VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

PROGRAMŲ SISTEMŲ KATEDRA

Programų kūrimo proceso vertinimas

Software development process assessment

Magistro baigiamasis darbas

Atliko: Aivaras Šilalė (parašas)

Darbo vadovas: asist. dr. Stasys Peldžius (parašas)

Recenzentė: doc. dr. Audronė Lupeikienė (parašas)

Vilnius – 2018

Santrauka

Programų kūrimo procesų vertinimo modeliai – reikalingi ir puoselėjami jau ne vieną dešimtmetį. Paskutiniais dešimtmečiais jų atsiranda vis daugiau ir vis kažkuo pranašesnių už pirmtakus. Tačiau procesų vertinimas išlieka ganėtinai subjektyvus, o ir oficialių įrankių rinkoje nėra. Praktikoje dažniausiai naudojamasi „Excel“ programine įranga.

Darbe gilinamasi į ontologijas ir jų savybes. Taip pat apžvelgiamas CMMI-DEV 1.3 procesų vertinimo modelis bei apskritai pats procesų vertinimas, kam tai reikalinga.

Darbe aprašomi įrankiai paremti ontologijomis, kurie buvo kurti siekiant palengvinti programų kūrimo procesų vertinimą. Sukuriama programų kūrimo procesų vertinimo modelių ontologija. Pagal ją aprašoma CMMI-DEV 1.3 modelio dalis, kuri yra išskaidoma iki labai žemo lygio, apimančio praktikos veiklas. Parodoma, kaip naudojantis ontologijomis galima aprašyti procesų vertinimo modelius. Pateikiami pavyzdžiai, kaip toks vertinimas padeda įvertinti įmones objektyviau. Toks įvertinimas padėtų ir įmonėms pačioms save įsivertinti.

Raktažodžiai: Procesų vertinimo modelis, CMMI, ontologija, procesų vertinimas, OWL.

Summary

Software process assessment models have been in active development over the last few decades. The quality of the models have been increasing, however, the overall process assessment task still remains partly subjective. There is also no single official tool in the market for such tasks.

In this work we investigate ontologies and their properties. Also we cover CMMI-DEV 1.3 process assessment model and process assessment in general.

We describe tools based on ontologies, which have a goal of making process assessment easier. A new software process assessment models ontology is created. Based on it, we describe CMMI-DEV 1.3 model parts, which are divided till the lowest level. It is shown how to describe process assessment models by using the created ontology. Also there are examples, how this assessment helps to assess company more objectively. Assessment like this helps companies to do self-assessment.

Keywords: Software process assessment, CMMI, ontology, process assessment, software engineering.

Turinys

Įvadas

Nuo pat programinės įrangos kūrimo pradžios įmonės susidurdavo su problemomis, jog projektai vėluoja, biudžetas yra viršijamas, o klientai galiausiai lieka nepatenkinti produktų kokybe [Mcq12].

Praėjusio amžiaus paskutiniais dešimtmečiais vyko tikra programinės įrangos kūrimo krizė. Nors per pastaruosius dešimtmečius situacija labai pagerėjo, kurią atspindi Standish Group atliekami tyrimai. Pagal šių tyrimų, kurie apima duomenis nuo 1994 metų iki nūdienų galime pasidžiaugti, kad sėkmingai įgyvendintų projektų skaičius išaugo nuo 16% iki 39% [CM01][CM13][CM15]. Pradėjus gilintis į projektų nesėkmių faktorius buvo pastebėta, kad nesėkmių priežastys nėra tik technologinės, bet lygiai tiek pat svarbu yra ir netinkamas programų kūrimo procesas.

Tobulėjant programų procesų kūrimui keitėsi ir užsakovų poreikiai. Būtent programinės įrangos užsakovai ir buvo programų kūrimo procesų vertinimo modelių atsiradimo iniciatoriai [Hum93]. Pritaikant modelius įmonėms buvo pastebėta, kad kuo įmonės proceso branda/gebėjimas yra didesnis, tuo įmonėms iškyla mažiau problemų su projektais. Ženkliai sumažėja visos sąnaudos, mažiau aptinkama defektų bei yra žymiai lengviau nustatyti projekto biudžetą bei terminą [HW04].

Vienas populiariausių procesų vertinimo modelių yra CMMI [SEI10]. Iki šiol yra sudėtinga skaityti tokius vertinimus. Surasti silpnąsias vietas.

OWL – Žiniatinklio (pasaulinio tinklo) ontologijos kalba (angl. Web Ontology Language). OWL buvo sukurta daugiau įvairiems įrenginiams, programoms suprasti informaciją, kuri yra patalpinta internete. W3C organizacija apibūdina OWL kaip vieningą semantinio žymėjimo kalbą skirtą aprašyti ontologijoms bei santykiams tarp jų, kuriomis bus galima dalintis visame pasaulyje per pasaulinį tinklą. OWL kasdien plačiai naudojama beveik kiekvienoje gyvenimo srityje [GUI05]. Nuo astronomijos ar šalies saugumo iki medicinos, biologijos ar autonominių robotų. Įvairiausiose industrijose [GUI05]. Visame pasaulyje kuriamos įvairios ontologijos. Vien medicinos srityje naudojama įvairiems tikslams, pavyzdžiui apsirašius ontologiją ir naudojantis semantika yra sukurta tokių įrankių kaip: pacientų duomenų bazė, kuri lengvai pasiekiama tarp įvairių klinikų. Taip pat įrankis, kuris padeda nustatyti ar transplantuojamas organas bus priimtas ar atmestas imuninės sistemos [SMN+11]. Dar vienas puikus pavyzdys GoodRelation žodynas, kurį naudoja dešimtys tūkstančių įmonių. Šis žodynas paremtas semantika ir padeda atlikti tikslesnes ir greitesnes paieškas. GoodRelationss žodyną naudoja tokios organizacijos kaip VolksWagen, Google, Yahoo ir t.t. VolksWagen šio žodyno dėka apsirašo visas prekes [GSO+11]. Google šį žodyną naudoja norėdama

pagerinti paiešką, padaryti ją labiau semantinę. Taip pat IMDB – filmų duomenų bazė. Ji taip pat naudojasi su OWL parašyta ontologija.

Naudotis OWL kalba aprašytomis ontologijomis yra labai patogu dėl jų lanksčios paieškos, kuri yra atliekama su SPARQL užklausomis. Būtent SPARQL užklausas dažniausiai naudoja duomenų paieškoms, nes nereikia žinoti stulpelių pavadinimų ir kitų niuansų.

Šio darbo tikslas - sukurti programų kūrimo proceso vertinimo ontologijos metmenis.

Šiam tikslui pasiekti bus sprendžiami šie uždaviniai:

- CMMI-DEV 1.3 proceso sričių bei specifinių praktikų analizė.
- Esamų procesų vertinimo modelių ontologijų analizė.
- Sukuriama ontologija pagal OWL anotaciją, kuri apimtų pasirinktą kiekį proceso sričių.
- Ontologijos pritaikymas CMMI-DEV 1.3 modeliui.
- Proceso vertinimas naudojantis sukurta ontologija.

Vertinimo įmonė gali praplėsti ontologiją ir pagal ją apsirašyti kokį nori procesų vertinimo modelį. Pilnai aprašius modelį būtų labai patogu ieškoti konkrečių veiksmų, kurie turi būti atliekami norint įgyvendinti tam tikrą praktiką. Tai leistų atlikti paieškas, kurios parodytų kokia pareigybė yra atsakinga už tam tikrus veiksmus. Aprašius modelį ir turint vertinimo duomenis, juos žymiai paprasčiau analizuoti, nei su „excel“, ar reliacinėje duomenų bazėje.

1. Procesų vertinimo modeliai

Praeito amžiaus nuo aštunto dešimtmečio vyko tikra programinės įrangos kūrimo krizė. Projektai vėlavo, didelė jų dalis buvo išvis nepabaigiami. Atsirandant vis daugiau programinę įrangą kuriančių įmonių, poreikiai toms įmonėms darėsi vis didesni. Reaguodami į susidariusią situaciją tokie mokslininkai kaip Edward Yourdon, Tom DeMarco, Larry Constantine, David Parnas, Gerald Weinberg vykdė programinės įrangos kūrimo procesų gerinimo tyrimus bei juos aprašė išleistose knygose bei straipsniuose. Jų tikslas buvo kaip įmanoma labiau pagerinti programinės įrangos kūrimo procesą bei jo kokybę. [PWC+93]

Ši programų procesų kūrimo krizė tęsėsi iki praėjusio amžiaus paskutinio dešimtmečio. Kokybė kažkiek gerėjo, tačiau, tačiau tokių produktų, kurie būtų sukurti sėkmingai ir tikrai naudojami buvo labai nedaug. Projektai buvo tragiškai nesėkmingi, o tai turėjo labai didelę reikšmę tiek piniginiu, tiek laiko atžvilgiu suinteresuotoms šalims. Pateiksiu kelis pavyzdžius, kad būtų galima lengviau įsivaizduoti:

- 1996 m. Arian-5 erdvėlaivis, kurį pagaminti kainavo 7 milijardus dolerių bei užtruko beveik 10 metų, buvo sunaikintas per minutę nuo paleidimo. Nelaimė įvyko dėl programuotojų paliktos klaidos.
- Per Persijos įlankos karą 1991 JAV naudojo raketas apsiginti nuo raketų, kurias leido Irako pusė. Deja, bet raketos tinkamai neveikė ir dėl to žuvo 28 Jungtinių valstijų kariai. Klaida vėlgi buvo iš programuotojų pusės.

Kaip praktika rodė, tai netgi sukurtus produktus būdavo sunku vadinti sėkmingais, nes jie viršydavo ir piniginius kaštus, ir laiko terminus. Tuomet buvo atliekami įvairiausi tyrimai, kol buvo prieita prie išvados, kad būtent nuo programinės įrangos kūrimo kokybės priklauso pačio produkto kokybė.

Programinę įrangą kuriančios įmonės susiduria su begale problemų ir iššūkių, pagrindiniai jų:

- Viršijamas biudžetas
- Prasta produktų kokybė
- Projektai vėluoja

Pagal Standish Group atliktų tyrimų rezultatus galime pamatyti kaip evoliucionavo ir kito programinių produktų sėkmingumas.

