibernetinėse–fizinėse–socialinėse sistemose, sukuriamas trasavimo modelis bei metodas palyginamas su tradiciniais trasavimo metodais.

Trečiajame skyriuje pasiūlytas metodas pritaikomas išmanaus miesto sistemos kūrime. Pateikiamas pavyzdys, kaip naudojantis kokybės namais galima trasuoti reikalavimus didelėse kibernetinėse–fizinėse–socialinėse sistemose.

Darbo pabaigoje pateikiami darbo rezultatai bei padarytos išvados.

Darbo atlikimo metodika

Tyrimas vykdomas jungiant induktyvinę ir deduktyvinę tyrimo strategijas. Induktyvinė tyrimo strategija yra tinkama sprendžiant problemas naujose srityse, kai nėra išsamios teorijos, kurią būtų galima taikyti. Kiberfizinių–socialinių sistemų kūrimo teorija kaip tik ir yra tokios srities pavyzdys. Taigi, tyrimo hipotezė buvo suformuluota induktyvinio tyrimo metu, įgijus naujas žinias nagrinėjant literatūrą, stebint realias kiberfizines–socialines sistemas ir visa tai apibendrinus. Taikant deduktyvinę tyrimo strategiją darbo rezultatai buvo gauti nuosekliai, per kelis žingsnius, ankstesnių argumentų išvadas verčiant paskesnių argumentų prielaidomis.

Teorinių rezultatų kūrimas grindžiamas bibliotekiniu tyrimu, stebėjimu, indukciniu metodu, dedukciniu metodu ir argumentavimo tyrimu. Argumentavime taikomas Aristotelio [Lea80] ir Toulmino [Tou03] argumento modelis

1. Reikalavimų trasavimo ypatumų analizė kiberfizinėse–socialinėse sistemose

1.1. Kibernetinių–fizinių–socialinių sistemų tipai

Kibernetinės–fizinės–socialinės sistemos gali būti laikomos nauja visos mus supančios kompiuterijos tobulėjimo faze. Be šios fazės taip pat galima išskirti kibernetines–fizines bei kibernetines–socialines, o taip pat kibernetines–fizines su žmogumi sistemas [**survey**].

Toliau skyriuje bus apžvelgiamos visos šios fazės.

1.1.1. Kibernetinės–fizinės sistemos

Kibernetinės–fizinės sistemos, kuriose operacijos yra stebimos, koordinuojamos, valdomos ir integruojamos bendraujant tarp skirtingų fizinių komponentų. Tokia sistema gali būti ir maža, ir ypatingai didelė, apjungianti didelį kiekį elementų [RLS+10]. Šios sistemos buvo plačiai pritaikytos išmaniojoje gamyboje, transporte, skubios pagalbos tarnybose ir kitose srityse [GPG+14].

Kadangi tai yra kibernetinės–fizinės sistemos, tai jos sudedamosios dalys gali būti perskirtos į dvi šeimas – kibernetinę ir fizinę [RCS+09]. Kibernetinės šeimos komponentams priskirkime:

1. Duomenų saugyklos – tai komponentai, kuriuose yra saugoma informacija. Priklausomai nuo sistemos sudėtingumo, tai gali būti tik pasyvi duomenų saugykla arba saugykla, kurioje per kitus sistemos komponentus gali būti pasiekiami, redaguojami ar įrašomi nauji duomenys.

2. Skaičiavimo komponentai – šie komponentai atlieka veiksmus su turimais duomenimis gaunamais iš kitų komponentų.

3. Duomenų įeities–išeities interfeisai – tai komponentai apdorojantys duomenys, gaunamus iš fizinio pasaulio. Pavyzdžiui, tai galėtų būti išmaniojo jutiklio programinė įranga (tvarkyklė).

Fizinės šeimos komponentai gali būti šie:

1. Šaltiniai – tai komponentai, kurie tiekia energiją kitiems komponentams.

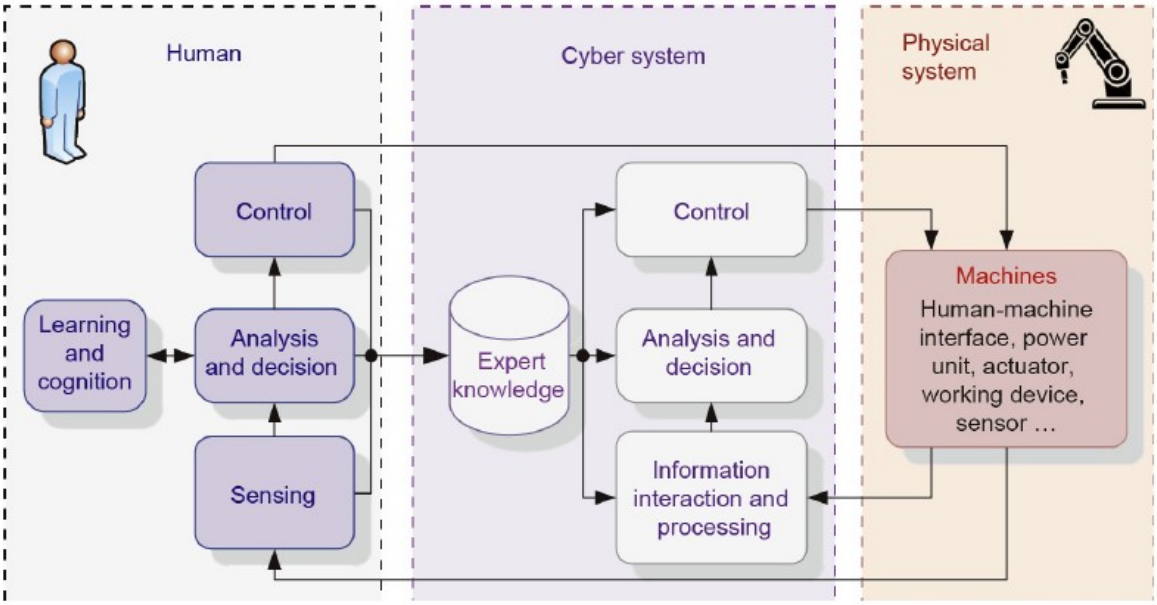
2. Energijos talpykla – komponentas, kuriame kaupiama energija, o tą energiją gali panaudoti kiti komponentai.

3. Fiziniai davikliai – tai yra jutikliai, kurie sąveikauja su aplinka fiziniame lygmenyje.

Šių abiejų šeimų veikimui kartu reikalingi kibernetiniai–fiziniai interfeisai, kurie apjungia tiek kibernetinę, tiek fizinę erdvę bendram darbui. Išskiriami du interfeisų tipai: P2C (fizinis kibernetiniui) bei C2P (kibernetinis fiziniui) [RCS+09].

1.1.2. Kibernetinės–fizinės sistemos su žmogumi

Tarp kibernetinės–fizinės ir kibernetinės–fizinės–socialinės sistemos yra tarpinė fazė, kuomet kibernetinė–fizinė sistema veikia dalyvaujant žmogui [YMZ20]. Kibernetinės–fizinės sistemos atveju mes turėjome tik fizinius ir programinius komponentus, kurie tarpusavyje sąveikauja interfeisais. Šiuo atveju, sąveiką turime ne tik tarp sistemos komponentų, bet ir žmogaus, valdančio tos sistemos darbą. Žmogus tampa ne tik pasyviu informacijos iš sistemos gavėju, tačiau ir pačios sistemos dalimi, kurdamas reikalingą informaciją ir pritaikydamas savo žinias [KM18].



1 pav. Sąveika tarp žmogaus ir fizinės–kibernetinės sistemos [KM18]

Tokios sistemos gali būti klasifikuojamos į tris kategorijas [MSL+13]:

1. Sistema, kurią žmogus tiesiogiai valdo.
2. Sistema, kurioje žmogus pasyviai stebi ir, reikalui esant, imasi reikalingų veiksmų.
3. Abiejų aukščiau išvardintų hibridas.

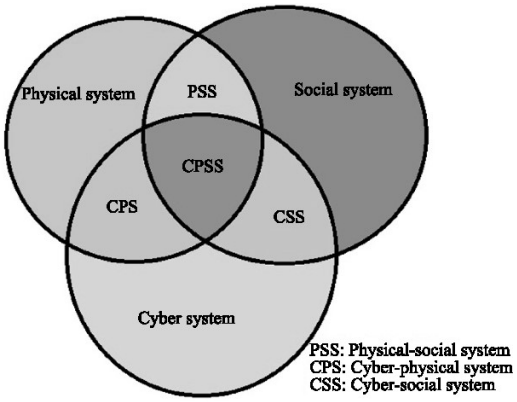
1.1.3. Kibernetinės–socialinės sistemos

Kibernetinės–socialinės sistemos kreipia dėmesį į žmonių socialinių įpročių ir santykiu analizę, kad suteiktų tinkamas informacines paslaugas, kurios galėtų pagerinti žmonių gyvenimo kokybę. Yra skirtingų tokių sistemų pavyzdžių. Socialiniai tinklai yra priskiriami kibernetinėms–socialinėms sistemoms. Žmonės naudojasi socialiniais tinklais, kad pasidalintų savo nuomonėmis, nuotraukomis ir t. t. Yra sukurta daugybė skirtingų socialinių tinklų programų [ZYL+16].

Kibernetinėms–socialinėms sistemoms taip pat galima priskirti e–valdžios sistemas, taip pat įvairius aukcionus [CI10].

1.1.4. Kibernetinės–fizinės–socialinės sistemos

Kibernetinės–fizinės–socialinės sistemos apjungia jutiklinius prietaisus, tinklą tarp jų, skaičiavimo serverius bei kitus fizinius objektus, kurie yra reikalingi palengvinti žmonių gyvenimą [ZYL+16]. Iš tiesų kibernetines–fizines–socialines sistemas galime laikyti U formos sistemomis, kurios vykdo tiek kibernetinių–fizinių, tiek kibernetinių–socialinių sistemų funkcijas [ZYL+16].

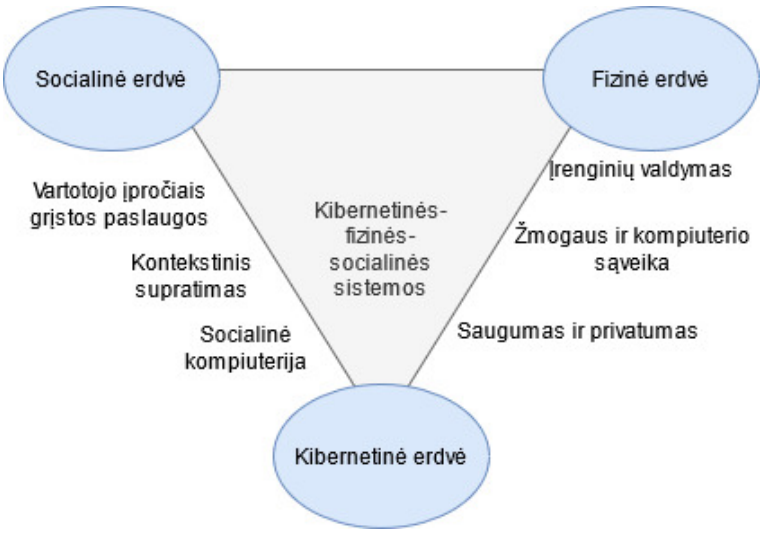


2 pav. Skirtingų sistemų palyginimas [ZYL+16]

Kibernetinių–fizinių–socialinių sistemų pranašumai pasirodo kai kuriais atvejais. Priešingai negu kibernetinės–fizinės arba kibernetinės–socialinės sistemos, kibernetinės–fizinės–socialinės sistemos gali suderinti abiejų prieš tai minėtų sistemų galimybes bei naudoti informaciją, pasiekiamą kibernetinėje, fizinėje ir socialinėje erdvėse bei pateikti skaičiavimus, kurie yra orientuoti į žmogų. Pagrindinė šių sistemų užduotis yra patenkinti žmonių socialinius poreikius ir reaguoti į fizinį pasaulį. Taip pat mokslininkai bando pritaikyti kibernetinių–fizinių–socialinių sistemų technologijas, kad palengvintų sudėtingų darbų atlikimą fiziniame pasaulyje ir taip pagerintų darbo našumą [ZYL+16].

Sparčiai plėtojant šias sistemas, jų pritaikymas randamas įvairiose srityse. Įprastai kibernetinės–fizinės–socialinės sistemos pasitelkiamos išmaniųjų namų, išmaniųjų transportavimo sistemų, išmaniųjų medicininių prietaisų, išmaniųjų miestų bei kitų sistemų kūrimui [ZYL+16].

Kibernetinės–fizinės–socialinės sistemos turi apimti įvairius jutiklius, tinklo bei skaičiavimo įrenginius bei kitus fizinius prietaisus, kad galėtų pagerinti žmogaus gyvenimą. Šių sistemų architektūra turėtų susidėti iš trijų dimensijų: fizinės, kibernetinės ir socialinės erdvės. Priešingai negu kibernetinėse–fizinėse sistemose, šiuo atveju žmogaus socialinė veikla turi būti glaudžiai susijusi su kibernetine ir fizine erdve.



3 pav. Fizinės, socialinės ir kibernetinės erdvės ryšys

Apibendrindami galime teigti, jog kibernetinės–fizinės–socialinės sistemos charakterizuojamos šiomis savybėmis:

1. Turi didelį sudedamųjų dalių skaičių, kuris gali dinamiškai kisti;
2. Sistema yra decentralizuota ir geografiškai išskirstyta ir nėra griežtų sistemos ribų;
3. Sąveikų skaičius tarp komponentų gali būti labai didelis;
4. Komponentai gali veikti nepriklausomai;
5. Sistemos resursai valdomi sudėtingomis strategijomis, kad būtų aukštas saugumo, integralumo bei palaikomumo lygis;
6. Sistema intensyviai sąveikauja su aplinka;
7. Operacijos tokioje sistemoje priklauso nuo konteksto (konkrečios situacijos, valstybės įstatymų ir t. t.) [Hor14]

1.2. Reikalavimų trasavimas

1.2.1. Reikalavimų inžinerija

Reikalavimas yra apibrėžiamas kaip savybių, atributų, funkcijų ir sistemos elgsenos aibė, reikalinga produktui pasiekti keliamus tikslus [Car00]. Reikalavimais yra apibrėžiama, ką sistema turi daryti. Reikalavimus sudaro sistemos tikslai, gyvavimo ciklai, operacinės būsenos, apribojimai bei interfeisai su kitomis sistemomis [Car00]. Reikalavimai dokumentuoja visų suinteresuotų šalių poreikius ir lūkesčius [Car00]. Literatūroje apibrėžiama, kad reikalavimai turi atspindėti visus suinteresuotų šalių poreikius, dėl kurių bus vykdomi sistemos pakeitimai [Ver05]. Reikalavimus gali sudaryti verslo, vartotojų, funkciniai, nefunkciniai, diegimo, našumo bei procesų reikalavimai [Car00]. Tinkamas specifikacijos paruošimas yra priklausomas nuo reikalavimų valdymo [Car00].