1 lentelė. Projektų įgyvendinimas [CM15][CM01][CM15]

	1994	1996	1998	2000	2011	2012	2013	2014	2015
Sėkmingas	16	27	26	28	29	27	31	28	29
Žlugęs	31	40	28	23	22	17	19	17	19
Pažeistas	53	33	46	49	49	56	50	55	52

Iš lentelės, kuri buvo sudaryta remiantis Standish Group atlikto tyrimo rezultatais galime matyti kaip išaugo sėkmingų projektų skaičius daugiau nei 2 kartus. Taip pat akivaizdu, kad nesėkmingų projektų skaičius sumažėjo beveik dvigubai. Svarbu paminėti, kad pažeistų projektų skaičius neženkliai, bet visgi sumažėjo. Svarbu suprasti kokie projektai yra priskiriami prie šių kategorijų, tad pamėginsiu pateikti apačioje. [CM13][CM15]

- Sėkmingas – projektas tilpo į visus jam nustatytus rėmus. Padarytas laiku ir tilpo į nustatytą biudžetą, o visos anksčiau apibrėžtos funkcijos įgyvendintos.
- Žlugęs – projektas atšauktas prieš jį pabaigiant arba net nepradėjus jo vystyti.
- Pažeistas – projektas įgyvendintas iki galo, bet jis galėjo viršyti biudžetą, būti įgyvendintas pavėluotai, taip pat projekto produkto kūrimo metu buvo atsisakyta tam tikro funkcionalumo.

Atsižvelgdama į nesėkmingus projektus JAV karinės oro pajėgos užsakė sukurti gebėjimo brandos modelį – CMM (angl. *Capability Maturity Model*). Tai buvo pirmas atspirties taškas tokių modelių kūrime.

1.1. CMMI

Iš pradžių buvo kuriami įvairius modeliai įvairioms sritims, bet galiausiai buvo nutarta, kad būtų naudingiau bei efektyviau turėti vieną modelį, kuris galėtų apimti visas sritis. Dėl šios priežasties JAV vyriausybė pasirūpino, kad būtų pradėtas kurti integruotas CMM modelis – CMMI (angl. *Capability Maturity Model Integration*).

1.1.1. Integruoto modelio teikiama nauda

CMMI pagrindinės praktikos yra publikuotos dokumentuose, kurie yra vadinami modeliais. Kiekvienas modelis yra orientuotas į tam tikrą sritį. CMMI Version 1.3 - yra aprašyti trys modeliai, kurie yra orientuoti į kūrimą, įsigijimą bei valdymą ir paslaugų teikimą.

1. CMMI kūrimui (CMMI-DEV). Orientuojasi į prekių bei paslaugų plėtros (kūrimo) procesus.
2. CMMI įsigijimui bei valdymui (CMMI-ACQ). Šiame modelyje yra orientuojamasi į tiekimo grandinės valdymą, užsakomų paslaugų valdymą ties valdžia, ties industrija.
3. CMMI paslaugų teikimas (CMMI-SVC). Visas dėmesys skiriamas paslaugų teikimui tiek įmonės viduje, tiek išorėje.

Žemiau pateiktame 1 paveikslėlyje pavaizduotas tolydžiosios CMMI architektūros modelis.

1.1.2. CMMI architektūros

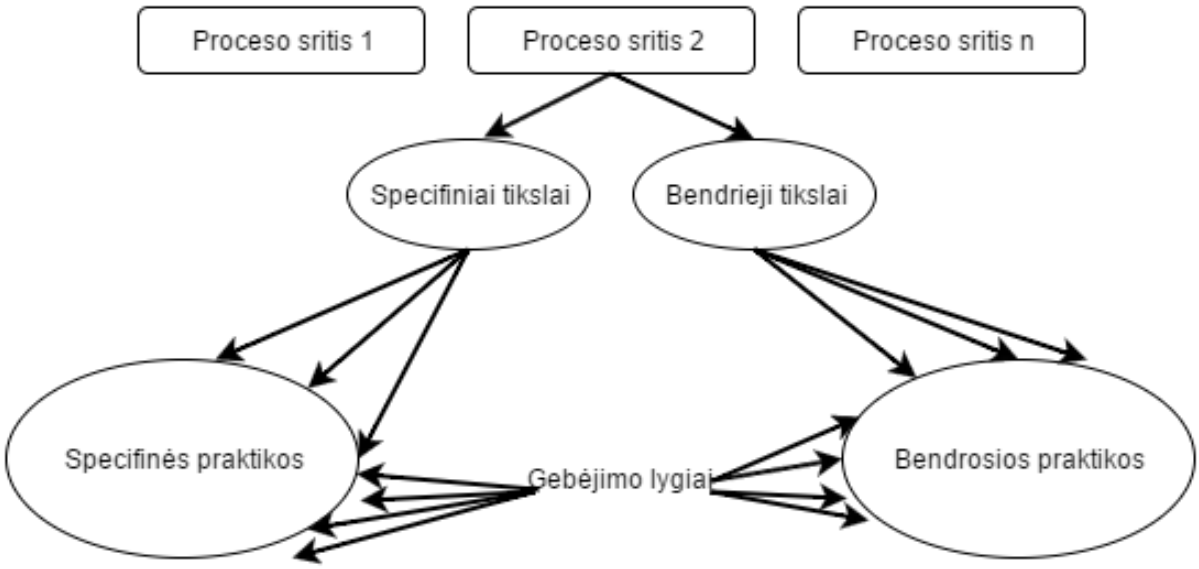
Jau anksčiau rašėme, kad CMMI turi dviejų tipų architektūrą. Organizacija privalo pati nuspręsti, kuris procesų gerinimo tipas jiems tinka labiau. Visame skyriuje autorius remiasi CMMI V1.3 dokumentacija. CMMI-DEV 1.3 versijos modelyje yra apibrėžtos iš viso 22 proceso sritys, kurios yra vienodos ir priklauso abiem architektūros modeliams. Proceso sritys skirstomos į keturias kategorijos.[SEI10]

Proceso sričių kategorijos:

1. Proceso valdymo kategorija – apima 5 sritis.
2. Projekto valdymo kategorija – apima 6 sritis.
3. Inžinierinių procesų kategorija – apima 6 sritis.
4. Palaikymo kategorija – apima 5 sritis.

Tolydžiosios architektūros modelis:

Žemiau pateiktame 1 paveikslėlyje pavaizduotas tolydžiosios CMMI architektūros modelis.



1 pav. Tolydžiosios CMMI architektūros modelis [SEI02]

Svarbu žinoti, kad visi CMMI modeliai su tolydžiąja architektūra atspindi savo gebėjimo lygį pagal savo turinį bei dizainą (kaip projektuoja visą darbų eigą). Gebėjimo lygis susideda iš susijusių specifinių bei bendrųjų praktikų proceso srityje. Būtent tai leidžia organizacijai visą dėmesį nukreipti į vieną proceso sritį arba jų aibę.

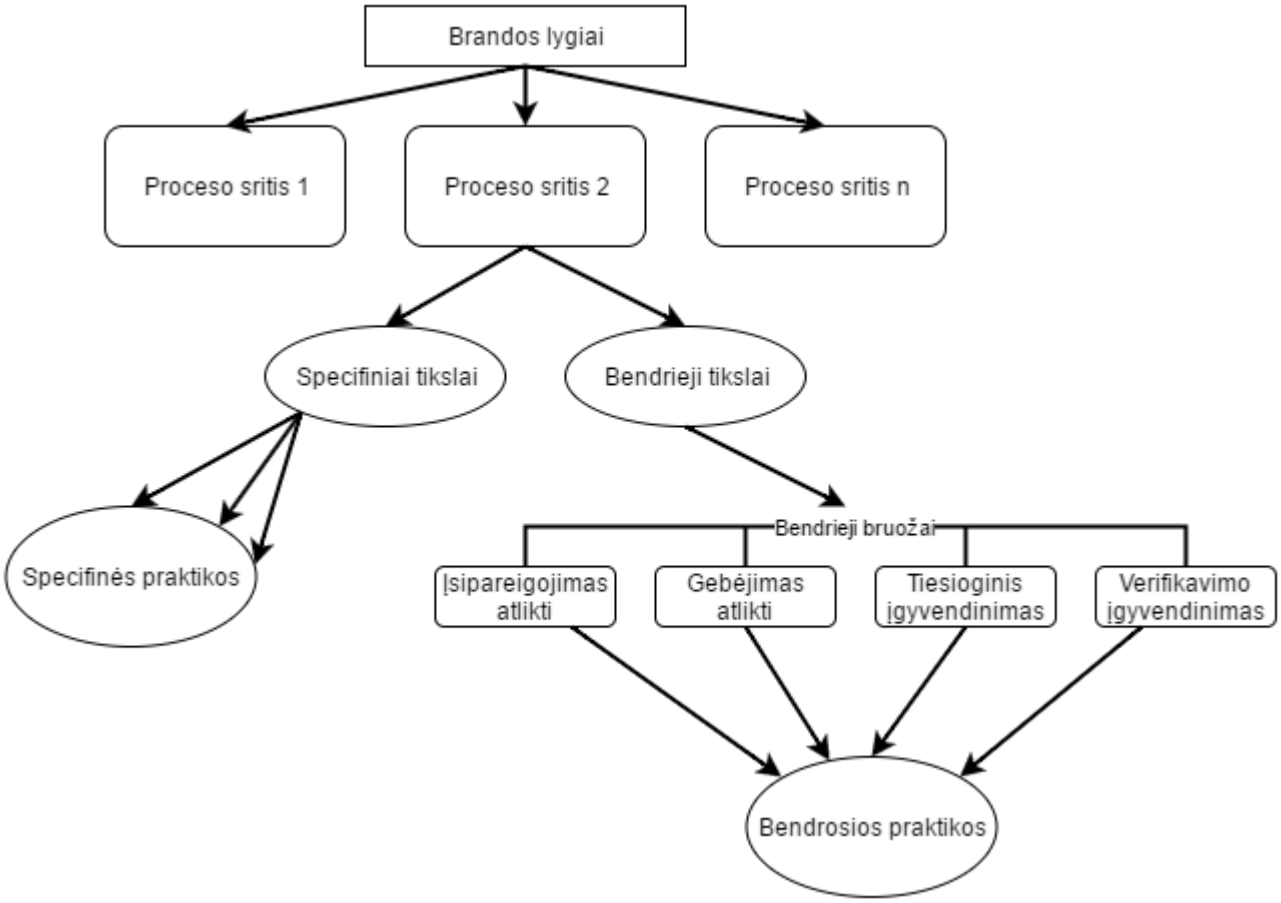
Tolydžiosios architektūros modelis turi 4 gebėjimo lygius, nuo 0 iki 3. Iki naujausios (CMMI V1.3) versijos buvo dar du lygiai – kiekybiškai valdoma ir optimizuojama proceso sritis. **Gebėjimo lygiai** koncentruojasi ties organizacijos savybių, tokių kaip kūrimas, kontroliavimas bei tobulinimas, gerinimu. Šie gebėjimo lygiai leidžia stebėti, vertinti bei demonstruoti procesų patobulinimus. Gebėjimo lygiai yra sudaryti iš bendrųjų tikslų, kuriuos sudaro bendrosios praktikos.

2 lentelė. Gebėjimo lygiai

Lygis	Gebėjimo lygis	Savybės
0	Nevykdomas (angl. *Incomplete*)	• Visiškai neatitinkantis arba dalinai neatitinkantis. Kai vienas arba daugiau specifinių tikslų procesų sričiai neatitinka.
1	Vykdomas (angl. *Performed*)	• Procesas atitinka specifinius tikslus proceso sričiai. • Praktikos gali būti pabaigtos nevisiškai tvarkingai, t.y. nesiremiant pilnai proceso dokumentacija ar planu.
2	Valdomas (angl. *Managed*)	• Praktikos įgyvendintos pagal visus planus ir taisykles. • Organizacijos darbuotojai yra pakankamai kvalifikuoti ir kompetentingi, galintys prisiimti atsakomybę už savo darbą bei jo pasekmes. • Šis gebėjimo lygis įtraukia į darbą abi puses. Tiek klientą tiek organizaciją, kuri užsiima tam tikro programinio produkto kūrimu.

Lygis	Gebėjimo lygis	Savybės
		• Nuo šio gebėjimo lygio visas proceso gerinimas yra stebimas, kontroliuojamas bei vertinamas (prižiūrimas)
3	Apibrėžtas (angl. *Defined*)	• Šis lygis turi prižiūrimą (tobulinamą) procesų aprašymą • Sudaro pagrindą projekto užduočių bei funkcionalumo planavimui, kūrimui bei tobulinimui. • Šį gebėjimo lygį turinčios organizacijos geba kokybiškai nuspėti įvairius aspektus (terminą, kaštus). • Valdomas procesas, kuriam yra pritaikytas visas rinkinys standartinių procesų remiantis pritaikymo

Pakopinės architektūros modelis:



2 pav. pakopinės CMMI architektūros modelis [SEI02]

Aukščiau pateiktame paveikslėlyje yra pavaizduotas CMMI pakopinės architektūros modelis.

Pakopinė CMMI architektūra pasižymi tuo, kad yra dalinama į 5 brandos lygius. **Brandos lygiai** suteikia galimybę nuspėti būsimą organizacijos elgesį pagal tam tikrą apribojimą ar apribojimų rinkinį. Brandos lygis - apibrėžtas evoliucinis brandos stabilizavimas link programinės įrangos

procesų brandos pasiekimo. Kiekvienas lygis stabilizuoja svarbią dalį organizacijos procesų. Siekdama vis didesnio brandos lygio organizacija juda link puikios produktų (paslauga taip pat produktas) kokybės. Pakopinėje architektūroje visai kaip ir tolydžiojoje kiekvienas lygis statomas ant prieš tai jau pastatytų lygių. Todėl organizacijai siekiant 3 brandos lygio ji turės atitikti visus reikalavimus 1 bei antro brandos lygio.

3 lentelė. Brandos lygiai

Lygis	Brandos lygis	Savybės
1	Pradinis (angl. *Initial*)	• Procesai dalinai įgyvendinami, bet viskas labai chaotiška • Įprastai organizacija negali užtikrinti stabilios aplinkos • Pastebima tendencija, kad įmonės neįvertina tinkamai projektų ir jų metų tiesiog jų atsisako. • Šio lygio organizacijos sukuria produktus, kurie veikia, bet labai dažnai viršija numatytą laiką bei biudžetą. • Funkcionalumas taip pat dažnai būna pažeistas ir kažko atsisakyta arba pakeista.
2	Valdomas (angl. *Managed*)	• Šiame lygyje reikalavimai jau yra valdomi ir procesai planuojami, ruošiami, išmatuojami bei kontroliuojami. • Egzistuojančios praktikos išsilaiko netgi būnant dideliam stresui. Procesus vykdo taip, kaip jie yra dokumentuoti. Moka valdyti stresą. • Darbų būsenos yra valdomos. Nustatomi rėžiai iki kada kas turi būti įvykdyta ir visą laiką stebima kaip vyksta procesai, kokioje stadijoje kas yra. • Abi pusės, tiek klientas, tiek organizacija yra įsipareigoję, sutikę su vieni kitų įsipareigojimais. • Pats darbo produktas atitinka jam keltus reikalavimus bei standartus.
3	Apibrėžtas (angl. *Defined*)	• Procesai yra tinkamai charakterizuojami ir suprantami, dėl to jie yra aprašomi standartais, procedūromis, aprašomos technologijos, kurios yra reikalingos. • Valdomi proceso tikslai paremti standartiniais procesais, taip pat yra užtikrinami, kad šie tikslai būtų tinkamai įgyvendinti. • Procesai yra tiksliau bei griežčiau apibrėžiami nei antrame brandos lygyje. • Procesai yra kokybiškai nuspėjami
4	Kiekybiškai valdomas (angl. *Quantitavely Managed*)	• Sub-procesai, kurie yra kontroliuojami naudojant statistinius bei kitus kiekybinius metodus, yra parenkami taip, kad prisidėtų prie bendros procesų veiklos. • Kokybė bei procesų pajėgumas yra suprantami matematiniais (statistiniais) terminais bei valdomi visą jų gyvavimo laiką.

Lygis	Brandos lygis	Savybės
		• Procesai yra išsamiai matuojami ir renkama visa informacija apie juos, kuri yra analizuojama ir panaudojama ateityje siekiant pagerinti procesus bei visą darbų kokybę. • Procesų veikimas yra kontroliuojamas naudojant statistinius ir kitus kiekybinius metodus, ir yra kiekybiškai nuspėjama.
5	Optimizuojamas (angl. *Optimizing*)	• Kiekybiškai valdomas procesas, kuris yra pakeistas ir pritaikytas atitikti esamus ir planuojamus verslo tikslus • Šis gebėjimo lygis koncentruojasi ties proceso našumo gerinimo tiek per pavienių, tiek novatoriškų technologijų patobulinimų. • Šiame gebėjimo lygyje taip pat reikia nesustoti tobulinti procesų, kad būtų galima išvengti įvairių procesų variacijų bei kažkokiu mastu pagerinti tų procesų identifikavimą, įvertinimą bei vykdymą. • Procesų pagerinimai parenkami remiantis statistiniais skaičiavimais, atsižvelgiant į tai, kurie pagerinimai suteiks daugiau naudos organizacijai.

1.1.3. CMMI proceso sričių struktūra

Komponentai yra sudedamosios proceso sričių dalys. Kitaip sakant – kiekviena sritis susidaro iš smulkesnių komponentų. Šie komponentai yra skirstomi į tris grupes.[SEI10]

4 lentelė. Komponentų grupės

Grupės pavadinimas	**Aprašymas**
Privalomi (angl. *required*)	Nurodo privalomus proceso tikslus, kuriuos pasiekti yra privaloma, norint įgyvendinti atitinkamo gebėjimo lygio reikalavimus.
Tikėtini (angl. *expected*)	Nurodo tikėtinus organizacijos veiksmus, kurių ji imasi mėgindama pasiekti privalomus komponentus. Įprastas organizacijai elgesys.
Informaciniai (angl. *informative*)	Nurodo atitinkamą informaciją, kuri organizacijai padeda pasiekti bei suprasti privalomus bei tikėtinus komponentus.

Minėtos kategorijas apima iš viso 10 komponentų. Patys svarbiausi komponentai yra 4:

- **Specifiniai tikslai** (angl. *Specific Goal*) – žymime – SG, šis komponentas patenka į privalomų komponentų grupę. Jis aprašo unikalias charakteristikas, kurias turi pasiekti proceso sritis.

- **Bendrieji tikslai** (angl. *General Goal*) – žymime – GG, šis komponentas patenka į privalomų komponentų grupę. Jis aprašo bendras charakteristikas visoms proceso sritims, kurias privaloma pasiekti.
- **Specifinės praktikos** (angl. *Specific Practices*) – žymime – SP, šis komponentas patenka į tikėtinų komponentų grupę. Jis aprašo konkrečias veiklas, kurios yra privalomos norint pasiekti specifinį tikslą.
- **Bendrosios praktikos** (angl. *General Practices*) – žymime – GP, šis komponentas patenka į tikėtinų komponentų grupę. Jis aprašo konkrečias veiklas, kurios yra reikalingos norint pasiekti susijusį bendrą tikslą.

Visi šie komponentai yra naudojami atliekant proceso srities gebėjimo vertinimą.

1.2. ISO/IEC 15504

ISO/IEC 15504 standartas yra išsivytęs iš SPICE (*Software Process Improvement and Capability dEtermination*), kurio ištakos veda į 1995 metais, kai buvo pristatytas šis procesų vertinimo modelis. Šis modelis įvardija kriterijus, kurie reikalingi norint įgyvendinti norimą procesą. Taip pat proceso sudedamąsias dalis. SPICE buvo sukurtas tokių pasaulinių organizacijų kaip IEC (*International Electrotechnical Commission*) bei ISO (*International Organization for Standardization*).

Iš pradžių tai buvo tik tolydinės architektūros procesų vertinimo modelis, tačiau pastarąjį dešimtmetį – 2008 metais, buvo išleista taip pat ir pakopinės architektūros versija ISO/IEC 15504-7:2008 [ISO07]. Šiuo metu yra išleistų ir naujų versijų ISO/IEC versijų.

1.2.1. Modelio dimensijos

ISO/IEC 15504 modelį sudaro dvi dimensijos – gebėjimo bei procesų.

Gebėjimo dimensija

Procesų gebėjimo dimensija susideda iš 6 lygių (nuo 0 iki 5), kurie susideda nuo nevykdomo, kai proceso tikslai nėra įgyvendinami, iki optimizuojamo – kuomet proceso tikslai gali būti optimizuojami. Atliekant vertinimą, galutinis darbo rezultatas ir yra proceso priskyrimas vienam iš lygių.[Rag07]

5 lentelė. Gebėjimo dimensija

Gebėjimo lygis	Aprašymas
0 lygis	Nevykdomas procesas. Šis lygis nurodo, kad procesas nėra vykdomas arba proceso tikslai nėra pasiekiami.
1 lygis	Vykdomas procesas. Šiame lygyje yra pasiekiami proceso tikslai.
2 lygis	Valdomas procesas. Šiame lygyje galime valdyti proceso atlikimą. Galime jį planuoti, atitinkamai koreguoti ir kontroliuoti rezultatus, kurių tikimės (pirmame lygyje tiesiog pasiekiamas tikslas, bet negalima visiškai valdyti tikslo pasiekimo eigos).
3 lygis	Apibrėžtas procesas. Šis lygis pranašesnis už antrąjį tuo, kad mes pasiekiame užsibrėžtus tikslus.
4 lygis	Prognozuojamas procesas. Šis lygis išsiskiria tuo, kad kitaip, nei trečiame lygyje, mes galima vykdyti procesą apibrėžtose ribose.
5 lygis	Optimizuojamas procesas. Kitaip, nei ketvirtame lygyje – šio lygio procesas yra kažkokiu būdu pastoviai gerinamas bei tobulinamas.