Reikalavimų valdymui pasitelkiama reikalavimų inžinerija. Reikalavimų inžinerija yra procesas, kurio metu siekiama nustatyti, ar sistema patenkina tikslus, dėl kurių ji yra kuriama. Tai yra daroma nurodant suinteresuotas šalis, jų poreikius ir juos dokumentuojant taip, kad būtų galima reikalavimus analizuoti ir įgyvendinti [Fin00].

Dokumentai, sukurti reikalavimų inžinerijos metu, viso produkto kūrimo metu turėtų būti trasuojami su aiškiai nustatytais ryšiais tarp artefaktų [Wie95]. Reikalavimai turėtų būti būti trasuojami nuo žemiausio iki aukščiausio lygmens [Car00].

Išskiriamos 5 fazės reikalavimų inžinerijoje [Fin00]:

1. Reikalavimų iškėlimas
2. Reikalavimų analizė ir modeliavimas
3. Reikalavimų dokumentavimas
4. Reikalavimų validavimas ir verifikavimas
5. Reikalavimų valdymas

Reikalavimų validavimui, verifikavimui ir valdymui reikalinga atsekti reikalavimus ir jų ryšius per skirtingus artefaktus. Toks atsekamumas yra vadinamas reikalavimų trasavimu. Apie reikalavimų trasavimą plačiau bus aprašyta kitame skyrelyje.

1.2.2. Reikalavimų trasavimas

Programos kūrimo ir palaikymo metu yra sukuriami įvairūs artefaktai. Šie artefaktai jungiami trasavimo ryšiais. Programos gyvavimo metu, šiuose artefaktuose gali atsirasti pakeitimų. Svarbu

šiuos pakeitimus peržiūrėti ir sekti tam, jog būtų sumažinta neigiamų padarinių sistemai rizika [STD21].

Reikalavimų trasavimas gali būti apibrėžiamas kaip reikalavimų įgyvendinimo sekimas tarp skirtingų sistemos įgyvendinimo fazių (artefaktų).

Trasavimas gali būti įgyvendinamas abejomis kryptimis. Atsekamumu į priekį vadiname procesą, kurio metu galime nustatyti, kuriems sistemos artefaktams reikalavimas padarė įtaką. Atsekamumu atgal vadiname procesą, kurio metu iš artefaktų nustatome, iš kokio reikalavimo kilo pokyčiai [LS01].

Reikalavimų trasavimas yra reikšminga sistemos kūrimo ciklo dalis, padedanti efektyviai valdyti reikalavimų pasikeitimus [AY15]. Reikalavimų trasavimas reikalingas tam, jog galėtume valdyti visų suinteresuotų grupių – projekto vadovų, analitikų, dizainerių, vartotojų ir kt. – reikalavimus, jų pokyčius bei prioritetus. Reikalavimų trasavimas padeda nustatyti, kaip reikalavimai susisieja per artefaktus viso programinės įrangos kūrimo metu [RJ01].

Literatūroje išskiriami 4 reikalavimų trasavimo tikslai:

1. Nustatyti ar reikalavimas yra reikalingas
2. Nustatyti, kaip projektavimo sprendimai padarys įtaką reikalavimo įgyvendinimui
3. Nustatyti, ar visi reikalavimai yra įgyvendinti
4. Nustatyti, ar visi projektavimo elementai yra būtini [KKK+08]

Sistemos projektavimo metu, reikalavimai dažnai keičiasi. Ypač dažnai tai nutinka, kuomet sistema yra kuriama judriuoju būdu [Jai07]. Šioje fazėje, reikalavimų trasavimas padeda sekti ir numatyti, kokią įtaką sistemai padarys norimi atlikti pakeitimai dar prieš įgyvendinant reikalavimą [AY15].

Testavimo procedūros taip pat gali būti trasuojamos su reikalavimais. Tokių būdu, trasavimas padeda sukurti bei modifikuoti testavimo procedūras [RJ01].

Pokyčiai sistemoje gali vykti ir po sistemos išleidimo. Klaidos, trikiai, pasikeitę procesai ar poreikiai gali sukurti naujus reikalavimus. Tokie sistemos pokyčiai vadinami sistemos evoliucija [LR03].

Tinkamai naudojant reikalavimų trasavimą, gali būti tiksliau prognozuojami įgyvendinimo kaštai bei laikas [RJ01].

Trasavimas turi būti valdomas naudojantis trasavimo sistemomis.

1.2.3. Reikalavimų trasavimo ryšiai

Literatūroje aptinkami 6 dažniausiai naudojami trasavimo ryšiai:

1. Reikalavimo–šaltinio
2. Reikalavimo–pagrindimo
3. Reikalavimo–žmogaus
4. Reikalavimo–reikalavimo
5. Reikalavimo–komponento
6. Reikalavimo–verifikacijos [Lei01].

4 pav. Trasavimo ryšiai

Apžvelkime šiuos ryšius.

Reikalavimo–šaltinio ryšys apibrėžia šaltinį, pagal kurį yra sukurtas reikalavimas. Šaltinis gali būti suinteresuotoji šalis, standartas arba koks nors kitas dokumentas [Ram12].

Ši informacija gali praversti validuojant, analizuojant, specifikuojant ir prioritetizuojant reikalavimus. Žinodami šią informaciją, galime nustatyti, ar reikalavimas gali būti pakeistas be kliento patvirtinimo ir ar reikalavimas yra galimas pakartotinai panaudoti. Taip pat ši informacija gali padėti kurti sistemų architektūras, kurių nereikės perkurti kiekvieną kartą keičiantis reikalavimams [Lei01].

Reikalavimo–pagrindimo ryšys apibrėžia priežastis, kodėl reikalavimas egzistuoja. Šis ryšys taip pat yra laikomas reikalavimo tikslu [Lei01].

Pagrindimo informacija yra svarbi analizuojant ir tikrinant, ar reikalavimas yra validuojamas ir išbaigtas. Pagrindimo ryšys leidžia nustatyti reikalavimo prioritetą, tikimybę reikalavimo pasikeitimo taip pat tai yra naudinga informacija keičiant ir pakartotinai panaudojant reikalavimus [Lei01].

Reikalavimo–žmogaus ryšys nustato, koks asmuo specifikavo konkretų reikalavimą. Ryšys turėtų nurodyti tiek žmogaus vardą, tiek jo pareigas. Tai yra reikalinga tam, kad konkrečiam žmogui palikus projektą, į jo vietą, tikėtina, ateis žmogus su panašiomis žiniomis [Lei01].

Šis ryšys reikalingas, nes žinant žmones, kurie yra reikalavimo autoriai, galima greitai gauti reikalingą informaciją apie reikalavimą tiesiog klausiant jų. Kadangi dažnai projektuose visa

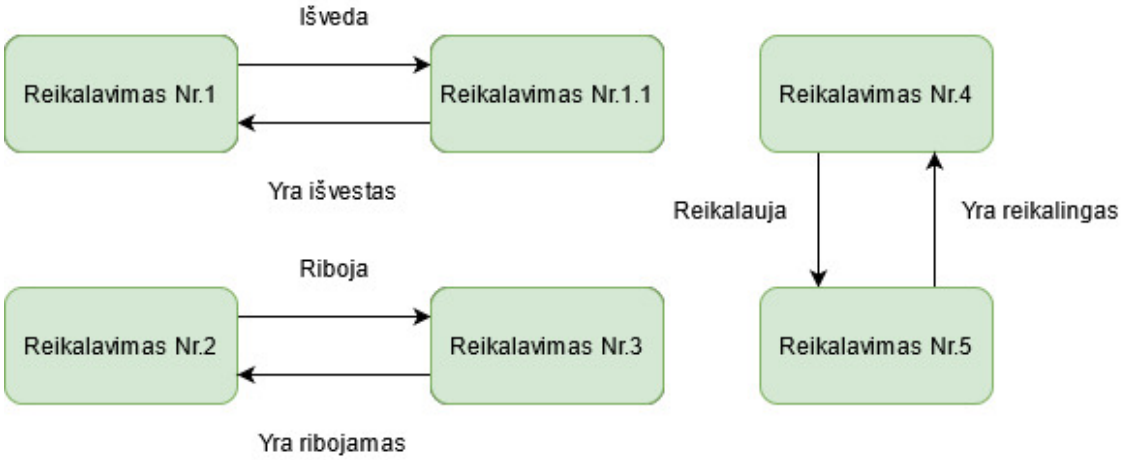
informacija nėra dokumentuojama, reikalavimo autorius gali būti vienintelis informacijos šaltinis [Lei01].

Reikalavimo–reikalavimo ryšis turi tris ryšio potipius.

1. Išveda/yra išvestas nurodo, jog reikalavimas detalizuoja kitą reikalavimą. Pavyzdžiui, jei reikalavimas Nr.1 specifikuoja, jog sistemoje duomenys turi būti šifruojami, tai reikalavimas Nr.1.1 turėtų apibrėžti šifravimo algoritmą. Tokiu būdu, reikalavimas 1.1 būtų išvestas iš reikalavimo 1. Šis ryšys reikalingas tam, jog galėtume garantuoti, kad visi aukšto lygio reikalavimai turi detalizuotus žemesnio lygio reikalavimus. Ši informacija yra naudinga, kuomet sistema yra palaikoma arba keičiasi aukštesniojo lygio reikalavimai [Lei01].

2. Riboja/yra ribojamas ryšys reiškia, kad jei reikalavimas Nr.2 specifikuoja, kad sistemoje turi būti rodomi kokie nors realieji skaičiai, o reikalavimas Nr.3 specifikuoja, kad sistemoje gali būti rodoma tik 12 skaitmenų skaičiai, tai reiškia, jog reikalavimas Nr.2 yra ribojamas reikalavimo Nr.3. Ši informacija padeda nustatyti, kurie reikalavimai negali būti keičiami arba įgyvendinami dėl kitų reikalavimų [Lei01].

3. Reikalauja/yra reikalingas ryšys nurodo, kokie reikalavimai reikalauja kitų reikalavimų įgyvendinimo. Šis ryšys yra reikalingas tam, kad per klaidą nebūtų panaikinti arba pakeisti reikalavimai, kurie yra būtina sąlyga kitiems reikalavimams įgyvendinti [Lei01].



5 pav. Reikalavimo–reikalavimo ryšių pavyzdžiai

Reikalavimo–komponento ryšys nurodo ryšį tarp reikalavimo ir komponento, kuris suprojektuotas to reikalavimo įgyvendinimui. Šis ryšys gali veikti į abi puses – ir nuo reikalavimo į komponentą ir atvirkščiai (nuo komponento į reikalavimą).

Reikalavimo–komponento ryšys padeda nustatyti, ar nei vienas reikalavimas nėra prarastas ir ar visi reikalavimai yra suprojektuoti, o iš to galime nustatyti, ar visi reikalavimai yra įgyvendinami

sistemoje. Taip yra ypač svarbu kompleksiškose sistemose bei sistemose, kuriose labai svarbus saugumas.

Šį informacija taip pat yra labai svarbi valdant projektą, nes galima nustatyti sistemos kūrimo tvarkaraštį, identifikuoti rizikingiausius komponentus, taip pat surinkti komandas pagal tai, kokių įgūdžių reikia konkretiems komponentams įgyvendinti. Pagal šią informaciją taip pat galima nustatyti sistemos baigtumo lygį bei numatyti būsimų pakeitimų kainą. Ši informacija taip pat padeda nustatyti priklausomybes tarp komponentų, kad yra ypač svarbu, jei komponentai kuriami skirtingose komandose ar net kompanijose [Lei01].

Paskutinis ryšys yra reikalavimo–verifikavimo. Šis ryšys nustato artefaktus, kurie realizuoja reikalavimus. Toks ryšys patvirtina, kad sistema įgyvendina jai keltus reikalavimus. Taip pat šiuo ryšiu galima nustatyti, kuriuos testavimo scenarijus reikia pertestuoti, įvykdžius naujus sistemos pakeitimus [Lei01].

1.2.4. Trasavimo praktikos

Yra apibrėžiamos šešios trasavimo praktikos [She14]:

1. Planuoti trasavimą: svarbu, jog projekto vadovai iškart suplanuotų trasavimą nuo pirminių projekto fazių ir pasirinktų tinkamą trasavimo modelį. Modelis turėtų užtikrinti visų artefaktų ir jų ryšių trasavimą [She14].

2. Pasirinkti tinkamus įrankius: trasavimas turėtų būti vykdomas naudojantis automatizuotais įrankiais, kuriais naudojantis būtų galima nustatyti trasavimo ryšius bei matyti visą su trasavimu susijusią informaciją [She14].

3. Kurti trasavimą inkrementiškai: tinkamų įrankių naudojimas suteikia galimybę suinteresuotoms šalims turit trasavimą ryšius inkrementiškai su kasdieniu darbu. Dauguma atveju, kuomet trasavimas pradedamas kurti ne nuo projekto pradžios, susiduriama su sunkumais, kuomet trasavimas sukuriamas nepilnai arba netiksliai, o tai trukdo tolesniam projekto vystymui [She14].

4. Modeliuoti trasavimo užklausas: užklausos padengia pagrindines veiklas, tokias kaip rasti visus reikalavimus, kurie susiję su konkrečiu nepavykusiu testu ar kt. Užklausos veikia kaip filtrai, eliminuojantys nereikalingus artefaktus ir ryšius [She14].

5. Vizualizuoti trasavimo pjūvius: vietoje trasavimo matricų naudoti kitokius vizualizavimo metodus, kuriuose būtų matomi visi tiesioginiai ir netiesioginiai artefaktai medžio struktūroje [She14].

6. Nuolat vertinti trasavimą: vizualiai pateikti trasavimo progresą visoms projektu suinteresuotoms šalims, kad jie galėtų tinkamai stebėti progresą [She14].

1.2.5. Reikalavimų trasavimo svarba

Reikalavimų trasavimo nebuvimas arba prastas įgyvendinimas gali turėti padarinių projekto rezultatams, nes žmonės, palikdami komandą, išsineša ir savo turimas žinias, o tai ypač aktualu šiandieniniame IT pasaulyje, kuomet didelė specialistų kaita [KS09]. Taip pat tai gali sukurti komunikacijos, tarpusavio supratimo klaidų, padaryti įtaką priimamų sprendimų kokybei, o tai veda į visos sistemos kokybės prastėjimą bei programos kūrimo proceso kainos neprognozuojamumą [DP98]. Taip pat, reikalavimų trasavimas yra svarbus versijų kontrolei ir artefaktų konfigūracijos valdymui [KS09]. Dėl šių priežasčių, reikalavimų trasavimas tampa būtinybe projektuose, kuriuose svarbu užtikrinti, jog visi užsakovo reikalavimai yra patenkinti ir darbas yra baigtas [SHM+98].

Egzistuoja ir ryšys tarp reikalavimų trasavimo bei gebėjimų brandos modelio (CMM) [KS09]. Pagal CMM modelį, antrojo brandos lygio įmonės turi trasuoti savo reikalavimus. Aukštesniuose lygiuose taip pat yra poreikis trasoms tam, kad būtų apibrėžtas aiškus ir prognozuojamas procesas [KS09].

Reikalavimų trasavimo praktikos diegimas gali brangiai kainuoti organizacijai. Reikia papildomai planuoti šiuos darbus, kad praktika būtų įdiegta. Pradinis biudžetas, gali padidėti net keletą kartų lyginant su projektais, kuriuose trasavimas nėra diegiamas. Reikia apmokyti darbuotojus, taip pat papildomai gali kainuoti įvairūs įrankiai trasavimui įgyvendinti. Vis dėlto, trasavimas atperka šią kainą padidėjusia produkto kokybe bei sumažintomis išlaidomis sistemos palaikymui [KS09].

1.2.6. Reikalavimų trasavimo problemos kibernetinėse–fizinėse–socialinėse sistemose

Šio darbo tikslas yra išanalizuoti reikalavimų trasavimo problemas konkrečiai kibernetinėse–fizinėse–socialinėse sistemose ir rasti toms problemoms sprendimo būdų. Kibernetinės–fizinės–socialinės sistemos įprastai būna sudarytos iš išskirstytų kompiuterių, išmaniųjų periferinių įrenginių ir socialinio aspekto. Nors reikalavimų trasavimas jau yra išvystyta programų inžinerijos disciplina, kibernetinės–fizinės–socialinės sistemos jai pateikia naujų iššūkių [KM14].

Sistemose, kurios yra kompleksiškos, o saugumo reikalavimai dideli, reikalavimų trasavimas vis dar išlieka problema, o technikos, naudojamos įprastiniam reikalavimų trasavimui, šioje sistemų klasėje netinka [Fin12]. Išskirstytos sistemos paprastai reiškia, jog kontrolė yra decentralizuota, todėl yra sunku sekti konkretaus reikalavimo įgyvendinimą visuose komponentuose, kurie sudaro sistemą [PE12].

Didelės apimties sistemos gali turėti tūkstančius tiek funkcinių, tiek nefunkcinių reikalavimų [WRS09]. Toks kiekis reikalavimų sukuria ir didelį kiekį ryšių tarp reikalavimų. Literatūroje randama išvada, jog tradiciniai reikalavimų trasavimo įrankiai nėra pritaikyti sistemoms, kuriose yra tūkstančiai reikalavimų [BS14].

Kadangi kibernetinės–fizinės–socialinės sistemos yra sistemos, sudarytos iš kitų sistemų, kurios yra pakartotinai panaudojamos, sistemos kūrimas dažniausiai neprasideda nuo nulio. Dėl to, reikalavimų trasavimo metu reikalinga susitvarkyti su „juodų dėžių“ interfeisais bei reikalavimų hierarchijomis [DSH16].

1.3. Reikalavimų trasavimo metodai

Literatūroje yra aprašomi skirtingi reikalavimų trasavimo metodai [BLT+12]. Šiame skyrelyje jie bus apžvelgti. Apžvalgoje bus siekiama patikrinti, ar aprašyti metodai atitinka šiuos kriterijus: turi būti galima apdoroti didelius reikalavimų kiekius bei, pasikeitus sistemos komponentams, dinamiškai keisti ir sistemai keliamus reikalavimus.

1.3.1. Trasavimo matricos

Trasavimo matricos yra įrankis, kurį galima naudoti reikalavimams verifikuoti. Matricos susieja reikalavimus tarp skirtingų artefaktų. Trasavimo matricose ryšiai tarp reikalavimų ir kitų artefaktų yra kuriami rankiniu būdu, todėl šis metodas turi mastelio ir ilgalaikio vystymo iššūkių [NH07].

Žingsniai trasavimo matricai paruošti:

1. surinkti visus turimus reikalavimus;
2. kiekvienam reikalavimui priskirti unikalų identifikacinį numerį;
3. susieti reikalavimą su kitu artefaktu.

Priklausomai nuo organizacijos, informacija pateikiama matricoje gali būti kitokia. Gali būti pateikiamas aprašymas, savininkas, testavimo scenarijai, jų rezultatai ir kita informacija. Kaskart šiai informacijai pasikeitus, trasavimo matricą būtina atnaujinti.

Trasavimo matrica		Programos reikalavimai				
		REIK001	REIK002	REIK003	REIK004	REIK005
Testavimo scenarijai	TS001	x				
	TS002		x			
	TS003		x			
	TS004			x		
	TS005			x		
	TS006			x		
	TS007				x	
	TS008				x	x
	TS009					x
	TS010					x

6 pav. Trasavimo matricos pavyzdys

Šio metodo privalumas yra jo paprastumas. Jei trasavimo matricos naudojamos nuo projekto pradžios, tam nereikalingas didelis pasiruošimas. Pasitelkiant šį metodą, nesunku pastebėti, kurie reikalavimai kuriuose artefaktuose pradingsta.

Vis dėlto, šis metodas turi ir trūkumų. Projektui plečiantis, gali būti sudėtinga turėti atnaujintą informaciją matricose. Taip pat, daugėjant sistemos artefaktų, gali pasidaryti sunku sekti reikalavimus, o matricų skaičius gali stipriai išaugti [CSD+05].

1.3.2. Trasavimo sąrašai

Dar paprastesnė trasavimo matricos versija yra trasavimo sąrašai. Trasavimo sąrašas kiekvienam reikalavimui priskiria susijusius artefaktus. Sąrašai yra kompaktiškesni negu matricos ir gali būti pildomi be galo. Naudojant šį metodą, gali būti keletas lentelių, pavyzdžiui, po vieną kiekvienam ryšiui. Trasavimo sąrašo trūkumas yra toks, kad sunku surasti atgalinį ryšį, nes kaskart reikia peržiūrėti visą sąrašą [Hok01].

Reikalavimas	Priklausomas nuo
R1	R3, R4
R2	R5, R6
R3	R4, R5
R4	R2
R5	R6

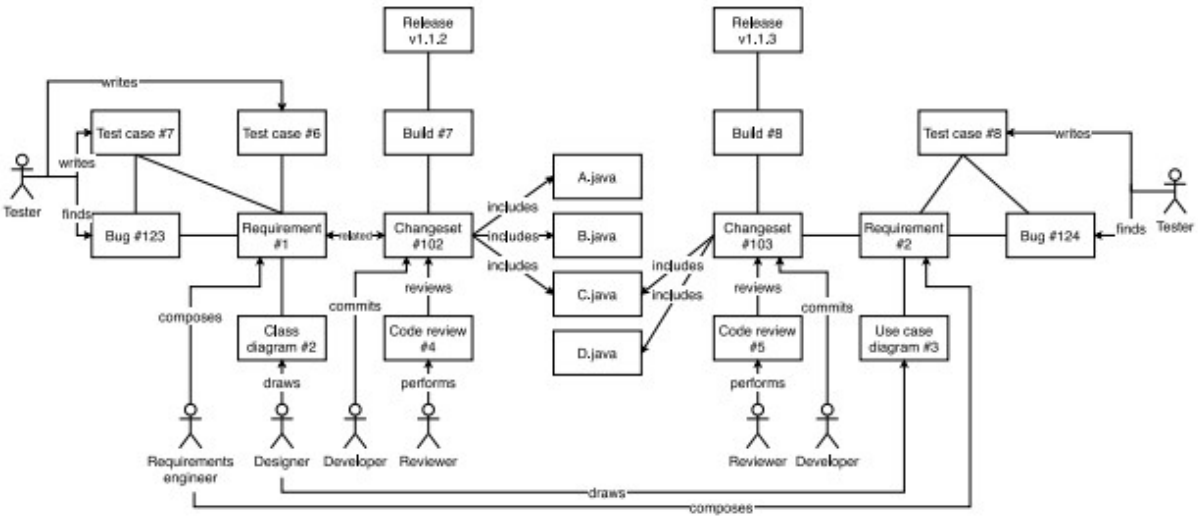
7 pav. Trasavimo sąrašo pavyzdys

1.3.3. Trasavimo medžiai

Grafai yra patogus būdas atvaizduoti ir analizuoti duomenis, kurie tarpusavyje turi ryšius [STD21]. Šiame darbe, grafai, kuriais yra vykdomas reikalavimų trasavimas, bus vadinami trasavimo medžiais.

Trasavimo medžiuose, artefaktus atspindi grafo viršūnės, o briaunos – trasavimo ryšius. Tam, kad būtų atskirti skirtingi trasavimo ryšiai, trasavimo medyje galima sukurti skirtingus viršūnių arba briaunų tipus arba jiems priskirti atitinkamus atributus. Produkto poreikis (arba pats produktas) šiuo atveju būtų medžio šaknis, o reikalavimai – iš šios šaknies išeinančios šakos.

Kaip matome 8 paveiksle , iš grafo galime matyti, kokius ryšius artefaktas turi su savo kaimynais. Šiame pavyzdyje, artefaktai ir komandos nariai yra vaizduojami, kaip medžio viršūnės, o ryšiai – briaunomis. Trasavimo medis padeda vizualizuoti ryšius tarp skirtingų artefaktų. Analitikas gali pamatyti artefaktą ir tai, kokią įtaką padarys pakeitimai jame, tikrindamas artefakto



8 pav. Trasavimo medžio pavyzdys [STD21]

kaimynus (kitus artefaktus, kurie susiję per briaunas) [STD21].

Taip pat, analizuodami trasavimo medį, galime pamatyti ir ne tokią trivialią informaciją, pavyzdžiui – artefaktų klasterius [STD21].

1.3.4. Kokybės funkcijų sklaida

Kokybės funkcijų sklaida yra vienas iš metodų, padedantis stebėti reikalavimus ir taip išvengti programinės įrangos perkūrimo.

Šio metodo tikslas yra tobulinant programos kūrimo procesą susikoncentruoti į tai, kas yra svarbiausia klientui. Kad tai būtų pasiekta, projekto komanda, specifikuodama reikalavimus, turi susirinkti suinteresuotųjų grupių reikalavimus ir jų prioritetus, o tada perduoti juos tolimesnėms programos kūrimo fazėms [Gov96].

Naudojant kokybės funkcijų sklaidą, reikalavimai yra skirstomi į tris grupes:

1. Laukiami reikalavimai – tai tokie reikalavimai, kurių gali neformuluoti nei viena suinteresuota pusė, bet visos jos mano, kad šie reikalavimai yra akivaizdūs, o jų neįgyvendinus bus susilaukta nepasitenkinimo bei nusistebėjimo vykdytojų neišmanymu;

2. Normalieji reikalavimai – tai tokie reikalavimai, kuriuos formuluoja bent viena iš suinteresuotųjų pusių, tačiau reikalavimas aktualus visiems, o jo neįgyvendinus bus susilaukta nepasitenkinimo;

3. Išskirtiniai reikalavimai – tokie reikalavimai, kurie suinteresuotoms šalims yra svarbūs, bet jų nerealizavus niekas nenustebs ir nebus sukeltas didelis nepasitenkinimas.

Nors remiantis šiuo metodu galima valdyti nefunkcinius reikalavimus, metodas nėra pritaikytas funkciniams. Taip yra todėl, kad funkciniai reikalavimai yra arba įgyvendinami, arba ne ir nėra tarpinės skalės, kurioje būtų galima vertinti reikalavimo įgyvendinimą [Gov96].

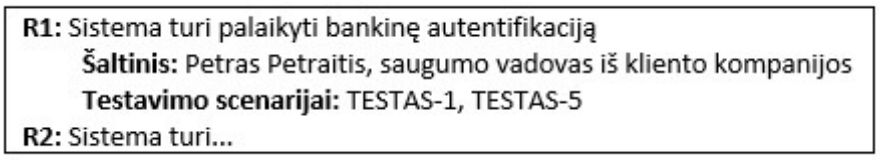
1.3.5. Trasavimas pagal identifikatorių

Trasavimo ryšiai gali būti dokumentuojami pagal identifikatorių. Identifikatorius šiuo atveju yra unikalus vardas arba numeris, kuris priskirtas kiekvienam artefaktui. Identifikatoriai čia būtini tam, kad būtų galima sukurti nuorodas į artefaktus [She14].

Šis metodas labiausiai tinka aprašyti vienas su vienu arba vienas su daug trasavimo ryšius. Pavyzdžiui, jei reikalavimas turi vieną testavimo scenarijų, tuomet turime vienas su vienu ryši, o jei daugiau – vienas su daug. Daug su daug ryšių aprašyti šiuo metodu gali būti sunku dėl didelės apimties [She14].