Kiekvienas brandos lygis, išskyrus nulinį, sudaro tam tikrą gebėjimo profilį, kuris susideda iš aibės procesų. Gebėjimo lygis suteikiamas, suteikiamas tik tuomet, kai visi žemesnių lygių proceso atributai įvertinti kaip pilnai pasiekti – F, o esamo lygio proceso atributai gali būti arba pilnai, arba iš dalies pasiekti – F arba L.[Rag07]

Vertinimas

6 lentelė. Vertinimas

Žymėjimas	Vertinimo skalė	Aprašymas
N – Nepasiektas	0 – 15%	Nėra jokių įrodymų, kad bus pasiektas vertinamo proceso atributas.
P – Mažąja dalimi pasiektas	16 – 50%	Yra įrodymų, kad vertinamo proceso atributas dalinai pasiektas (bent mažąja dalimi). Kad santykinai priartėjama prie vertinamo proceso atributo.
L – Didžiąja dalimi pasiektas	51 – 85%	Yra įrodymų, kad vertinamo proceso atributas beveik pasiektas, bet dar turi santykinai nedaug trūkumų.
F – pilnai pasiektas	86 – 100%	Yra įrodymų, kad vertinimo proceso atributas pilnai pasiektas ir visi reikšmingi trūkumai yra pašalinti.

Proceso atributai lygiams

7 lentelė. Proceso atributų bei lygiai

Lygis	Proceso atributai	Įvertinimas
0 lygis	-	-
1 lygis	Proceso atlikimo atributas	L arba F
2 lygis	1 lygio atributai	F
	Proceso vykdymo valdymo atributas	L arba F
	Darbo produktų valdymo atributas	L arba F
3 lygis	1 lygio atributai	F
	2 lygio atributai	F
	Proceso apibrėžimo atributas	L arba F
	Proceso įdiegimo atributas	L arba F
4 lygis	1 lygio atributai	F
	2 lygio atributai	F
	3 lygio atributai	F
	Proceso matavimo atributas	L arba F
	Proceso kontrolės atributas	L arba F
5 lygis	1 lygio atributai	F
	2 lygio atributai	F
	3 lygio atributai	F
	4 lygio atributai	F
	Proceso inovacijos atributas	L arba F
	Proceso optimizavimo atributas	L arba F

Iš pateiktos lentelės aukščiau matome, kad ties kiekvienu lygiu prisidėjo po papildomus du proceso atributus ir kaip buvo minėta anksčiau – kiekvienas lygis, išskyrus nulinį, privalo įgyvendinti pilnai (F) žemesnio lygio atributus.

Procesų dimensija

Proceso dimensijos atveju visos veiklos yra grupuojamos pagal tipą. Praktikos apibrėžia proceso atlikimą, bet ne tai, kaip jis atliekamas. Praktikų atlikimas gali būti nesistemingas, nenusakomas ar netgi prastai suplanuotas, bet svarbiausia, kad rezultatai patenkintų proceso tikslus.

2. Vertinimas pagal kelis vertinimo modelius

Įmonėms pradėjus kreipti dėmesį į kokybę ir norint turėti gerą vardą bei pripažinimą imta labai plačiai naudoti įvairiausios programinės įrangos procesų kūrimo gerinimo metodologijos. Jau tada metodologijų buvo daugiau nei viena. Ir įmonėms atsirado poreikis atitikti daugiau nei vienos metodologijos standartus bei kriterijus. Pavyzdžiui, CMM, CMMI, ISO 27000, ISO 9001, ISO/IEC 15504. Šie modeliai turėjo skirtingus būdus siekti geresnių organizacijos procesų, bet jie visi apibrėžė geriausias programinės įrangos kūrimo praktikas. Deja, bet atlikti vertinimą yra išties santykinai brangu nepaisant organizacija didelė ar maža. Negana to, kad norint atlikti vertinimus pagal skirtingus modelius kainuoja labai daug visų resursų, bet net ir tarp versijų norint žinoti, kiek, pavyzdžiui, modelio X naujesnės versijos įvertinimas būtų pasakyti neįmanoma.

Žemiau lentelėje pateikiami pavyzdiniai duomenys, nurodantys kiek organizacijai kainuoja pasiekti atitinkamus brandos lygius pagal CMM.[Mit17]

8 lentelė. Brandos lygių pasiekimo kaštai [Mit17]

Projekto dydis - 200 000 pradinio kodo eilučių Programuotojo metinis įkainis – 110 000$						
Brandos lygis	Trukmė	Darbo sąnaudos mėn.	Klaidų kiekis	Vidutinė kaina milijonais $	Minimali kaina milijonais $	Maksimali kaina milijonais $
CMM 1	30	600	61	5.5	1.8	>100
CMM 2	18.5	143	12	1.3	0.96	1.7
CMM 3	15	80	7	0.73	0.52	0.93

Pagal pateiktą lentelę matome, kad sudėtingiausia yra pasiekti pirmąjį lygi, tai reikalauja tiek piniginiu, tiek laiko atžvilgiu labai didelių kaštų. Galime matyti, kad tiek trukmė, tiek vidutinė kaina yra net kelis sykius mažesnė, kai norime pasiekti jau antrąjį lygį. Tai patvirtina įsitikinimą, kad sudėtingiausia yra pasiekti pirmą brandos lygį, o po to viskas vyksta daug sklandžiau ir greičiau.

Nepaisant to, kad programinės įrangos procesų gerinimas – SPI (angl. *Software Process Improvement*) yra vienas svarbiausių veiksnių organizacijos procesų gerinime, svarbu atsižvelgti ir į tai, kad visi šie modeliai buvo projektuoti taip, kad būtų pritaikytas tik vienas modelis organizacijai ir net nebuvo minčių, kad tuo pačiu metu organizacija galėtų įgyvendinti kelis modelius. Didžiausią paklausą tarp įmonių turi populiariausieji CMMI bei ISO/IEC 15504 modeliai.

Yra išskiriami pagrindiniai 5 faktoriai, kurie lemia, kad įmonės nori apimti kelis modelius vienu metu:

- Tam tikrų modelių sertifikatų turėjimas nepadidina tiesiogiai pardavimų ir įmonės populiarumo. Sertifikatų turėjimas padidina pasitikėjimą tarp klientų ir organizacijos, kas turi įtakos geresnei pozicijai kaip organizacijai rinkoje.
- Organizacijų susijungimas. Tai nėra toks jau retas dalykas, tad natūralu, kad jeigu organizacijos jungiasi, tai jos gali naudoti skirtingus modelius, tad turi gebėti juos suderinti, kad visa organizacija veiktų kaip vienas darinys, pagal tas pačias procedūras bei taisykles.
- Rinkos su specifinėmis nišomis. Pavyzdžiui įmonės, kurios nori dirbti su NASA užsakymais JAV, turi turėti CMMI-DEV, o jeigu nori dirbti su ESA Europoje, tai joms taip pat reikalingas ir SPICE for SPACE.
- Spartus organizacijos augimas. Organizacijos gali pasinaudoti kitų modelių geriausiomis praktikomis bei privalumais, kad gautų kaip įmanoma didesnę naudą tam tikros srities gebėjimo didinimui.
- Organizacijos pačios nusprendžia naudoti naujus modelius tikėdamos, kad tai joms gali padėti pagerinti visos organizacijos darbų sklandumą, pavyzdžiui sumažinti paslaugų/prekių savikainą arba pritraukti daugiau klientų keliant pasitikėjimą.

Dažniausiai kylanti problema – suderinamumas

Modelius išties sudėtinga suderinti, nes kiekvienas modelis yra savitas ir apibūdina tik patį save, savo procesų struktūrą, esybes, terminologiją, supratimą tarp skirtingų dalykų. Mėginant suderinti kelis modelius yra labai sudėtinga, kadangi atsiranda neatitikimai terminologijoje, metoduose, sąvokose. Būtent dėl šios priežasties mokslininkai pradėjo naudoti ontologijas, kurių dėka jie galėtų susieti skirtingų modelių terminus bei ryšius tarp jų. Pastaraisiais dešimtmečiais buvo mėginta kurti įvairiausius modelius, kuriems užtektų atlikti vieną procesų vertinimą, o tuomet iš jo būtų galima atvaizduoti įverčius pagal skirtingus modelius.

Naudojantis šiomis gairėmis yra sukurtas tarpinis procesų vertinimo modelis (TPAM), kuris suteikia galimybę organizacijai atlikti vertinimą pagal tarpinį modelį, o po to atvaizduoti įverčius pagal norimą modelį. Šiuo metu į modelį yra pridėti du modeliai – CMMI-DEV bei ISO/IEC 15504.[Pel14]

2.1. Tarpinis procesų vertinimo modelis

Autorius kurdamas šį modelį turėjo aiškią idėją, kurią išgrynino ir modelį įgyvendino. Modelis paremtas idėja, kad būtų galima gauti vertinimus pagal skirtingus vertinimo modelius, atlikus vos vieną organizacijos procesų vertinimą, Į kokius modelius norėtume atvaizduoti įvertinimą, pasirenkame patys ir pridedame prie tarpinio modelio. Įtraukiant modelius į TPAM, praktikos skaidosi iki žemiausio lygio – kol jų skaidyti nebeįmanoma. Taip apskaičiuojama kiek TPAM praktika dengia PAM praktika.[PR12]

Darbe yra iškelti šeši reikalavimai, kuriuos pasak autoriaus turi tenkinti TPAM:[Pel14]

1. Vieninga įtraukiamų vertinimo modelių terminija.
2. Vieningas modelių aprašymas.
3. Empirinis ir aprašomasis modelis
4. Naujo procesų vertinimo modelio įtraukimo metodika
5. Vertinimo rezultatų interpretavimas
6. Vertinimo rezultatų atvaizdavimas

TPAM autorius suprasdamas, kad terminija yra viena iš opiausių problemų ėmėsi ją spręsti nuo pat pradžių. Darbe siūloma sukurti ontologiją pagal kurią būtų galima aprašyti visus procesų vertinimo modelius. Žemiau pateiktoje lentelėje matoma kokius terminus priskyrė TPAM ontologijoje ir kaip jie skyrėsi ISO/IEC 15504 bei CMMI modeliuose.