1.3.6. Trasavimas pagal atributus

Dar vienas būdas trasuoti reikalavimams yra pridėti atributus prie artefaktų aprašų. Šiuo metodu, priešingai nei prieš tai aprašytu, patogiau galima trasuoti daug ryšių. Vis dėlto, kadangi atributai sukuria ilgą dokumentą, nes jiems reikalingi žodiniai ir ilgi aprašymai, trasavimo matricos gali būti parankesnis metodas aprašyti daug su daug ryšius [She14]



9 pav. Trasavimo pagal atributus pavyzdys

Kaip matome paveiksle 18, kiekvienas iš reikalavimų turi savo identifikacinį numerį. Toliau yra apibrėžiamas reikalavimas, o po juo gali būti nurodomi reikalavimo atributai. Šiuo atveju, kaip atributą nurodome reikalavimo šaltinį bei su reikalavimu susietus testavimo scenarijus [She14]. Šiuo būdu trasuodami reikalavimus, galime pateikti daug detalesnės informacijos negu trasuodami matricomis arba sąrašais.

1.3.7. Trasavimo metodai kibernetinėms–fizinėms–socialinėms sistemoms

Darbo metu taip pat buvo atlikta paieška, kad būtų nustatyta, ar egzistuoja jau esantys trasavimo problemų sprendimo būdai konkrečiai kibernetinėms–fizinėms–socialinėms sistemoms. Paieška buvo vykdoma pasitelkiant šiuos resursus:

1. ScienceDirect
2. ResearchGate
3. ACM DL
4. IEEE

Paieška buvo vykdyta pagal raktažodžius: requirements traceability for cyber–physical–social systems, requirements traceability, requirements management for cyber–physical–social systems, requirements management, requirements engineering for cyber–physical–social systems, requirements engineering.

Atlikus paiešką pagal duotus raktažodžius pateiktose bibliotekose, viešai arba universitetui prieinamuose straipsniuose, nebuvo rasta analizės ar trasavimo problemos sprendimo būdų kibernetinėms–fizinėms–socialinėms sistemoms.

1.4. Skyriaus išvados

Rezultatai

Šiame skyriuje buvo apžvelgtos su reikalavimų trasavimu susijusios temos. Darbe apibrėžtos kibernetinės–socialinės–fizinės sistemos bei jų ypatumai.

Remiantis kibernetinių–fizinių–socialinių sistemų ypatumais, buvo vertinami reikalavimų trasavimo metodai.

Prieš metodų analizę buvo skiriamas dėmesys reikalavimų trasavimo apžvalgai, nustatytos priežastys, kodėl reikalavimų trasavimas yra svarbi sėkmingo projekto dalis

Trasavimo metodų apžvalga		
Metodas	Privalumai	Trūkumai
Trasavimo matricos	Paprastos naudoti, nereikalauja daug žinių	Didėjant reikalavimų skaičiui, tampa sudėtingai įgyvendinamos, reikia surinkti didelį informacijos kiekį prasidedant projektui, neleidžia dinaminio elementų įtraukimo arba išmetimo.
Trasavimo sąrašai	Gali būti pildomi be galo	Didėjant reikalavimų skaičiui, darosi sunku sekti reikalavimus, neleidžia dinaminio elementų įtraukimo arba išmetimo.
Trasavimo medžiai	Aiškus grafinis atvaizdavimas	Didejant reikalavimų skaičiui, tampa sudėtingai įgyvendinami, neleidžia dinaminio elementų įtraukimo arba išmetimo
Kokybės funkcijų sklaida	Padeda prioritetizuoti bei lokalizuoti reikalavimus	Nepritaikyta tiesiogiai reikalavimams trasuoti.
Trasavimas pagal identifikatorių	Aiškūs aprašai ir paprastas naudojimas	Metodas labai jautrus artefaktų skaičiaus augimui, neleidžia dinaminio elementų įtraukimo arba išmetimo
Trasavimas pagal atributus	Aiškūs aprašai ir paprastas naudojimas, galima pateikti daug informacijos apie artefaktą	Didejant reikalavimų skaičiui, tampa sudėtingai įgyvendinami, neleidžia dinaminio elementų įtraukimo arba išmetimo

1 lentelė. Trasavimo metodų palyginimas

Kadangi nėra metodo, kuris puikiai tiktų pagal apibrėžtus kriterijus, toliau šiame darbe bus kuriamas metodas, kuris pagerintų reikalavimų trasavimą kibernetinėse–fizinėse–socialinėse sistemose.

Išvados

Apibendrinus darbe pateiktą informaciją, buvo padarytos šios išvados:

1. Galime išskirti tris ypatumus, pagal kuriuos šios sistemos išsiskiria iš kitų. Dėl šių ypatumų, reikalavimų trasavimas susiduria su naujais iššūkiais:
   (a) Kibernetinės–fizinės–socialinės sistemos yra didelės apimties ir turi didelius kiekius reikalavimų.
   (b) Kibernetinėse–fizinėse–socialinėse sistemose komponentų skaičius ir sudėtis gali dinamiškai kisti.
   (c) Kibernetinėse–fizinėse–socialinėse sistemose reikalavimų įgyvendinimas siejasi per kelis realizacinius lygmenis (autonominis sistemos elementų veikimas ir visos sistemos bendras veikimas).
2. Trasavimas turi būti vykdomas dėl šių priežasčių:
   (a) Sumažinami nuostoliai, kuomet žmonės palieka komandą.
   (b) Palengvina versijavimą ir konfigūracijos valdymą.
   (c) Padeda kontroliuoti programos kūrimo procesus.
   (d) Palengvina programinės įrangos palaikymo procesus.
3. Literatūros apžvalgoje, nerasta būdų, kaip išspręsti sunkumus, kurie kyla trasuojant reikalavimus kibernetinėms–fizinėms–socialinėms sistemoms.
4. Analizė parodė, kad esami trasavimo metodai, vykdant trasavimą kibernetinėse–fizinėse–sistemose, nėra tinkami.

2. Kokybės funkcijų sklaidos taikymas trasavimui įgyvendinti

Šioje darbo dalyje bus kuriamas trasavimo modelis, kuriuo būtų pasiekiamas tikslas lengviau trasuoti Kibernetinių–fizinių–socialinių sistemų reikalavimus. Kad sukurtume tokį modelį, reikia išsiaiškinti, kokie artefaktai sudaro kibernetines–fizines–socialines sistemas.

2.1. Trasuojami objektai

Artefaktas yra dokumentuotas žingsnis programos kūrimo procese. artefaktai gali turėti daug skirtingų formų. Tai gali būt duomenų modelis, reikalavimo aprašymas, programos modulis, testavimo atvejis, kokybės reikalavimas ar kt. Dokumentuodami artefaktus, surenkame programos kūrimo veiklas iš skirtingų požiūrio taškų. Įprastai, artefaktai yra susiję – vėlesniems artefaktams įtaką daro ankstesnėse programos kūrimo fazėse sukurti artefaktai.

2.1.1. Sistemos kūrimo fazės

Išskirkime penkis artefaktų lygmenis pagal tai, kokiose programos kūrimo fazėse jie atsiranda.

Analizė. Pirmoji sistemos kūrimo fazė, kurioje atsiranda pirmieji artefaktai yra analizė. Šioje fazėje iškeliamas verslo poreikis sistemos sukūrimui. Nusprendus, kokią sistemą reikia sukurti, reikalavimai produktui yra apibrėžiami ir dokumentuojami. Šioje fazėje dažniausiai dalyvauja veiklos ekspertai ir galutiniai būsimos sistemos naudotojai, o esminis šios fazės tikslas yra išsiaiškinti reikalavimus būsimai sistemai. Kuomet reikalavimai suformuluojami, jie yra formaliai suspecifikuojami į dokumentus, vadinamus reikalavimų specifikacijomis. Įprastai šie dokumentai aprašo funkcinius ir nefunkcinius sistemos reikalavimus, interfeisų reikalavimus, dizaino reikalavimus bei programavimo standartus tam, kad panaikintų galinčias kilti dviprasmybes ar apibrėžtų neapgalvotas vietas.

Projektavimas. Šioje fazėje apibrėžiami projektinio lygmens reikalavimai programos sričiai, architektūrai, sistemos komponentams, interfeisams bei duomenims. Kadangi visi projektiniai sprendimai šiame lygmenyje turi tenkinti reikalavimus, iškeltus analizės metu, dažnai yra naudojamos formalios priemonės tam pagrįsti, pavyzdžiui, UML diagramos. Kuo daugiau formalių projektinių sprendimų yra priimama, tuo mažesnė klaidos tikimybė. Taip pat formalizuodami reikalavimus projektiniame lygmenyje sukuriame galimybės atgaliniam atsekamumui.

Visi reikalavimai šioje fazėje turi tapti projektiniais sprendimais, todėl kiekvienas reikalavimas turi turėtį ryšį bent su vienu projektiniu sprendimu: komponento aprašymu, modeliu, maketu ar kt.

Realizacija. Šios fazės metu, remiantis anksčiau sukurta dokumentacija bei modeliais, sukuriamas produktas. Šiuo metu sistema taip pat yra dokumentuojama, bei joje ieškoma klaidų. Šios fazės metu sukuriamas programinis kodas, o kibernetinių–fizinių–socialinių sistemų atveju ir ištisos kiberfizinės sistemos, susidedančios ne tik iš programinio kodo, tačiau ir iš fizinių sistemos elementų.

Palaikymas. Tai yra procesas, kurio metu jau esama sistema yra modifikuojama tam, kad sistema veiktų geriau arba šalinamos sistemos klaidos, arba keičiama sistemos elgsena dėl pasikeitusių išorės veiksnių (pavyzdžiui, pasikeitusių įstatymų). Šioje fazėje ypač svarbu, jog dokumentacija, paruošta analizės ir projektavimo fazėse būtų kaip įmanoma teisingesnė, o ryšiai tarp dokumentacijos ir realizacijos metu sukurto produkto būtų kuo aiškesni, nes tai sutaupo laiką modifikuojant sistemą.

Šios fazės metu, žinoma, taip pat yra kuriama dokumentacija, kurioje aprašomi visi įvykdyti pakeitimai bei vykdomas testavimas tam, kad būtų įsitikinta, jog pakeitimai veikia taip, kaip ir turėtų.

Uždarymas. Periodas, kurio metu sustabdomas sistemos palaikymas, šiame darbe vadinamas uždarymu. Sistemos uždaromos tuomet, kai sistemai tam, kad atitiktų esamus poreikius, reikalingi drastiški pakeitimai. Šios fazės metu taip pat teisinga dokumentuoti priežastis, kodėl produktas buvo uždarytas.

Susisteminkime artefaktus pagal fazes, kuriose jie yra keičiami.

1. Analizė
    (a) Reikalavimų aprašymai
    (b) Modeliai
    (c) Maketai
2. Projektavimas
    (a) Reikalavimų aprašymai
    (b) Komponentų aprašymai

(c) Modeliai

(d) Maketai

3. Realizacija

(a) Reikalavimų aprašymai

(b) Komponentų aprašymai

(c) Modeliai

(d) Maketai

(e) Programinis kodas

(f) Testavimo scenarijai

(g) Duomenų laikmenos

(h) Testavimo rezultatai

(i) Techninė aparatūra

(j) Klaidos

4. Palaikymas

(a) Reikalavimų aprašymai

(b) Modeliai

(c) Maketai

(d) Programinis kodas

(e) Testai

(f) Duomenų laikmenos

(g) Testavimo rezultatai

(h) Komponentai

(i) Techninė aparatūra

(j) Klaidos

5. Demontavimas

(a) Dokumentacija

2.2. Trasavimo modelis

2.3. Reikalavimai trasavimo modeliui

Mūsų nagrinėjamos sistemos nėra homogeniškos – susideda iš daug skirtingų komponentų, kurie gali būti tiek programinis kodas, tiek fizinė įranga. Trasavimą šiame modelyje laikysime bet kokį ryšį tarp bet kurių sistemos artefaktų, kurie bus sutinkami sistemos kūrimo cikle.

Turime apibrėžti modelį taip, kad jo pagalba galėtume gauti trasavimo informaciją apie kibernetines–fizines–socialines sistemas ir suteikti galimybę modelio naudotojams papildyti trasavimo informaciją reikalingais atributais. Tokiu būdu šį modelį bus galima pritaikyti visoms kibernetinėms–fizinėms–socialinėms sistemoms.

Trasavimo modeliais įprastai naudojasi trijų rūšių vartotojai:

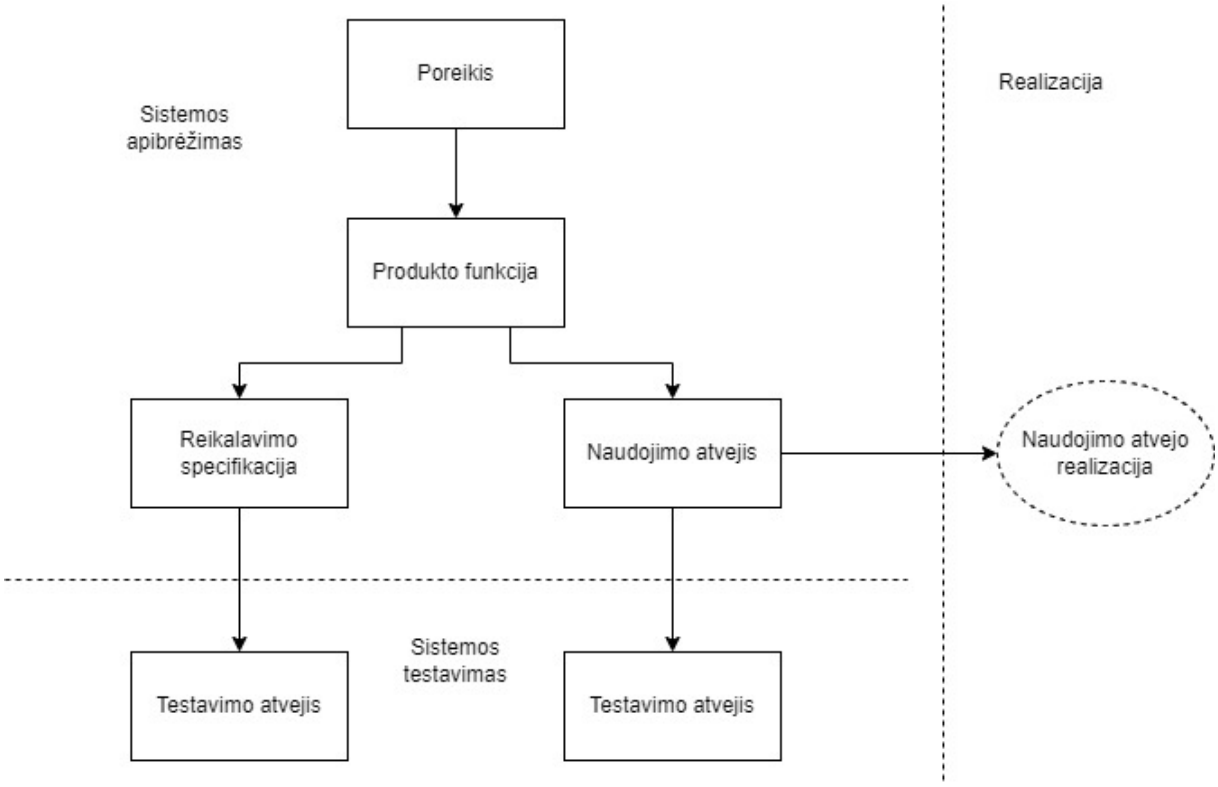
1. Administratoriai: supranta trasavimo modelį, įvairius apribojimus ir charakterizacijas.
2. Inžinieriai: valdo įrankius, kuriais naudojantis gali būti sukurti ir stebimi trasavimo ryšiai, vykdo trasavimą pagal administratorių nurodytas taisykles.
3. Dalykinės srities ekspertai: naudodamasis įrankiais kontroliuoja trasavimo informaciją tarp skirtingų sistemos artefaktų ir supranta šių ryšių praktinę prasmę.

Reikalavimai modeliui:

1. Modelis turi leisti nustatyti ryšius tarp skirtingų sistemos artefaktų, kurie gali būti tiek homogeniški, tiek heterogeniški.
2. Modelis turi turėti vienas su vienu, vienas su daug ir daug su daug ryšių tarp artefaktų.
3. Modelis turi turėti ryšių kryptis (nurodyti, iš kurio artefakto kuris kitas artefaktas kyla).
4. Modelis turi apriboti galimybę sudaryti netinkamus ryšius tarp artefaktų.
5. Modelis turi būti lankstus, kad būtų galima pridėti naujus artefaktus arba ryšius nekeičiant pačio modelio.

2.3.1. Trasavimo objektai ir ryšiai

Skirtingi projektai turi skirtingus sistemų artefaktus ir skirtingus jų organizavimo būdus. Vis dėlto, egzistuoja generalizuotas artefaktų modelis trasavimo strategijai, kuris yra bendras daugumai



10 pav. Generalizuotas trasavimo modelis [Lef96]

projektų. Modelis seka nuo abstrakčių reikalavimų iki labiau detalizuotų, tuomet persikelia į realizacinį bei testavimo lygmenis. Nepriklausomai nuo to, koks konkretus projektas bus vykdomas, generalizuotas modelis turėtų atrodyti taip, kaip nurodyta 10 pav.

Šis modelis rodo, kad sekame reikalavimus iš sistemos apibrėžimo (reikalavimų) srities į įgyvendinimo ir testavimo sritis. Žinoma, kuriant realią sistemą, yra daug papildomų elementų, kuriuos reikia trasuoti, tačiau šiame generalizuotame modelyje nurodyti pagrindiniai artefaktai dažnai apima daugumą poreikių. Papildykime šį modelį artefaktais, kurie atsiranda kuriant kibernetinę–fizinę–socialinę sistemą [Lef96].

Sugrupuokime anksčiau pateiktus kibernetinių–fizinių–socialinių sistemų artefaktus pagal tris generalizuoto modelio sritis:

1. Sistemos apibrėžimas
   (a) Reikalavimų aprašymai
   (b) Modeliai
   (c) Maketai
   (d) Komponentų aprašymai
2. Sistemos testavimas

(a) Testavimo scenarijai

(b) Testavimo rezultatai

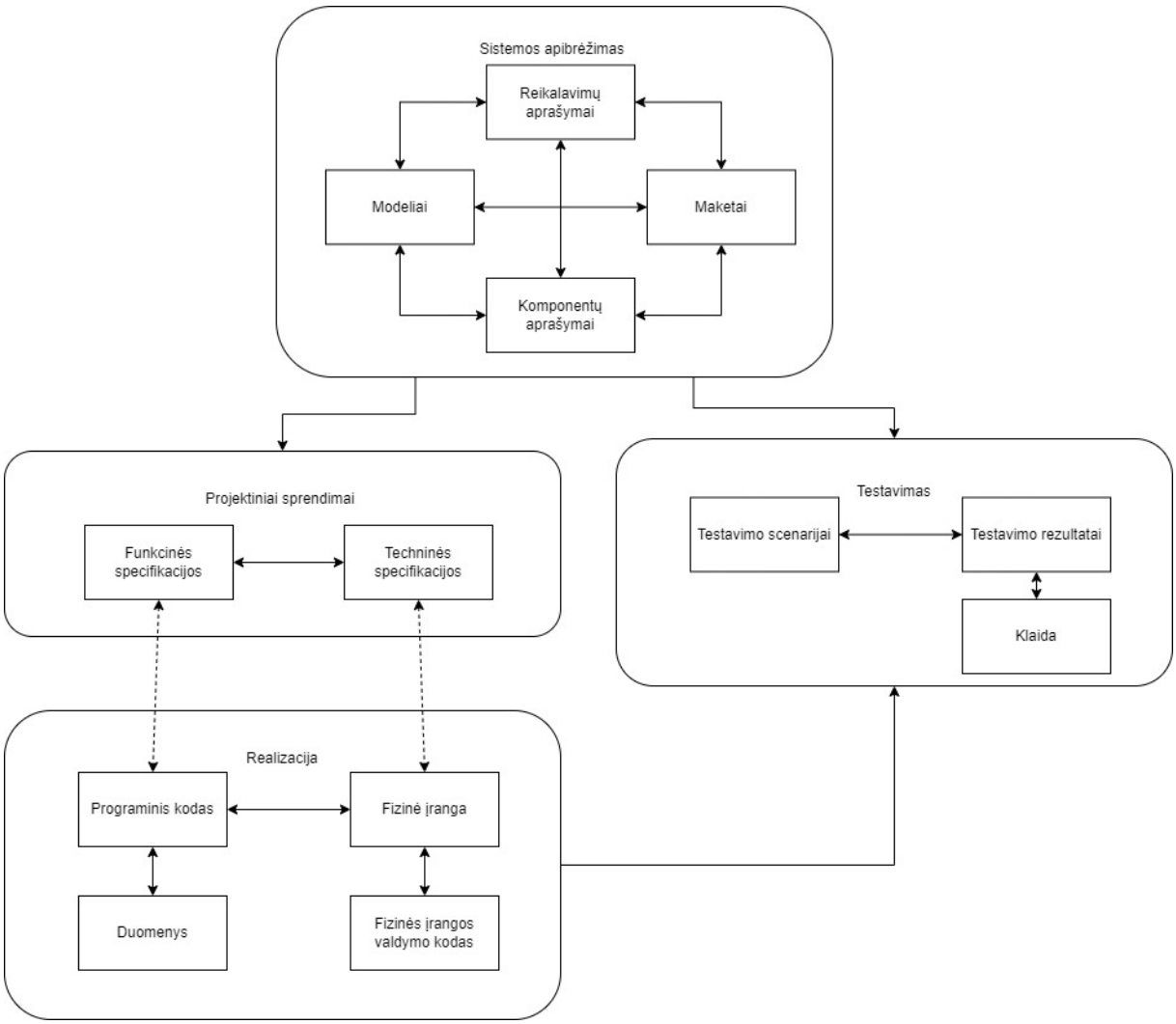
(c) Klaidos

3. Realizacija

(a) Programinis kodas

(b) Fizinė įranga

(c) Duomenys



11 pav. Kiberfizinių–socialinių sistemų artefaktų ryšiai

2.4. Kokybės funkcijų sklaida

Kokybės funkcijų sklaida (KFS) yra į klientą orientuotas būdas kurti produktus. Šiuo metodu, produkto kūrimo komandos gali struktūruotai kurti produktus, kurie atlieptų rinkos poreikius per

daug skirtingų lygmenų [Gov96].

KFS metodika nusakoma šešiais baziniais žingsniais:

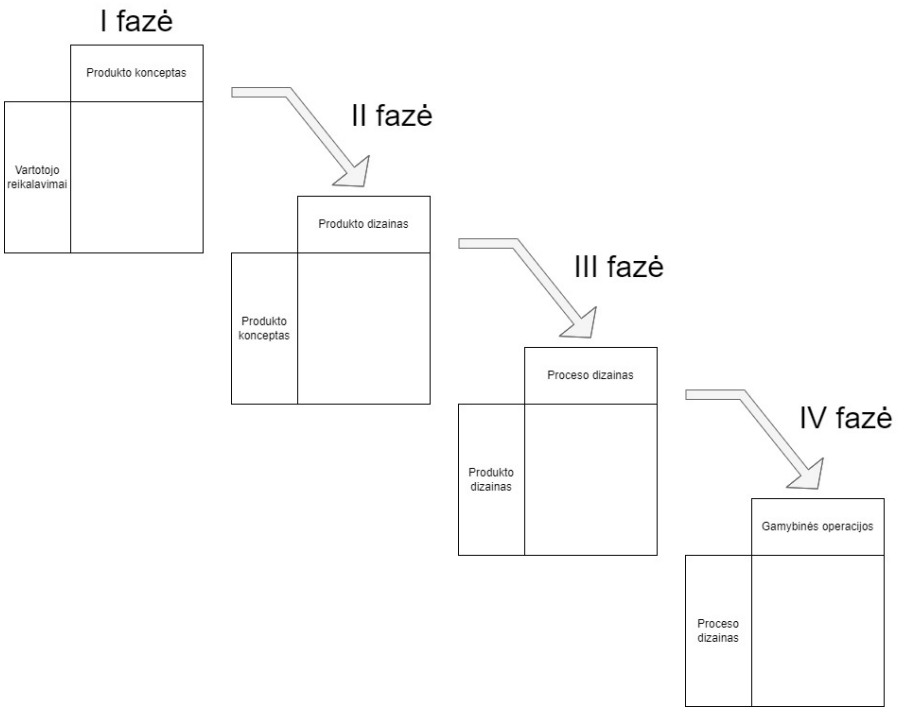
1. Užsakovo reikalavimų formulavimas užsakovo terminais.
2. Sistemos reikalavimų ir jos charakteristikų ryšių nustatymas.
3. Sistemos reikalavimų transformavimas į lokalizuotas funkcines specifikacijas.
4. Specifikacijų transformavimas į sistemos reikalavimus.
5. Reikalavimų vertinimo metodų parinkimas.
6. Grįžtamasis ryšys užsakovui ir kitiems projekto dalyviams.

Esminis KFS instrumentas yra kokybės namas (angl. House of quality), kurio pagalba atliekama duomenų analizė įkomponuojant turimas žinias apie klientus, konkurentus, procesus, sistemas, technologijas ir t. t. Kokybės namas – tai konkurencingo produkto kūrimo priemonė, kuri padeda užtikrinti vartotojo poreikių įgyvendinimą kuriant produktą. Jis suformuojamas per eilę žingsnių sudarant matricas. Tai lankstus instrumentas, kuris gali būti pritaikomas pagal poreikį, patobulinamas ir išplečiamas [R02].

KFS panaudojimo privalumai:

1. Geresnė kokybė
2. Tobulesnis kompanijos veikimas
3. Mažesnės projektavimo ir gamybos kainos
4. Padidintas produkto patikimumas
5. Sumažintas sprendimo/planavimo laikas
6. Padidintas techninio ir kito personalo produktyvumas
7. Sumažėję garantiniai reikalavimai
8. Patobulintos marketingo galimybės
9. Patobulintas sprendimų priėmimas
10. Labiau į vartotoją orientuota darbo jėga

Kokybės funkcijų sklaidą įprastai sudaro keturios fazės:



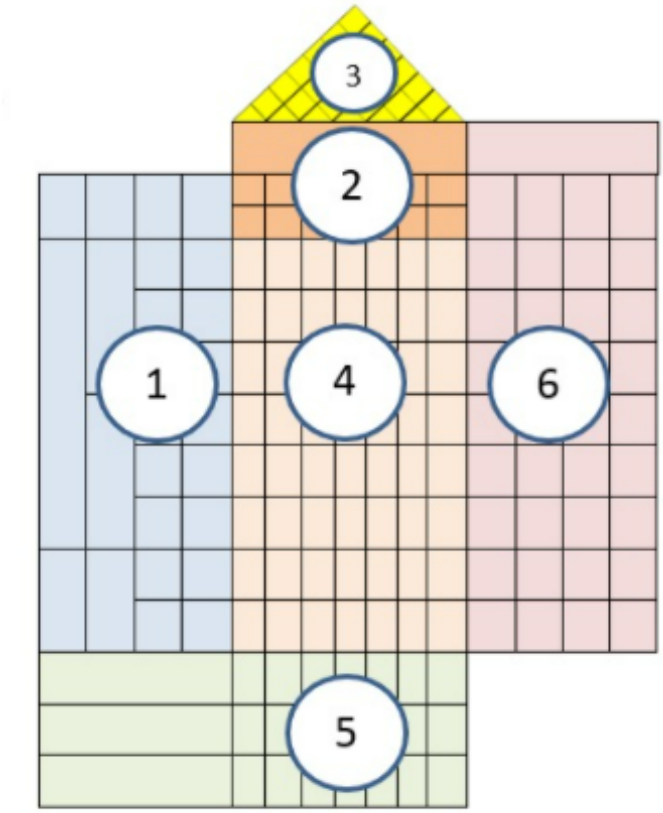
12 pav. Kokybės namo fazės

1. Produkto kokybės planas
2. Komponentų kokybės planas
3. Technologijų kokybės planas
4. Gamybos kokybės planas

Kiekvienam etapui yra kuriamas naujas kokybės namas iš pirmojo vieną dalį perkeliant į antrąjį, iš antrojo į trečiąjį, o iš trečiojo į ketvirtąjį. Iš seno kokybės namo perkeliama dalis „Kaip?“ ir įkeliama vietoje „Ką?“ (12 pav.)