9 lentelė. Terminija tarp modelių

TPAM	ISO/IEC 15504	CMMI
Visuminis procesas (angl. *Organizational Process*)	-	Procesas (angl. *Process*)
Vardinis procesas (angl. *Named Process*)	Procesas (angl. *Process*)	Proceso sritis (angl. *Process Area*)
Tikslas (angl. *Purpose*)	Proceso tikslas (angl. *Process Purpose*)	Tikslo formuluotė (angl. *Purpose Statement*)
Rezultatas (angl. *Outcome*)	Proceso rezultatas (angl. *Process Outcome*)	Specifinis tikslas (angl. *Specific Goal*)
Praktika (angl. *Practice*)	Bazinė praktika (angl. *Base Practice*)	Specifinė praktika (angl. *Specific Practice*)
Bendroji savybė (angl. *General Property*)	Proceso atributas (angl. *Process Attribute*)	Bendrasis tikslas (angl. *General Goal*)

TPAM	ISO/IEC 15504	CMMI
Bendroji praktika (angl. *General Practice*)	Bendroji praktika (angl. *Generic Practice*)	Bendroji praktika (angl. *General Practice*)
Praktikos gebėimas (angl. *Practice Capability*)	-	-
Praktikos svoris (angl. *Practice Weight*)	-	-
Gebėjimo lygis (angl. *Capability level*)	Gebėjimo lygis (angl. *Capability Level*)	Gebėjimo lygis (angl. *Capability Level*)

Taip pat svarbu atkreipti dėmesį į tai, kad buvo įvestos ir naujos sąvokos, pavyzdžiui, „Praktikos svoris“, kuri nurodo kokią dali atvaizduojamos praktikos ji dengia (procentais).

TPAM išsiskiria nuo CMMI bei ISO/IEC 15504 modelių ne tik papildomomis sąvokomis terminologijoje, bet ir tuo, kad apima tik tolydinę reprezentaciją. Tolydinė reprezentacija skirta būtnet vardinių procesų gebėjimui vertinti, o vertinimo rezultatas ir yra procesų gebėjimo profilis. Pastarąjį sudaro gebėjimo lygis kiekvienam vardiniam procesui. [PR13][Pel14]

TPAM yra sudaryta iš 6 gebėjimo lygių, o žemiau esančioje lentelėje yra pavaizduota kaip tie lygiai yra susiejami su CMMI bei ISO/IEC 15504.

10 lentelė. Gebėjimo lygiai tarp modelių

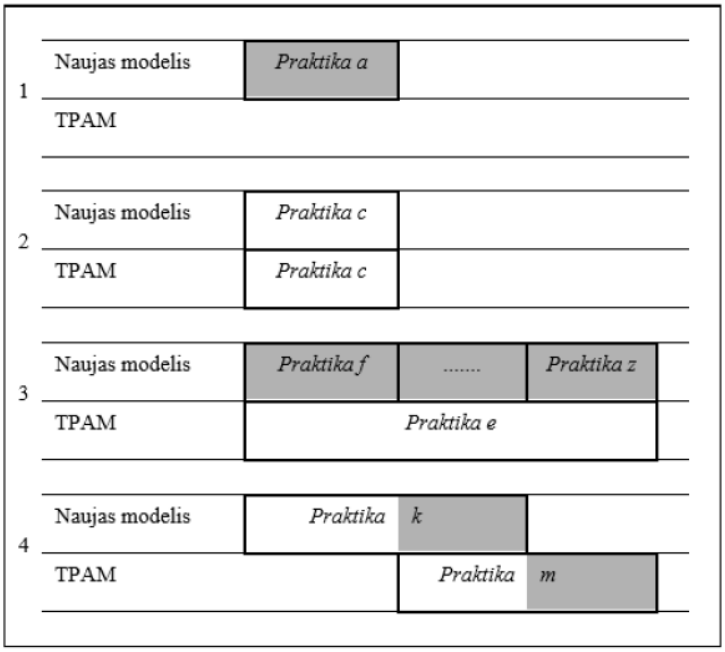
Lygis	**TPAM ir ISO/IEC 15504**	**CMMI**
0	Nevykdomas (*angl. Incomplete*)	Nevykdomas (*angl. Incomplete*)
1	Vykdomas (*angl. Performed*)	Vykdomas (*angl. Performed*)
2	Valdomas (*angl. Managed*)	Valdomas (*angl. Managed*)
3	Apibrėžtas (*angl. Established*)	Apibrėžtas (*angl. Defined*)
4	Prognozuojamas (*angl. Predictable*)	-
5	Optimizuojamas (*angl. Optimizing*)	-

Kadangi TPAM iš pradžių buvo tuščias ir į jį pridėjus ISO/IEC 15504, jų lygiai visiškai sutapo. Deja, bet palyginus su CMMI matosi, kad lygių pavadinimai šiek tiek skiriasi. Nors tokio termino kaip praktikos gebėjimas kituose modeliuose ir nėra, ši sąvoka yra reikalinga norint TPAM išsaugoti įtrauktų naujų modelių bendrųjų praktikų procentinį įvertinimą.

TPAM autorius kaip pagrindą ir patį pirmą modelį, kuris buvo pridėtas į TPAM pasirinko tarptautinį standartą – ISO/IEC 15504-5. Tad iš pradžių TPAM buvo užpildomas šiomis praktikomis ir tada TPAM bei ISO/IEC 15504-5 sutapo 100%. Po to sekė antrasis modelis – CMMI-DEV. Visi kiti modeliai buvo įtraukiami tokia pačia tvarka ir buvo galimi keturi scenarijai:[PR12]

1. CMMI-DEV praktika yra nauja TPAM, tuomet TPAM ši praktika būna pridedama į TPAM.
2. CMMI-DEV praktika pilnai atitinka TPAM esančią praktiką, tuomet niekas nėra pridedama.
3. CMMI-DEV praktikos yra žemesnio lygio ir kur kas detalesnės, nei TPAM praktikos.
4. CMMI-DEV praktika padengia TPAM praktikas tik iš dalies.

TPAM autorius puikiai iliustravo praktikų pasirinkimų variantus žemiau esančiame paveikslėlyje:



3 pav. Praktikų pasirinkimų variantai[Pel14]

TPAM susideda iš dviejų lygmenų, kurie atitinkamai taip pat susideda iš savo lygmenų.

- Vaizdavimo lygmuo – šio lygmens idėja yra apžvelgti TPAM, išnagrinėti vizualiai vardinius procesus, praktikas ir sąryšius su PAM, kurie yra įtraukti į TPAM.

- Vertinimo lygmuo – šio lygmens idėja yra vertinimo rezultatų transformacija į kitus proceso vertinimo modelius. Pavyzdžiui, tai galėtų būti CMMI-DEV į ISO/IEC 15504-5. Šis lygmuo susideda iš trijų dalių:
    - Pirmoji dalis, kuri yra skirta aprašyti tarpinio modelio vardiniams procesams, rezultatams, praktikoms. Ši dalis atitinkamai yra atnaujinama, kai į TPAM yra pridedamas naujas modelis, arba atsinaujina kurio nors modelio versija, kuris yra įtrauktas į TPAM.
    - Antroji dalis, kuri susidaro iš modelių, kurie yra įtraukti į TPAM. Svarbu akcentuoti taisyklę, kuri sako, kad turi būti įgyvendinta pilno uždengimo taisyklė. Ši taisyklė reiškia, kad kiekviena TPAM praktika turi būti pilnai padengta vienos arba kelių praktikų iš įtrauktų modelių.[PR13][Pel14]
    - Trečioji dalis, kuri yra atsakinga už vertinimo rezultatų transformaciją suteikia galimybę atlikti vertinimą pagal TPAM ir pagal poreikį atvaizduoti rezultatus į įtrauktus modelius. Taip pat leidžia atlikti rezultatų transformaciją tarp modelių, pavyzdžiui atlikus įvertinimą pagal CMMI-DEV, juos transformuoti į TPAM ir tuomet šiuos rezultatus transformuoti į kitą modelį, pavyzdžiui ISO/IEC 15504-5

TPAM yra įgyvendintos dvi galimybės atvaizduoti rezultatus:

- NPFL išraiška
- Procentinė(%) išraiška

Svarbu pastebėti, kad vertinimas procentais yra tikslesnis, nes padeda žmogui įsivaizduoti koks tiksliai gautas rezultatas. Kadangi NPFL išraiška tiesiog suteikia galimybę žinoti kokiuose rėžiuose yra rezultatas. Pavyzdžiui „P“ ar „L“ rėžiai turi po 30% intervalą, tad intervalas išties didelis. Todėl yra tikslingiau gauti rezultatus procentais. Žemiau pateiktoje lentelėje pavaizduoti NPFL išraiškos su atitinkamais procentiniais įvertinimais.[PR13][Pel14]

11 lentelė. NPFL atitiktis procentais

NPFL išraiška	Procentinė išraiška
N –nepasiekiamas	0 – 15%
P – dalinai pasiekiamas	16 – 50%
L – didžiąja dalimi pasiekiamas	51 – 85%
F – pilnai pasiekiamas	86 – 100%

Jeigu buvo atliktas vertinimas pagal TPAM ir gauti įverčiai pagal NPFL, gauti įverčius procentais yra įmanoma. Net nepaisant to, kad procentiniai rėžiai kiekvienam NPFL skalės elementui ganėtinai dideli, yra numatytos trys galimybės, kaip galima pasirinkti ir gauti sau norimą rezultatą. NPLF konvertavimas į procentus gali būti atliktas trimis būdais: pesimistiniu, optimistiniu ir vidutiniu:

Pesimistinis variantas taiko į apatinį rėžį: F->85%, L->50%, P->15%, N->0%

Optimistinis variantas taiko į viršutinį rėžį: F->100%, L->84%, P->49%, N->14%

Vidutinis variantas taiko į aritmetinį vidurkį: N->8%, P->33%, L->67%, F->93%

Naudojantis tarpiniu procesų vertinimo modeliu galima gauti įverčius tarp įvairių procesų vertinimo modelių. Šis modelis padeda gauti įverčius tarp įvairių procesų vertinimo modelių. Bet vertintojas turi būti ekspertas abiejų modelių, kad galėtų viską atlikti kompetentingai ir objektyviai.