Kokybės namą sudaro 6 dalys, kurios yra pažymėtos 14 paveiksle. Jų reikšmės įvardintos žemiau:

1. Verslo ir vartotojų reikalavimai kartu su santykinės svarbos reitingu (kairioji namo pusė).
2. Sprendimo reikalavimai kartu su santykinės svarbos reitingu (viršutinė namo dalis).
3. Koreliacijos ir priklausomybės tarp susijusių reikalavimų (namo stogas).
4. Trasavimo ryšys tarp skirtingų lygmenų sprendimų (centrinė namo dalis).
5. Tikslai bei apskaičiuotos kiekybinės metrikos bei prioritetai (namo apačia).
6. Konkurencijos matrica (dešinioji namo pusė).



13 pav. Kokybės namo sandara

Kokybės funkcijų sklaida yra plačiai taikoma Japonijoje, Amerikoje ar Europoje. Užsienio literatūroje yra daugybė straipsnių, kuriuose pateikiami pavyzdžiais ir mokslininkų tyrinėjimais kaip kokybės funkcijų sklaidos metodą galima sujungti su kitais kokybės vadybos modeliais ir pritaikyti organizacijos kūrimo produktui, paslaugai, procesams ar veiklai pagerinti [Kra14]. Šiame darbe kokybės funkcijų sklaida bus panaudota pademonstruoti, jog tai gali būti ir įrankis, kurio pagalba galima trasuoti reikalavimus kibernetinės–fizinėse–socialinėse sistemose.

Kokybės funkcijų sklaidos taikymas reikalavimų trasavimui

Pritaikykime kokybės funkcijų sklaidos kokybės namus reikalavimų trasavimui kibernetinėse–fizinėse–socialinėse sistemose.

Tam, kad būtų galima adaptuoti kokybės funkcijų sklaidos metodą reikalavimų trasavimui, turime apsibrėžti šias sąvokas:

1. Reikalavimas: poreikis sistemos funkcionalumui, kuris arba buvo išreikštas kliento, arba paimtas iš aukštesnio lygio realizacijos.

2. Svarba: matricoje gauta svarbos reikšmė įvertinus realizacinius sprendimus.

3. Kokybės faktoriai: sąrašas faktorių, kurie apibrėžia, ar sistemai keliamas reikalavimas yra patenkinamas. Kokybės faktoriai apibrėžia tokius sistemos atributus kaip patikimumas, saugumas ir t. t.

4. Koreliacija: realizacinių sprendimų įtakos lygis įgyvendinant aukštesnio lygio reikalavimus. Kokybės namuose įprastai naudojama trijų lygių koreliacijos skalė.

5. Kokybės namo stogas: viršutinė (trikampė) kokybės namo dalis, kurioje nurodoma, ar keliami sistemos reikalavimai neprieštarauja vienas kitam.

Sistemos projektavimo ir kūrimo fazėse yra pravartu žinoti, kokie projektiniai sprendimai yra svarbiausi t. y. labiausiai koreliuoja su sistemai iškeltais reikalavimais. Kokybės funkcijų sklaida yra tinkamas metodas šiai problemai spręsti, nes šis metodas buvo sukurtas tam, jog transformuotų kliento poreikius į projeketinius sprendimus. Naudojantis kokybės funkcijų sklaidos metodo įrankiu vadinamu kokybės namu, galima sekti reikalavimo kelią nuo pat kliento poreikio iki realizacinių lygmenų. Ir klientui, ir sistemos kūrimo komandai yra pravartu turėti vizualizuotą matricą, kurioje matytųsi, koks projektinis sprendimas atliepia kokį reikalavimą. Naudojant šį metodą, galima matyti, kokiam sprendimui įgyvendinti reikalinga skirti daugiau resursų. Taip galima pasiekti geresnę sistemos kokybę ir didesnį klientų pasitenkinimą.

Kokybės nameliuose kiekviename lygmenyje nurodomos ir koreliacijos tarp reikalavimų. Tai yra nurodoma namo stoge. Kokybės namo stogas yra trikampė matrica, aprašanti skirtingų sistemos kokybės charakteristikų t. y. sistemos techninių reikalavimų tarpusavio sąveiką. Koreliacijos gali būti stipriai teigiamos, teigiamos, neigiamos ir stipriai neigiamos. Neigiamos koreliacijos nurodo, kad reikalavimai tarpusavyje konfliktuoja. Tai reiškia, kad norint įgyvendinti sistemą, reikia keisti reikalavimus, kad konfliktų nebeliktų.

Dešinėje namelio pusėje nurodoma varžovų tiekiamų sistemų vertinimo matrica. Vis dėlto, reikalavimų trasavime ši kokybės namo dalis nėra reikalinga, todėl jos toliau nenagrinėsime ir eliminuosime iš kokybės namo.

Naudojant kokybės namą gali paaiškėti, kad nors kuris nors reikalavimas ir yra labai svarbus užsakovui, bet dėl nepakankamos vykdytojų kvalifikacijos ar dėl kokių nors kitų priežasčių tą reikalavimą įgyvendinti yra labai sunku ir, apsiėmus tai padaryti, gali būti viršytas projekto biudžetas, nespėjama sukurti sistemos numatytu laiku arba galbūt projektas gali ir apskritai sužlugti.

Tuščia ryšių matricos eilutė parodo, kad atitinkamas aukštesnio lygmens reikalavimas žemesniame lygmenyje neturi jokių atitikmenų arba, kitaip tariant, nėra lokalizuotas. Tai reiškia, jog reikalavimas iš esmės nėra įgyvendinamas. Analogiškai, jei egzistuoja tuščias matricos stulpelis,

tai reiškia, jog realizacinis sprendimas neatitinka jokio aukštesnio lygio reikalavimo, todėl tokia realizacija yra perteklinė.

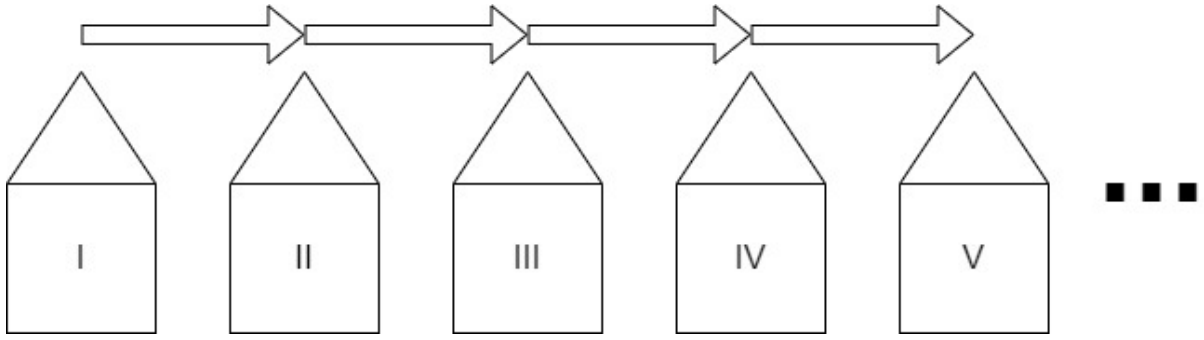
Apatinėje matricos dalyje taip pat pateikiama svarbi informacija apie reikalavimus ir jų įgyvendinimą. Šioje dalyje nurodoma papildoma informacija, pavyzdžiui tai, koks svarbus yra reikalavimo įgyvendinimas, taip pat kiek glaudžiai jis susijęs su kitais reikalavimais bei kaip sudėtinga būtų atlikti pakeitimus.

Kibernetinėms–fizinėms–socialinėms sistemoms netinka tradicinis kokybės funkcijų sklaidos procesas su 4 nameliais, kadangi jau ankstesniame skyrelyje išsiaiškinome, kad šios sistemos turi daugiau artefaktų sluoksnių. Vis dėlto, kokybės nameliai nereikalauja griežtos artefaktų hierarchijos. Kokybės namelių galima kurti tiek, kiek yra reikalinga konkrečioje situacijoje, perkeliant namelio viršaus dalį į kairę pusę, o viršuje surašant žemesnio sluoksnio artefaktus.

Pavyzdžiui, nameliai galėtų apibrėžti šiuos ryšius:

1. Kliento poreikis ir produkto funkcija
2. Produkto funkcija ir reikalavimo specifikacija
3. Reikalavimo specifikacija ir testavimo atvejis
4. Testavimo atvejis ir fizinis komponentas
5. Fizinis komponentas ir programinis kodas

Esant poreikiui, galima tęsti namelių struktūrą ir pridėti daugiau artefaktų arba keisti esamus.



14 pav. Ryšiai tarp namelių

2.5. KFS pranašumai prieš kitus trasavimo metodus

Literatūros apžvalgoje buvo pastebėta, kad esami trasavimo metodai nesprendžia kai kurių trasavimo klausimų, kurie kyla kuriant kibernetines–fizines–socialines sistemas. Šie klausimai apima tai, jog šiose sistemose reikalavimų aibė dinamiškai kinta, yra dideli reikalavimų kiekiai, o

taip pat realizaciniai sprendimai siejasi per autonominį vieno elemento bei visos sistemos veikimą. Kokybės funkcijų sklaidos įrankis kokybės namas visų šių problemų neišsprendžia, tačiau turi pranašumų prieš tradicinius metodus.

Kokybės namo pranašumas prieš kitus metodus išryškėja būtent tuomet, kai visos sistemos realizacija priklauso nuo autonominio kiekvieno sistemos elemento veikimo ir bendro visos sistemos veikimo. Palyginkime kaip reikalavimai transformuojasi į realizaciją naudojant įvairius reikalavimų trasavimo metodus.

Pasitelkime paprastą vieno reikalavimo pavyzdį sistemos, kuri turi užtikrinti vienodą temperatūrą visame name. Šio reikalavimo realizacijai reikės turėti centrinę valdymo sistemą, kurioje būtų galima nustatyti temperatūrą. Taip pat bus reikalingi autonomiškai veikiantys šildymo elementai, kurių kiekvienas turės palaikyti nustatytą temperatūrą. Pavaizduokime šį reikalavimą reikalavimų matricoje:

	Nustatyti temperatūrą sistemoje	Nustatyti temperatūrą autonominiame vienete
Palaikyti temperatūrą name	x	x

15 pav. Tradicinė reikalavimų matrica

Kaip matome, taip pavaizduoti reikalavimą reikalavimų matricoje yra galima, tačiau susiduriame su problema. Nepaisant to, jog galime įžvelgti, kad abu stulpeliai yra susiję pagal reikalavimą, nėra aišku, jog tai yra skirtinguose komponentuose įgyvendinamos dalys. Nustatyti temperatūrą sistemoje yra visos sistemos reikalavimas, o nustatyti temperatūrą autonominiame vienete ir yra tik to autonominio vieneto ribose. Taigi nepaisant to, jog abu stulpeliai rodo realizaciją, tai gali klaidinti, kadangi realizacija yra skirtinguose komponentuose.

Alternatyvus būdas pavaizduoti reikalavimo realizaciją būtų piešti dvi skirtingas matricas skirtingiems komponentams. Vienoje matricoje pavaizduoti reikalavimo santykį su realizacija visos sistemos lygmenyje, o kitoje matricoje – reikalavimo santykį su autonominiu sistemos elementu (16 pav.)

Sistemos lygmuo

	Nustatyti temperatūrą sistemoje
Palaikyti temperatūrą name	x

Autonominis lygmuo

	Nustatyti temperatūrą elemente
Palaikyti temperatūrą name	x

16 pav. Dviejų matricų kompozicija

Vis dėlto, taip vaizduodami reikalavimo realizavimą, nematome ryšio tarp sisteminio ir autonominio realizavimo, nes matricos nėra tarpusavyje susietos. Taip pat toks atvaizdavimas reikalauja didesnio matricų kiekio.

Reikalavimas	Realizacija
R1	P1, P2

2 lentelė. Trasavimo pavyzdys naudojant trasavimo sąrašą

Trasavimo sąrašai, trasavimas pagal identifikatorių bei trasavimas pagal atributus šiuo atveju būtų realizuojamas panašiai. Panagrinėkime trasavimo sąrašo pavyzdį.

Šiame sąraše reikalavimas atvaizduojamas sąryšiu su realizacija. Dėl paprastumo, reikalavimą įvardinkime kaip R1, o realizacijas kaip P1 ir P2.

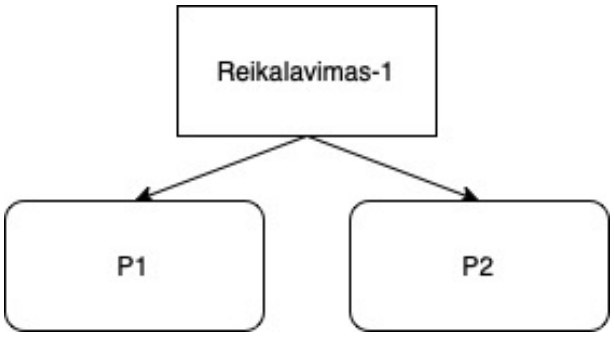
Trasuodami pagal atributą turėtume formuoti reikalavimų aprašymus ir pridėti sąrašus prie reikalavimo aprašymo. Savo esme, šis metodas nesiskiria nuo trasavimo sąrašų, tačiau yra kitaip atvaizduojamas. Trasuodami per atributus, nekuriame lentelės, o tik nurodome, kokie sprendimai įgyvendina šį reikalavimą. Praktikoje toks trasavimas galėtų atrodyti taip:

Reikalavimas-1: Palaikyti temperatūrą name
<kita informacija apie reikalavimą>
Realizacijos: P1, P2

17 pav. Trasavimas pagal atributus, kur P1 ir P2 anksčiau aprašytos realizacijos

Taip pat visi šie metodai neturi atgalinio atsekamumo. Tai reiškia, kad susiejęs reikalavimą su realizacija, negalime tiesiogiai atsekti reikalavimo pagal realizaciją.

Kaip paskutinę alternatyvą patikrinkime trasavimo medį. Trasavimo medis gerai vizualiai atvaizduoja ryšį tarp reikalavimo ir realizacijos. Nepaisant to, jog šiame medyje nėra pavaizduota priklausomybė tarp realizacijų, ta būtų galima padaryti įvedant naujus ryšius. Vis dėlto, šio metodo trūkumas yra laikas, kurio reikia medžiui nupiešti. Kuriant didelę sistemą, kurioje egzistuoja didelis ryšių skaičius, įdėti ar ištrinti elementą gali būti labai sudėtinga. Taip pat, trasavimo medis išreikštinai nenurodo reikalavimų prioritetų, kas kuriant informacines sistemas taip pat yra pravartu.



18 pav. Trasavimo medžio pavyzdys

2.6. KFS trūkumai ir ateities tyrimai

Nepaisant to, jog kokybės namas turi pranašumų prieš kitus trasavimo metodus, jis nėra tobulas. Literatūros analizės metu buvo nustatyta, jog kibernetinės–fizinės–socialinės sistemos taip pat išsiskiria tuo, jog sistema gali dinamiškai kisti, joje gali atsirasti nauji sudedamieji elementai, o tai reiškia, kad ir reikalavimai tokiai sistemai dinamiškai kinta. Šios problemos nesprendžia ir kokybės namas. Tobulinti kokybės namą, jungti jį su kitais metodais arba tyrinėti visiškai naują požiūrį į reikalavimų inžineriją gali būti tolesni šios krypties darbai.

Šiame darbe taip pat iš esmės buvo nagrinėtas tik sąryšis tarp reikalavimo ir realizacijos lygmenų, darant prielaidą, jog rekursiškai tai galima pritaikyti bet kuriems lygmenims, tačiau tai reikia įrodyti.

2.7. Apibendrinimas

Šiame skyriuje buvo kuriamas trasavimo modelis ir pritaikomas trasavimo metodas reikalavimams kibernetinėse–fizinėse–socialinėse sistemose trasuoti. Sudaryta trasuojamų objektų aibė, kuri, buvo nustatyta, gali būti nebaigtinė ir plečiama. Pagal tai sukurtas trasavimo modelis, numatyti ryšiai tarp trasuojamų objektų. Kad trasavimas būtų įmanomas, parinktas trasavimo metodas. Nuspręsta taikyti modifikuotą kokybės funkcijų sklaidos kokybės namo įrankį.

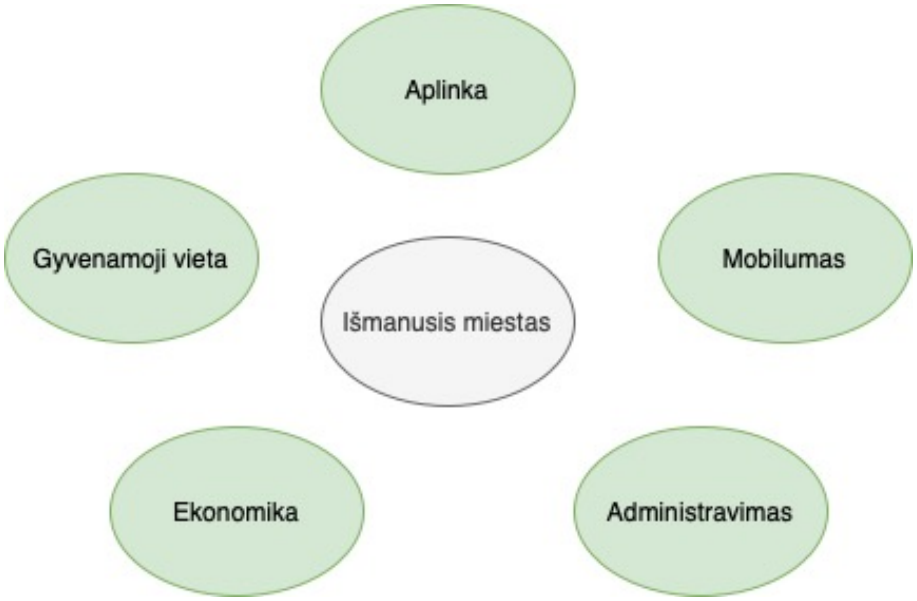
Buvo nustatyta, kad ne visos kokybės namo dalys reikalingos trasavimo uždaviniui įgyvendinti, todėl kokybės namas buvo modifikuotas atsisakant tam tikrų jo dalių (konkurencijos matricos, taip pat modifikuota metrikų ir prioritetų dalis).

Lyginant su kitais trasavimo metodais pastebėta, jog kokybės namas leidžia lengviau pastebėti ryšius tarp autonominių ir kibernetinės–fizinės–socialinės sistemos realizacijų. Naudodami kokybės namą taip pat galime numatyti, ar reikalavimai neprieštarauja vienas kitam, ar jų realizacijos yra įmanomos, o taip pat, kokia realizavimo svarba bei prioritetas. Kokybės namas taip pat veikia abiem kryptim, todėl nesunku ne tik iš reikalavimo atsekti realizacijas, bet ir atvirkščiai.

3. KFS panaudojimas kuriant išmanaus miesto sistemą

Praktiniam pritaikymui pasitelkime išmaniojo miesto pavyzdį. Išmaniojo miesto konceptas apjungia daugybę skirtingų sistemų, kurios tarpusavyje sąveikaudamos ir sudaro išmaniojo miesto visumą. Išmanusis miestas Aukščiausiame abstrakcijos lygmenyje turėtų turėti tokius modulius:

1. Išmanioji aplinka
2. Išmanusis mobilumas
3. Išmanusis administravimas
4. Išmanioji ekonomika
5. Išmanioji gyvenamoji vieta



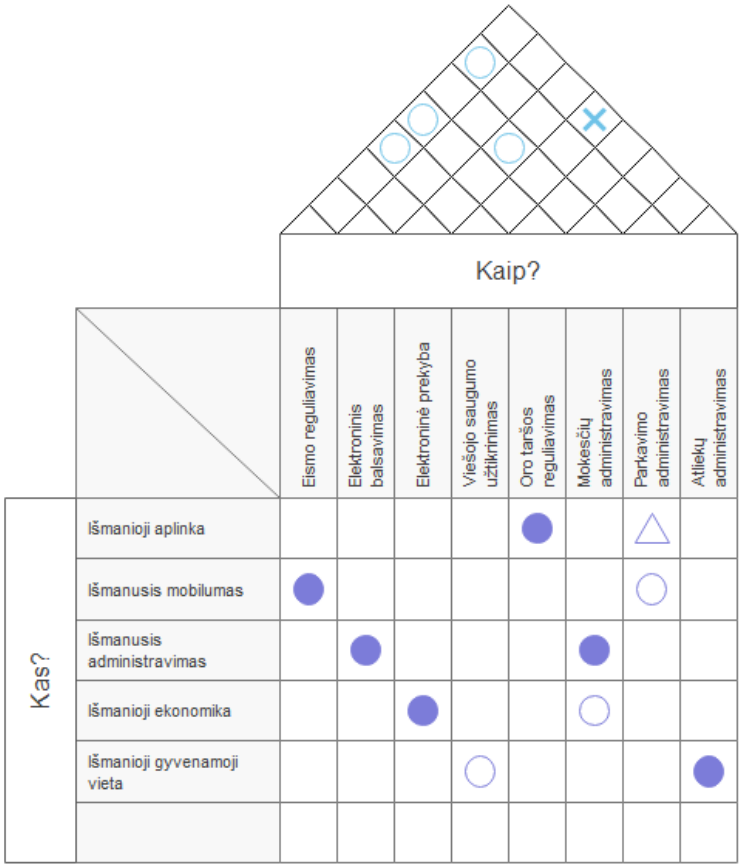
19 pav. Išmaniojo miesto moduliai

Kad būtų sukurti šie moduliai, reikalinga išskirti poreikius, kuriuos reikia įgyvendinti. Todėl šioje dalyje iškeliami poreikiai posistemėms. Poreikių sąrašą galėtų sudaryti tokie elementai:

1. Oro taršos reguliavimas
2. Eismo reguliavimas
3. Mokesčių administravimas
4. Elektroninis balsavimas
5. Elektroninė prekyba

6. Viešojo saugumo užtikrinimas
7. Parkavimo administravimas
8. Atliekų administravimas

Sąrašas gali būti tęsiamas ir toliau, bet tam, kad iliustruotume kokybės namo pritaikymą, tai yra pakankamas elementų kiekis.



20 pav. Ryšiai tarp išmaniojo miesto vizijos ir abstrakčių sistemos poreikių

Toliau kokybės namelius nagrinėsime tik per vienos posistemės – eismo valdymo prizmę. Ši posistemė turi tu aukšto lygio tikslus – reguliuoti eismą bei administruoti automobilių stovėjimą. Tokia posistemė taip pat bus kibernetinė–fizinė–socialinė nes apims daugelį sistemų ir jutiklių, kurie tarpusavyje sąveikaudami užtikrins saugų ir efektyvų eismą.

Ši posistemė turės daug artefaktų sluoksnių. Jei imsime vien eismo reguliavimą, tai šis iki įgyvendinimo pereis per dar kelis sluoksnius. Pradžioje detalizuokime smulkesnes funkcijas, kurios turi būti atliktos, o pagal tai turėtų būti keliami funkciniai bei techniniai reikalavimai sistemos įrangai bei sąveikai tarp skirtingų sistemos elementų.

Panagrinėkime šiuose reikalavimus. Surašykime techninius bei funkcinius reikalavimus kiekvienai iš funkcijų.

21 pav. Aukšto abstrakcijos lygmens reikalavimo detalizacija į smulkesnius reikalavimus

1. Apskaičiuoti optimalią eismo valdymo strategiją
   (a) Įvertinti eismo judėjimo duomenis
   (b) Parinkti eismo valdymo strategiją
2. Reguliuoti šviesoforus pagal pasirinktą strategiją
   (a) Apskaičiuoti šviesoforų keitimosi algoritmą
   (b) Keisti šviesoforų spalvas
3. Gauti informaciją iš daviklių apie eismo užimtumą
   (a) Perduoti duomenis iš daviklių į centrinę eismo valdymo sistemą
   (b) Fiksuoti automobilių srautą
4. Pakeisti kelio ženklinimą pagal pasirinktą strategiją
   (a) Pakeisti kelių eismo ženklus
5. Fiksuoti kelių eismo pažeidimus

(a) Registruoti automobilių judėjimą

(b) Įvertinti automobilių greitį

6. Išrašyti baudas pažeidėjams

(a) Pritaikyti baudą pažeidėjui

(b) Išsiųsti pranešimą pažeidėjui

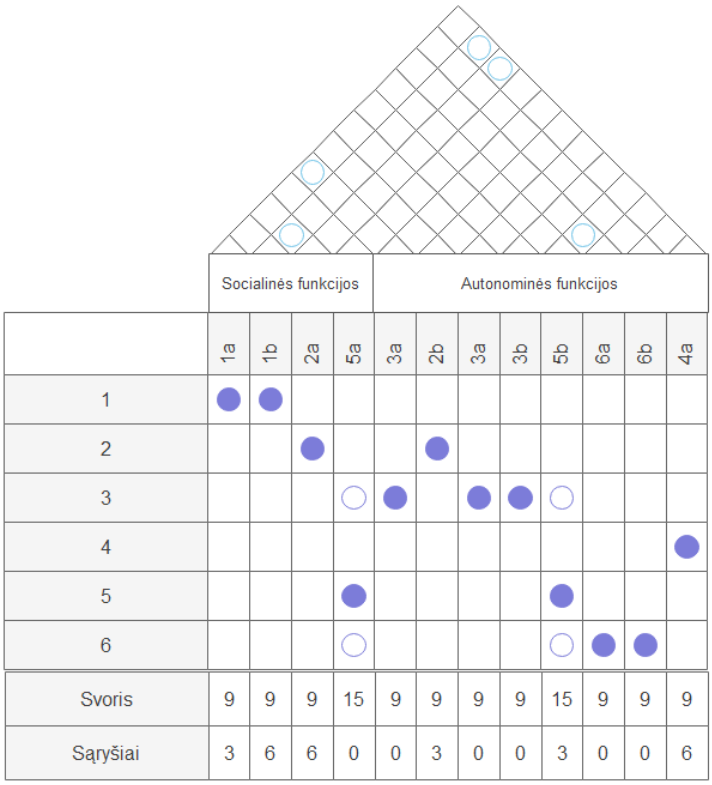
Nupieškime šiems reikalavimams kokybės namelį (22 paveiksle). Įvertinkime tai, kad detalesni reikalavimai gali būti atskirti į dvi dalis – centrinio valdymo sistemos (t. y. tai, kas atlieka sistemų tarpusavio sąveikos vaidmenį) ir autonomines funkcijas, kurios šiame kontekste reiškia, jog sistemos komponentas pats atlieką tam tikrą veiksmą.

Kai kurie iš funkcijų reikalavimų yra tarpusavyje susiję. Užpildydami šiuos ryšius namo stoge, galime matyti, kurios funkcijos yra susijusios. Sąryšiai tarp reikalavimų gali būti teigiami, stipriai teigiami, neigiami arba stipriai neigiami. Neigiami ryšiai nurodo, kad reikalavimai vienas kitam prieštarauja ir gali būti neįgyvendinami, o teigiami nurodo, kad vienas kitą papildo. Šiuo atveju matome, kad tam, jog būtų įgyvendinti 1a ir 1b reikalavimai reikalingi daviklių duomenys, todėl tarp šių reikalavimų yra teigiami sąryšiai.

Kuo žemesnio lygmens reikalavimas susijęs su daugiau aukštesnio lygmens reikalavimų, tuo sąryšių (centrinėje) matricoje matysime daugiau sąryšių. Šioje matricoje gali būti pažymėti trijų lygių sąryšiai. Labai stiprus, stiprus ir silpnas. Priešingai negu tarp vieno lygmens reikalavimų, čia negali būti neigiamų sąryšių. Priskyrus svertines reikšmes kiekvienam sąryšiui (+9 labai stiprus, +3 stiprus ir +1 silpnas) galime apskaičiuoti kiekvienos funkcijos svorį. Šiuo atveju matome, kad registruoti automobilių judėjimą yra didžiausio svorio funkcija, o tai reiškia, kad ji susijusi su daugiausia aukštesnio lygio reikalavimų. To pasekoje, galime šią funkciją vertinti kaip kertinę tam, kad būtų įgyvendintas visos posistemės funkcionalumas.