3. Formali ontologija ir informacinės sistemos

Ontologija – šis terminas iki 1990 m. buvo naudojamas tik filosofijos mokslų srityje. Jis reiškė – būties teoriją. Pirmą kartą ontologijos sąvoka paminėta siejant su kompiuterių mokslų 1967m.[Mea67]

Kadangi ontologija turi daugybę apibrėžimų, šiame darbe ją apsibrėžiame kaip - aiškią konceptualizacijos specifikaciją, kurios du pagrindiniai aspektai pateikti žemiau:

- Ontologija apibrėžia sąvokas, ryšius ir kitus skirtumus, kurie yra svarbus modeliuojant žinių bazę.
- Specifikacija pateikiama reprezentacinio žodyno (klasių, ryšių ir pan.) formatu, kuris turi tam tikrus suvaržymus tarpusavy.[LÖ08]

Aprašant ontologiją nurodomi rėmai bei sustruktūrizuojamos dalykinės žinios apie naudojamus (realius) reiškinius, įvykius, terminus bei sąvokas. Visų tų terminų, faktų priklausomybę bei tarpusavio sąryšius. Atsirado vis daugiau mokslinių tyrimų skirtų ontologijoms. Su laiku ontologijos įgavo ypatingą reikšmę dirbtiniame intelekte (angl. *Artificial Intelligence*), kompiuterinėje lingvistikoje (angl. *Computational Linguistics*) bei duomenų bazių srityje (angl. *Database Theory*). Pirmasis augimas pastebėtas 1990 m. Ontologija darėsi vis populiaresnė. [Hef00] Plito ir sritys, kuriose ji buvo naudojama. Ontologijos populiarėjimas netgi paskatino susikurti tokią konferenciją kaip FOIS (Formal Ontology and Information Systems). Ontologijos didžiausias populiarėjimas žymimas 2003 – 2005 metais. Kaip pavyzdį galime paminėti, kad nuo pirmosios SWWS (International Semantic Web Working) 2001 m. iki trečiosios konferencijos ICSW (International Semantic Web Conference) 2003 m. konferencijos pranešimų susijusių su ontologijomis, jų panaudojimų bei plėtojimu pakilo beveik 400%. [Miz11]

Su laiku atsirado vis daugiau sričių, kuriuos pripažįsta ontologijos svarbą. Keletas sričių yra pateikta žemiau:

- Žinių inžinerija (angl. *Knowledge engineering)*
- Žinių atvaizdavimas (angl. *Knowledge representation*)
- Kokybės modeliavimas (angl. *Qualitative modelling*)
- Duomenų bazių projektavime (angl. *Database design*)
- Informacijos modeliavimas (angl. *Information modeling*)

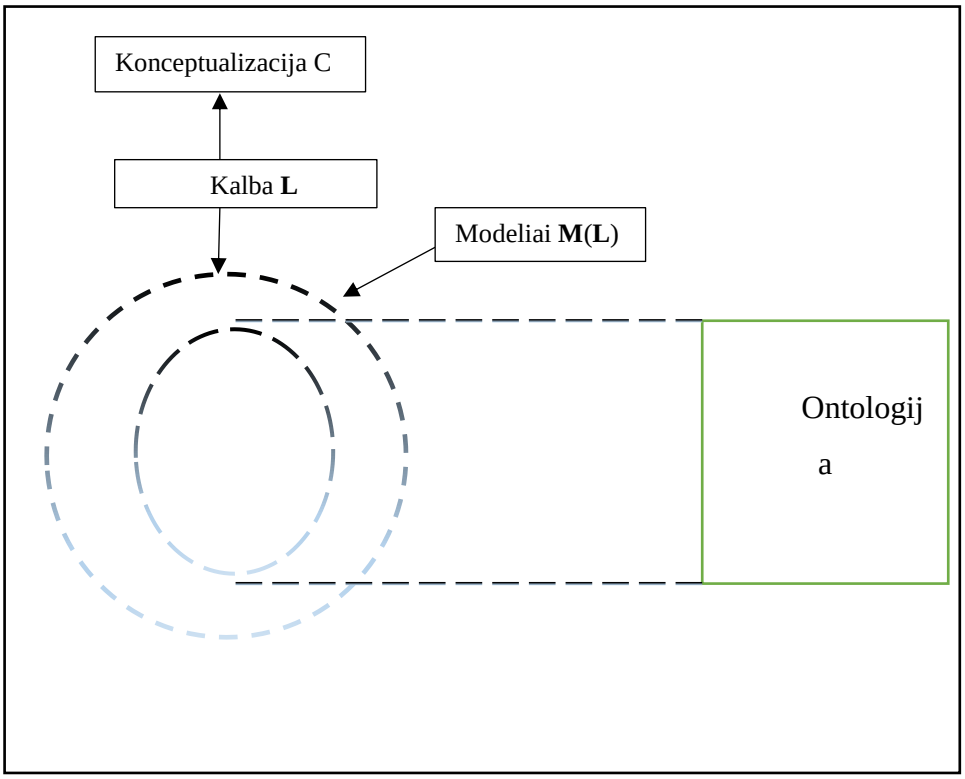
- Informacijos integravimas (angl. *Information integration*)
- Objektinis analizavimas (angl. *Object-oriented analysis*)

Ir daugelis kitų, tokių kaip medicina, mechanika, elektroninė komercija, geografinės sistemos, biologinės sistemos, teisinės sistemos.

Iš metodinės pusės pagrindinis ypatumas yra tarpdisciplininis bei lingvistinis požiūris kuris suformuoja labai griežtą ir tikslią žinių bazę. Iš architektūrinės pusės pagrindinė savybė yra tai, kad ontologija gali užimti informacinės sistemos rolę.[Gua98]

3.1.Kas yra ontologija?

Žemiau pateiktame 4 paveikslėlyje yra pavaizduota kas yra ontologija ir kokį tai turi sąryšį su konceptualizacija ir visais modeliais. Visų pirmą svarbu suprasti, kad ontologija mums tarsi žodynas į kurį galime aprašyti ką tik norime. Bet kokias sritis, bet kokias sąvokas, bet kokius ryšius. Turint tokį žodyną, jam galima pritaikyti tam tikrus modelius. Žemiau pateiktame 4 paveikslėlyje matome, kad yra konceptualizacija „C“, kuri šiame paveikslėlyje atspindi visą kas mus supa – visas sąvokas, visus ryšius. Toms sąvokoms aprašyti naudojame kalbą L. Naudojantis šia ontologija yra aprašytas modelis M. Jis yra aprašytas L kalba, todėl matome žymėjimą M(L).

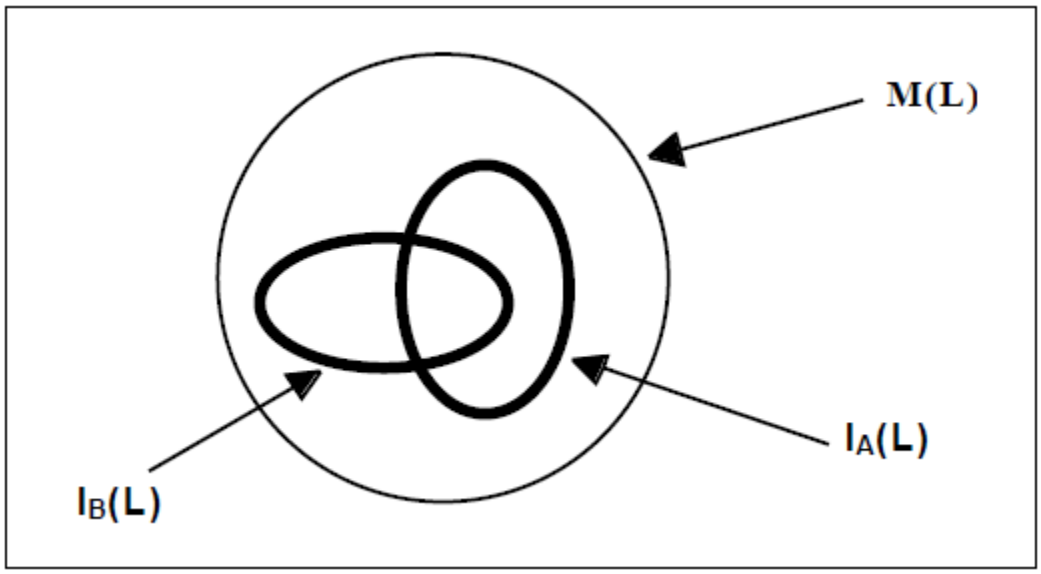


4 pav. Ontologijos samprata konceptualizacijoje

3.2. Ontologijų integravimo problema

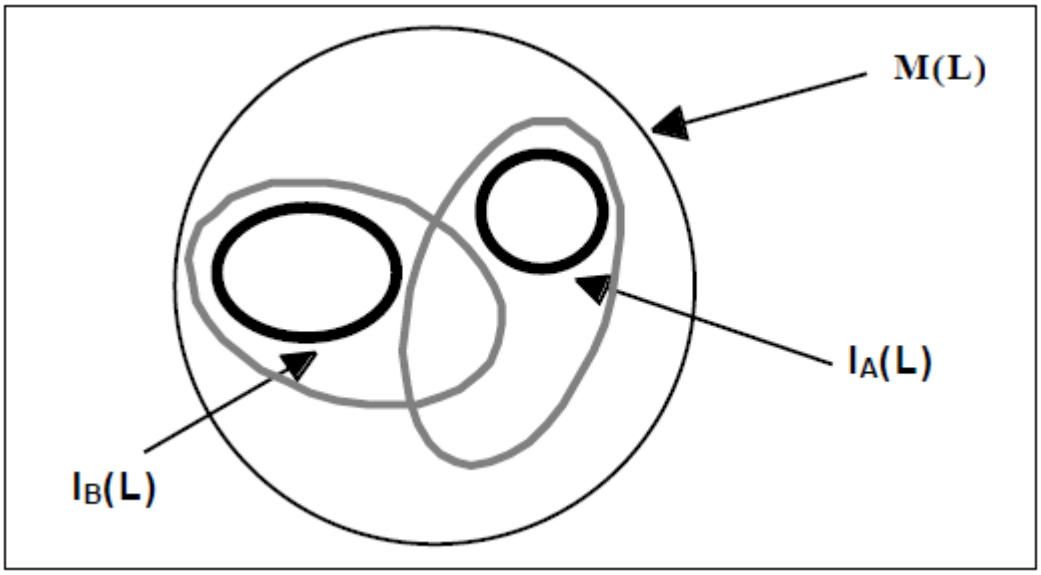
Visų pirmą svarbu paminėti, kad netgi jei dvi sistemos įsisavina tą patį žodyną, tai dar nėra jokių garantijų, kad jų informacija sutampa ir jie bus vienodi, nebent jų konceptualizacija yra ta pati. Turint omeny, kad kiekviena sistema turi savo konceptualizacija. Tad iš čia kyla labai svarbi sąlyga, norint tapatinti dvi sistemas – jų konceptualizacija turi būti vienoda, bent jau iš dalies. [Gua98]

Žemiau pateiktame paveikslėlyje matome dvi sistemas **A** ir **B**, kurios naudoja tą pačią kalbą **L,** gali komunikuoti tik tokiu atveju, jeigu rinkiniai modelių $\mathbf{I}_A(\mathbf{L})$ ir $\mathbf{I}_B(\mathbf{L})$ yra siejami konceptualizacijos, kuri iš dalies sutampa.