Kitu atveju, jei tarp reikalavimo ir įgyvendinimo nėra apskritai jokio sąryšio, galima teigti dvieju tipų teiginius. Jei reikalavimas neturi jokio sąryšio su realizaciniais elementais, tuomet šis reikalavimas lieka neįgyvendinamas. Tuo atveju, jei realizacinis elementas neturi sąryšio su jokiu reikalavimu, galima teigti, jog realizacinis elementas yra nereikalingas (perteklinis), nes jis neįgyvendina jokio aukštesnio lygio reikalavimo.

Šie sąryšiai taip pat atsispindės matricos apatinėje dalyje, nes jei realizacinis elementas neturės jokio sąryšio su aukštesnio lygio reikalavimu, jo svoris bus nurodytas kaip 0, o tai indikuoja, kad elementas nėra reikalingas.



22 pav. Kokybės namas, parodantis sąryšius tarp reikalavimų

Tokiu praktiniu pavyzdžiu iliustruojame kokybės namų taikymą trasuojant reikalavimus kibernetinėse–fizinėse–socialinėse sistemose. Kaip matome, šis įrankis yra pravartus, kuomet turime vieno lygmens, tačiau skirtingų kategorijų reikalavimų.

Namai kuriami visais lygmenimis. Kuriant realią išmaniojo miesto sistemą, reikėtų daug namų lygmenų. Nors standartiniu atveju darbe buvo apibrėžta 5 namų struktūra, realioje tokio tipo sistemoje namų lygmenų būtų kur kas daugiau. Taip pat, kadangi reikalavimų kiekis būtų didelis, reikėtų skaldyti vieno lygmens namus į keletą skirtingų, nes didelėse matricose lengva pasimesti.

Skyriaus apibendrinimas

Šiame skyriuje buvo pateiktas pavyzdys, kaip naudojantis kokybės funkcijų sklaidos modifikuotu kokybės namu galima trasuoti reikalavimus realioje sistemoje. Kaip sistemos pavyzdys pasirinkta išmaniojo miesto sistema susidedanti iš kelių skirtingų modulių. Skyriuje pateiktos kokybės namo matricos bei aprašyti žingsniai, kuriuos atliekant trasuojami reikalavimai.

4. Rezultatai ir išvados

4.1. Rezultatai

Darbe buvo pasiekti šie rezultatai:

1. Nustatyti trasavimo uždavinio ypatumai kibernetinėse–fizinėse–socialinėse sistemose.
2. Sudarytas sąrašas artefaktų, kurie sudaro kibernetines–fizines–socialines sistemas.
3. Kokybės funkcijų sklaidos metodas pritaikytas trasuoti reikalavimus kibernetinėse–fizinėse–socialinėse sistemose.
4. Pateiktas praktinis trasavimo pavyzdys naudojant kokybės namą.

4.2. Išvados

Darbe buvo padarytos šios išvados:

1. Atlikus literatūros analizę ir realių sistemų stebėjimą nustatyta, kad kibernetinėse–fizinėse–socialinėse sistemose kiekvienas komponentas gali veikti ir autonomiškai, ir visos sistemos lygmenyje, o taip pat tokios sistemos yra labai didelės apimties, todėl įprasti trasavimo metodai šioms sistemoms netinka.
2. Kokybės funkcijų sklaidos metodas ir konkreti jo technika kokybės namas (angl. House of Quality) turi būti naudojamas trasuoti reikalavimus kibernetinėms–fizinėms–socialinėms sistemoms, nes
   (a) tai įgalina susieti autonominius ir bendrus sistemos reikalavimus tarpusavyje;
   (b) kiekybinės metrikos leidžia nustatyti reikalavimų įgyvendinimo prioritetus.
3. Norint pritaikyti kokybės namą reikalavimams trasuoti, iš jo turi būti pašalinta konkurentų matricos dalis.

Literatūra

[AY15] Ahmed Mubark Alsalemi ir Eng-Thiam Yeoh. A survey on product backlog change management and requirement traceability in agile (scrum). *2015 9th Malaysian Software Engineering Conference (MySEC)*, p. 189–194, 2015. DOI: 10.1109/MySEC.2015.7475219.

[BLT+12] David Broman, Edward Lee, Stavros Tripakis ir Martin Törngren. Viewpoints, formalisms, languages, and tools for cyber-physical systems. P. 49–54, 2012-10. DOI: 10.1145/2508443.2508452.

[BS14] Manfred Broy ir Albrecht Schmidt. Challenges in engineering cyber-physical systems. *Computer*, 47:70–72, 2014-02. DOI: 10.1109/MC.2014.30.

[Car00] Joseph Carr. Requirements engineering and management: the key to designing quality complex systems. *The TQM Magazine*, 12:400–407, 2000-12. DOI: 10.1108/09544780010351760.

[CCM03] Jane Cleland-Huang, Carl Chang ir Christensen MJ. Event-based traceability for managing evolutionary change. *Software Engineering, IEEE Transactions on*, 29:796–810, 2003-10. DOI: 10.1109/TSE.2003.1232285.

[CI10] Antonio Cordella ir Federico Iannacci. Information systems in the public sector: the e-government enactment framework. *The Journal of Strategic Information Systems*, 19(1):52–66, 2010. ISSN: 0963-8687. DOI: https://doi.org/10.1016/j.jsis.2010.01.001. URL: https://www.sciencedirect.com/science/article/pii/S0963868710000028.

[CSD+05] Jane Cleland-Huang, Raffaella Settimi, Chuan Duan ir Xuchang Zou. Utilizing supporting evidence to improve dynamic requirements traceability. P. 135–144, 2005-01. ISBN: 0-7695-2425-7. DOI: 10.1109/RE.2005.78.

[DP98] Ralf Dömges ir Klaus Pohl. Adapting traceability environments to project-specific needs. *Commun. ACM*, 41(12):54–62, 1998-12. ISSN: 0001-0782. DOI: 10.1145/290133.290149. URL: https://doi.org/10.1145/290133.290149.

[Dre18] Falko Dressler. Cyber physical social systems: towards deeply integrated hybridized systems. P. 420–424, 2018-03. DOI: 10.1109/ICCNC.2018.8390404.

[DSH16] B. Dowdeswell, R. Sinha ir E. Haemmerle. Torus: tracing complex requirements for large cyber-physical systems. *2016 21st International Conference on Engineering of Complex Computer Systems (ICECCS)*, p. 23–32, 2016. DOI: 10.1109/ICECCS.2016.013.

[Fin00] *ICSE '00: Proceedings of the Conference on The Future of Software Engineering*. Limerick, Ireland, 2000. Association for Computing Machinery. ISBN: 1581132530.

[Fin12] A. Finkelstein. Requirements and relationships: a foreword," software and systems traceability. *Springer London*, 6, 2012.

[GF94] O. C. Z. Gotel ir C. W. Finkelstein. An analysis of the requirements traceability problem. *Proceedings of IEEE International Conference on Requirements Engineering*, p. 94–101, 1994. DOI: 10.1109/ICRE.1994.292398.

[Gov96] C.P.M. Govers. What and how about quality function deployment (qfd). *International Journal of Production Economics*, 46-47:575–585, 1996. ISSN: 0925-5273. DOI: https://doi.org/10.1016/0925-5273(95)00113-1. URL: https://www.sciencedirect.com/science/article/pii/0925527395001131. Proceedings of the 8th International Working Seminar on Production Economics.

[GPG+14] Volkan Gunes, Steffen Peter, Tony Givargis ir Frank Vahid. A survey on concepts, applications, and challenges in cyber-physical systems. *KSII Transactions on Internet and Information Systems*, 8:4242–4268, 2014-12. DOI: 10.3837/tiis.2014.12.001.

[Hok01] Mia Hokkanen. Requirements traceability, 2001.

[Hor14] Imre Horvath. *What the design theory of social-cyber-physical systems must describe, explain and predict?* 2014-01, p. 99–120. ISBN: ISBN 978-1-4471-6338-1. DOI: 10.1007/978-1-4471-6338-1_2.

[YMZ20] Dao Yin, Xinguo Ming ir Xianyu Zhang. Understanding data-driven cyber-physical-social system (d-cpss) using a 7c framework in social manufacturing context. *Sensors (Basel, Switzerland)*, 20, 2020-09. DOI: 10.3390/s20185319.

[Jai07] Sanjay Jain. An agile methodology for requirements elicitation an agile methodology for requirements elicitation. 2007-01.

[JLW+15] Dongyao(Tony) Jia, Kejie Lu, Jianping Wang, xiang zhang xiang ir Xuemin Shen. A survey on platoon-based vehicular cyber-physical systems. *IEEE Communications Surveys and amp Tutorials*, 18, 2015-03. DOI: 10.1109/COMST.2015.2410831.

[KKK+08] Vassilka Kirova, Neil Kirby, Darshak Kothari ir Glenda Childress. Effective requirements traceability: models, tools, and practices. *Bell Labs Technical Journal*, 12:143–157, 2008-12. DOI: 10.1002/bltj.20272.

[KM14] S.K. Khaitan ir James Mccalley. Design techniques and applications of cyberphysical systems: a survey. *IEEE Systems Journal*, 9:1–16, 2014-07. DOI: 10.1109/JSYST.2014.2322503.

[KM18] Matthew Krugh ir Laine Mears. A complementary cyber-human systems framework for industry 4.0 cyber-physical systems. *Manufacturing Letters*, 15, 2018-02. DOI: 10.1016/j.mfglet.2018.01.003.

[Kra14] E. Krancevičiūtė. Kokybės funkcijos išskleidimo metodologijos taikymas buhalterinių paslaugų kokybės gerinimui. 2014.

[KS09] A Kannenberg ir Hossein Saiedian. Why software requirements traceability remains a challenge. *The Journal of Defense Software Engineering*, 22:14–19, 2009-07.

[Lea80] Jonathan Lear. *Aristotle and Logical Theory*. Cambridge, England: Cambridge University Press, 1980.

[Lef96] Dean A. Leffingwell. The role of requirements traceability in system development. 1996.

[Lei01] Virve Leino. Documenting requirements traceability information: a case study, 2001.

[LR03] Meir M. Lehman ir Juan F. Ramil. Software evolution—background, theory, practice. *Information Processing Letters*, 88(1):33–44, 2003. ISSN: 0020-0190. DOI: https://doi.org/10.1016/S0020-0190(03)00382-X. URL: https://www.sciencedirect.com/science/article/pii/S002001900300382X. To honour Professor W.M. Turski's Contribution to Computing Science on the Occasion of his 65th Birthday.

[LS01] Virve Leino ir R. Sulonen. Documenting requirements traceability information : a case study master ' s thesis. 2001.

[MSL+13] Sirajum Munir, John Stankovic, Mike Liang ir Shan Lin. Cyber physical system challenges for human-in-the-loop control. 2013-06.

[NH07] Aamrah Naqvi ir Syed Hyder. End-to-end requirement traceability through contribution structures and requirements traceability matrices. *Journal of Engineering and Sciences ISSN 2306-8620*, 1:1–7, 2007-01.

[PE12] Birgit Penzenstadler ir Jonas Eckhardt. A requirements engineering content model for cyber-physical systems. 2012-09. DOI: 10.1109/RES4.2012.6347692.

[PZJ+16] Hervé Panetto, Milan Zdravkovic, Ricardo Jardim-Goncalves, David Romero, J. Cecil ir István Mezgár. New perspectives for the future interoperable enterprise systems. *Computers in Industry*, 79:47–63, 2016. ISSN: 0166-3615. DOI: https://doi.org/10.1016/j.compind.2015.08.001. URL: http://www.sciencedirect.com/science/article/pii/S0166361515300312. Special Issue on Future Perspectives On Next Generation Enterprise Information Systems.

[R02] Stancikas E. R. Kokybės vadyba lietuvos integracijos į europos sąjungą procese. 2002.

[Ram12] Lachana Ramingwong. A review of requirements engineering processes, problems and models. *International Journal of Engineering Science and Technology*, 4, 2012-06.

[RCS+09] Akshay Rajhans, Shang-Wen Cheng, Bradley Schmerl, David Garlan, Bruce Krogh, Clarence Agbi ir Ajinkya Bhave. An architectural approach to the design and analysis of cyber-physical systems. *ECEASST*, 21, 2009-11. DOI: 10.14279/tuj.eceasst.21.286.

[RJ01] B. Ramesh ir M. Jarke. Toward reference models for requirements traceability. *IEEE Transactions on Software Engineering*, 27(1):58–93, 2001. DOI: 10.1109/32.895989.

[RLS+10] Ragunathan Rajkumar, Insup Lee, Lui Sha ir John Stankovic. 44.1 cyber-physical systems: the next computing revolution. *Proceedings - Design Automation Conference*:731–736, 2010-01. DOI: 10.1145/1837274.1837461.

[She14] Tafadzwa Shereni. Effective and efficient requirement traceability in the software development and information technology industry, 2014.

[SHM+98] Xiping Song, William Hasling, Gaurav Mangla ir William Sherman. Lessons learned from building a web-based requirements tracing system. P. 41–50, 1998-01. DOI: 10.1109/ICRE.1998.667807.

[STD21] Emre Sülün, Eray Tüzün ir Uğur Doğrusöz. Rstrace+: reviewer suggestion using software artifact traceability graphs. *Information and Software Technology*, 130:106455, 2021. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2020.106455. URL: https://www.sciencedirect.com/science/article/pii/S0950584920300021.

[Tou03] Stephen E. Toulmin. *The Uses of Argument*. Cambridge University Press, 2 leid., 2003. DOI: 10.1017/CBO9780511840005.

[Ver05] Bleistein Verner Cox. Requirements engineering and software project success: an industrial survey in australia and the u.s. 13, 2005. DOI: 10.3127/ajis.v13i1.73. URL: https://journal.acs.org.au/index.php/ajis/article/view/73.

[Wie95] R. Wieringa. An introduction to requirements traceability. 1995.

[WRS09] Krzysztof Wnuk, Björn Regnell ir Claes Schrewelius. Architecting and coordinating thousands of requirements – an industrial case study. P. 118–123, 2009-06. ISBN: 978-3-642-02049-0. DOI: 10.1007/978-3-642-02050-6_10.

[ZYL+16] Jing Zeng, Laurence Yang, Man Lin, Huansheng Ning ir Jianhua Ma. A survey: cyber-physical-social systems and their system-level design methodology. *Future Generation Computer Systems*, 105, 2016-08. DOI: 10.1016/j.future.2016.06.034.