5 pav. Modeliai, kurių ontologijos iš dallies sutampa

Įsivaizduokime, kad dabar šie du rinkiniai pritaikytų modelių yra dviejų skirtingų ontologijų. Tad galimas ir toks variantas, kai ontologijos sutampa, o patys modeliai ne. Tai reiškia, kad iš apačios į viršų būdu sistemos integracija iš skirtingų kelių ontologijų gali neveikti. Ypatingai, jeigu tos ontologijos sutelktos tiek koncepciniais ryšiais. Paveikslėlyje, kuris pateiktas žemiau matome, kad ontologijos gali kažkiek sutapti, bet patys modeliai ne. Todėl darant bet kokias integracijas, reikia elgtis išties atsargiai. [Gua98]

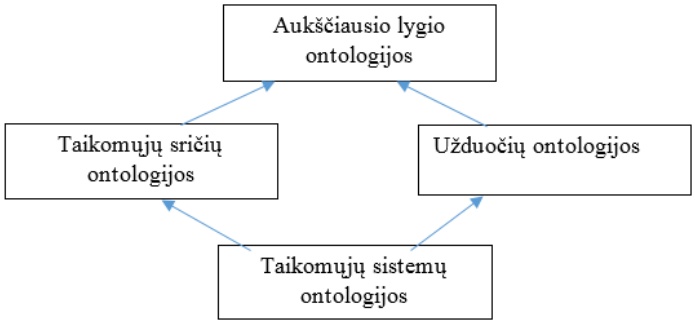


6 pav. Pavyzdys, kai ontologijos dalinai sutampa, bet modeliai nesutampa visai

Analogiškas pavyzdys galėtų būti su ISO ir CMMI-DEV modeliais. Jeigu jie aprašyti naudojantis ontologija, tai iš dalies jie panašūs dėl sąvokų, ryšiu. Bet jie sutampa tik iš dalies ir jų lyginti ir tapatinti mes negalime. Tačiau šie modeliai yra skirtingi.

3.3.Ontologijų skirstymas lygiais

Remiantis „Formal Ontology in Information Systems“ N. Guarino 1998 m. bei autoriaus siūlymu galėtume išskirti ontologiją į kelis lygius pagal jų priklausomybę nuo užduoties. Kadangi ontologija – sutartų faktų, žodynų, reiškinių aprašymas, dalinimasis bei daug kartinis panaudojimas, todėl ir skirstymas susidaro pagal šiuos komponentus. Yra pagrindinė žinių bazė, kuri yra skirstoma į bendrą žinių bazę, kuri apibrėžia bendrinius atvejus, reiškinius bei žodynus, ir į daugiau specifinę žinių bazę, kuri yra pritaikyta konkrečiai situacijai, konkrečiam atvejui. „... ontologiją (sauganti situacijas – nepriklausomas nuo konkrečios informacijos) ir „Esminę“ žinių bazę (sauganti informaciją – priklausoma nuo konkrečios informacijos).“ [Gui05]



7 pav. Ontologijų skirstymas lygiais

Aukščiausio lygio ontologija – šiame lygyje yra aprašomos pačios pagrindinės sąvokos, tokios kaip – objektas, įvykis, veiksmas, esybė, materija, laikas, erdvė. Šio lygio ontologijos yra nepriklausomos nuo konkrečios problemos ar žinių bazės. Jos labiausia abstrakčios.

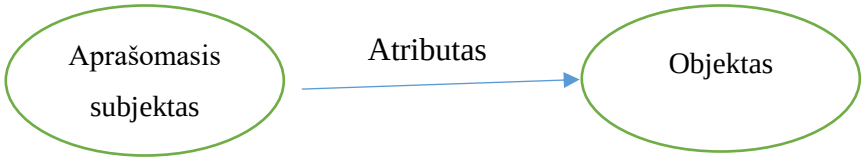
Taikomųjų sričių bei **užduočių ontologijos** – šiame lygyje aprašomi atitinkami žodynai, pritaikyti konkrečiai sričiai (pvz. teisei, medicinai, automobiliams) arba pagrindinės užduotys ir veiksmai (pvz. medicina – sveikatos apžiūra. Automobiliai – techninė apžiūra), kurie yra aprašyti aukščiausiame lygyje.

Taikomųjų sistemų ontologijos – aprašo pagrindines sąvokas kurios priklauso tiek taikomųjų sričių tiek užduočių ontologijoms. Šiame lygyje aprašomos sąvokos dažnai atitinka roles, kurias atlieka taikomojoje ontologijoje aprašytos esybės.

3.4.Picos ontologija

Picos ontologijos apžvalgoje, yra labai nuosekliai paaiškinta apie pačią ontologija. Kas tai yra, kaip ją galime panaudoti. Šis straipsnis padeda geriau suprasti ontologijos paskirtį, kaip ji gali pasitarnauti norint įgyvendinti mūsų išsikeltus tikslus. Šiame skyriuje autorius mėgins vaizdžiai apibūdinti ontologijas, kad būtų galima lengviau susidaryti bendrą įspūdį.[Hor11]

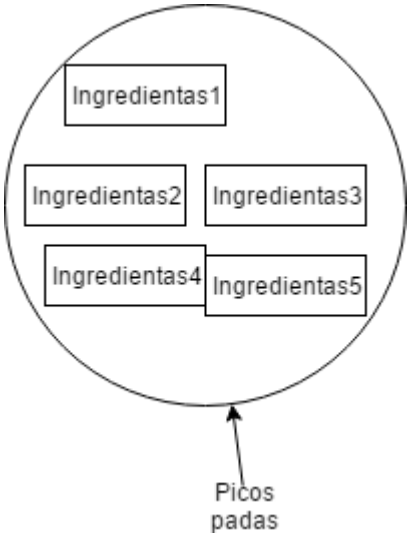
Visų pirmą svarbu paminėti, kad kuriant OWL ontologiją viską reikia apibrėžti trilypiu ryšiu:



8 pav. Trilypis ryšys

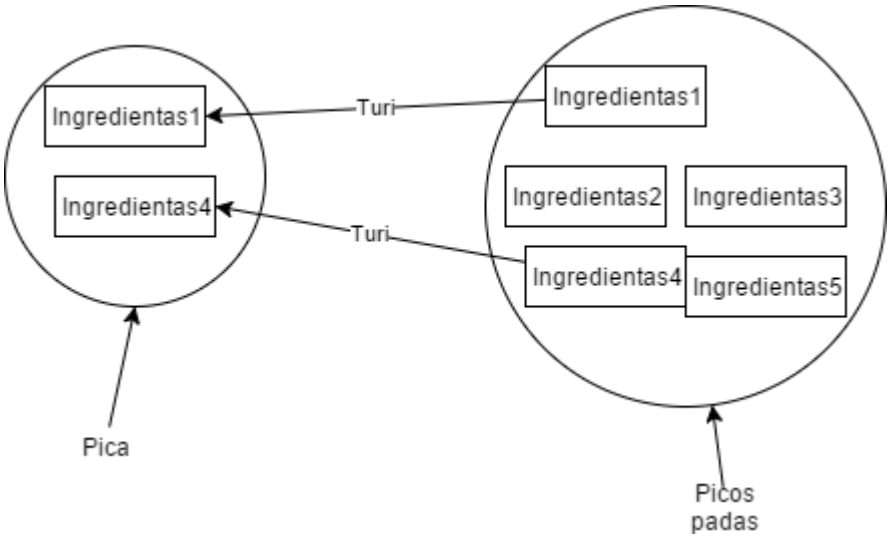
Šiame darbe buvo aprašoma picos ontologija. Įsivaizduokime, kad buvo aprašyti trilypiai ryšiai, tokie kaip **Pica-Turi-PicosPadą**. Žemiau paveikslėliuose yra pateikta ganėtinai abstraktūs, bet tuo pačiu ir informatyvūs paveikslėliai, kurie paaiškina kelis esminius dalykus:

- **Picos padas** susideda iš ingredientų, tam tikrų esybių. Pavyzdžiui tai galėtų būti: miltai.



9 pav. Picos pado sudėtis

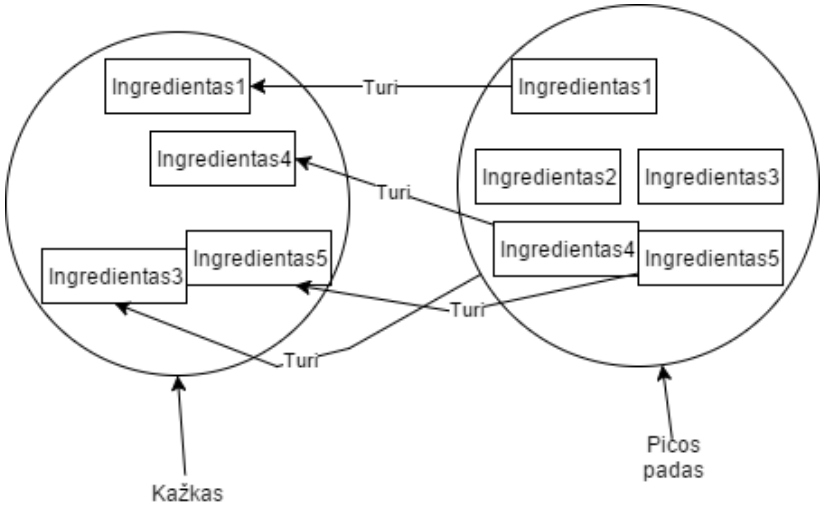
- **Pica** yra poklasė **kažko**, kas turi bent vieną **picos pado** ingredientą (sudedamąją dalį), bet neturi pakankamai kitų ingredientų, kad būtų pica. Todėl pica atrodo taip:



10 pav. Ryšys tarp picos ir Picos pado

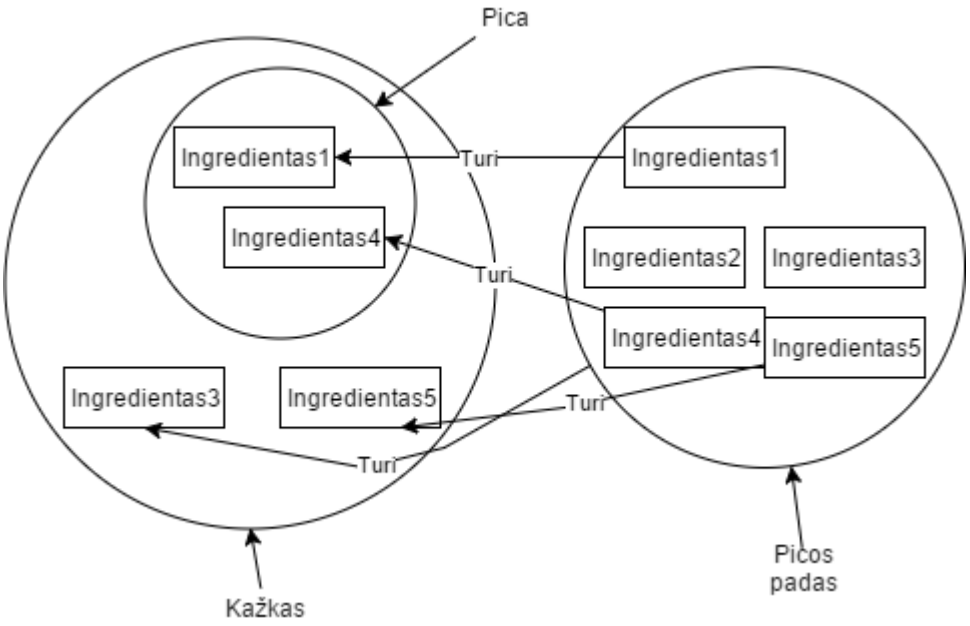
- **Kažkas** yra ganėtinai abstrakti esybė, kuri turi tam tikrų picos pado ingredientų, bet tik tiek. Ši esybė yra kažkokios esybės poklasė, taipogi, bet šiam pavyzdžiui mes labai

nesigilinsime. Taigi kažkas irgi turi tam tikrų picos pado ingredientų. Ši esybė diagramoje atrodo visai panašiai kaip ir **pica**.



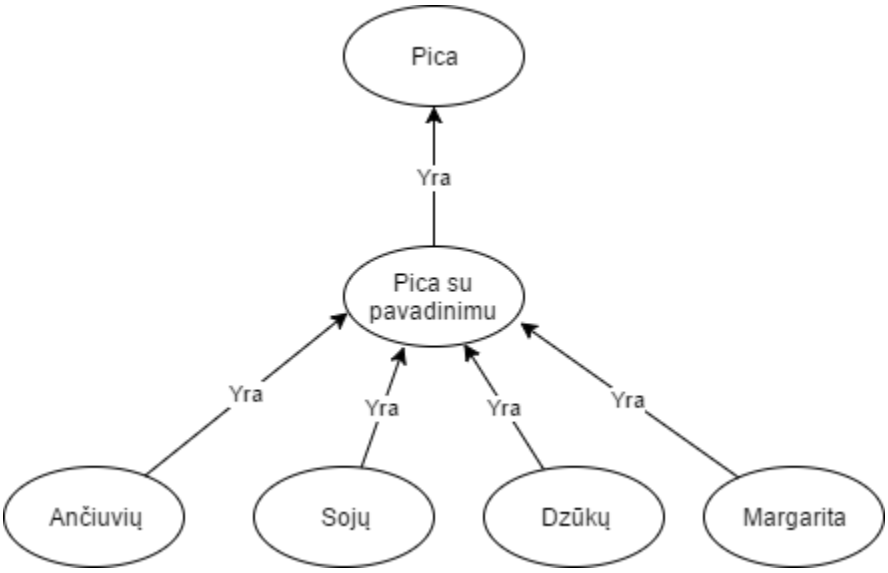
11 pav. Ryšys tarp Picos pado ir neapibrėžtos esybės

- Visas tris diagramas apjungus gautume diagramą, kurioje būtų matyti, kad tiek **pica**, tiek **kažkas**, turi kažkokių **picos** pado ingredientų (taip pat, tai priklauso nuo pačios picos. Picos padai irgi būna įvairiausi. Galėtume juos vėlgi išskirti pagal tai kokius ingredientus jie turi.). Pateiktame pavyzdyje **kažkas** turi ingredientus, kurie pažymėti 1,3,4,5 numeriais. O pica turi tik 1 ir 4, todėl ji ir yra **kažko** poklasė. **Kažkas** šiuo atveju galėtų būti tiesiog **tešla iš miltų**.



12 pav. Pavyzdys, kaip Pica priklauso kažko esybei ir turi picos pado ingredientų

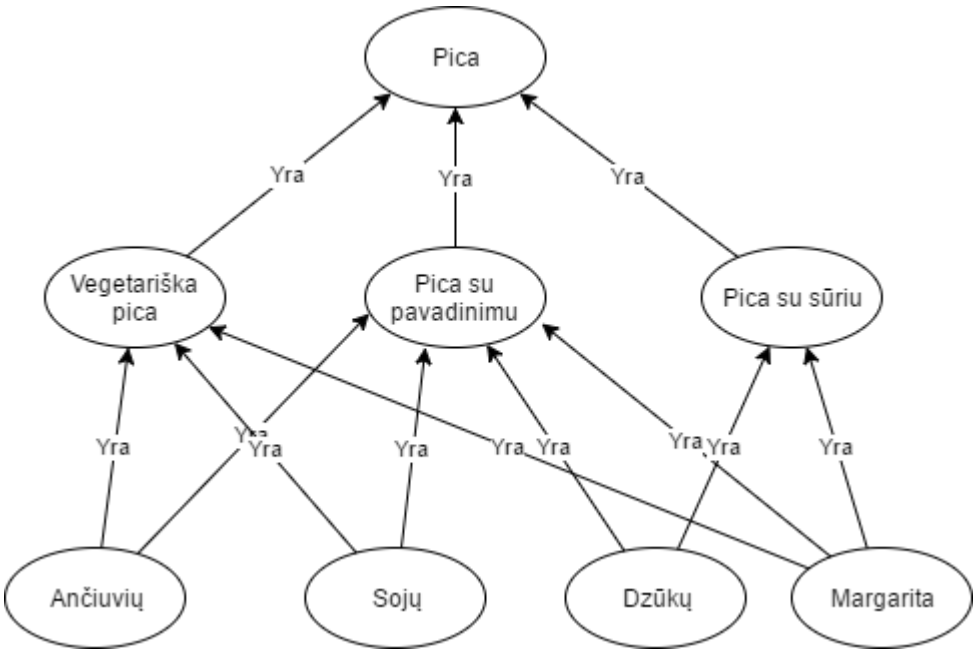
Toliau pamėginkime įsigilinti į pačią picą. Juk picos turi pavadinimus, tad išskirkime tokią klasę, kaip **pica su pavadinimu**. Ši klasė irgi turi esybių, kurios atitinką ją. šiuo atveju, **Dzūkų** pica yra **pica su pavadinimu**, kas reiškia, kad **Dzūkų** pica yra ne tik **pica su pavadinimu**, bet ir **pica**.



13 pav. Picą bei jos poklasės

Paprasčiausiai ištarus sau mintyse ar garsiai, kad pica, kuri turi pavadinimą yra pica – atrodys akivaizdu, kad net nereikia to sakyti. Tad žemiau autorius pateikia būtent tokį pavyzdį:

Taip pat yra svarbu parodyti, kaip viena esybė gali priklausyti ir atitikti kelias kitas esybes. Juk jų gali būti galybė, jos gali turėti pavadinimus, būti sudarytos iš daugybės įvairiausių ingredientų. Taip pat jos gali būti priskiriamos įvairiausioms kategorijoms. Juk tuo pačiu metu pica gali būti ir tarp vegetariškų, ir tarp picų su sūriu kategorijų. Žemiau esančiame paveikslėlyje autorius pateikia picos hierarchiją, kad būtų galima lengviau įsivaizduoti.



14 pav. Picų priklausomybė kelioms esybėms

Paanalizuokime iš šio pavyzdžio picą „Margarita“:

- **Margarita** turi sūrio savo sudėtyje, todėl ji priskiriama **Pica su sūriu** kategorijai.
- **Margarita** turi pavadinimą (Margarita), todėl ji priskiriama **Pica su pavadinimu** kategorijai.
- **Margarita** yra vegetariška pica, todėl ji priskiriama **Vegetariška pica** kategorijai.

Nauda iš tokių ontologijų:

Pasirašę tokią ontologiją, galėtume iš jos ištraukti labai daug informacijos (reikėtų naudoti SPARQL užklausas).

- Galėtume surasti visas picas, kurios savo Sudėtyje turi sūrio.
- Galėtume atlikti sudėtingesnes paieškas, pavyzdžiui susirasti picas, kurios turi mūsų norimus ingredientus ir neturi tokių, kurių nemėgstame. Taip pat kartu pažiūrėti, kad būtų priskirtos kažkokiai kategorijai, pavyzdžiui, vegetariškoms picoms. Pavyzdys galėtų atrodyti taip – vegetariška pica, kuri sudėtyje turi sūrio, lašišos, bet neturi svogūnų ir alyvuogių.
- Ontologija suteikia galimybę mums lyginti picas. Galėtume pasižiūrėti kiek % **Margarita** pica atitinka **Dzūkų** picą. Būtų lyginama pagal ingredientus. Galėtume matyti, kad jos pavyzdžiui sutampa 90%.

3.5. Praktikoje įgyvendintos ontologijos

Šiame skyriuje aprašytos įvairios ontologijos, kurie turėjo įvairių tikslų, nuo modelių suderinimo iki lengvesnio modelių skaitymo.

3.5.1. H2mO harmonizacijos ontologija

Autoriai Cesar Pardo, Francisco J.Pino, Felix Garcia, Mario Piattini, Maria Teresa Baldassarre atliko šį mokslinį tyrimą spręsdami modelių nesuderinamumo problemą.

Analizuodami esamą situaciją rinkoje jie išskyrė kaip pagrindinę problemą – žodyną. Jie išskyrė keletą pagrindinių problemų susijusių su žodynu:

- Kai kurie tyrimai skirti ištirti ryšius tarp skirtingų modelių naudoja tokias sąvokas kaip sinergija (angl. *synergy*) arba suderinamumas (angl. *compatability*). Tačiau nenurodo jokios skalės, kuri yra naudojama palyginimams. Todėl nėra aišku kiek jie skirtingi ir kiek jie panašūs iš tikro.
- Taip pat tokie žodžiai kaip jungti (angl. *combining*) bei sulieti (angl. *merge*) yra naudojami kalbant apie kelis skirtingus modelius, tačiau nėra nurodoma skirtumas tarp šių sąvokų, kai kalbame apie tuos žingsnelius, kai norime integruoti du modelius iškart. Skirtumas yra, bet jis nėra apibrėžtas.
- Šiek tiek geresnė situacija, tačiau vis dar opi problema su žodžiais palyginimas (angl. *comparison*) ir kartografavimas (angl. *mapping*). Pavyzdžiui, norint identifikuoti panašius elementus tarp dviejų modelių, mes naudosime kaip tik šias sąvokas, tačiau svarbu suvokti, kad jos turi skirtingas reikšmes bei yra naudojamos visiškai skirtingais atvejais. Nedaug žmonių žino kada kurią sąvoką naudoti ir leidžia sau tai daryti pagal nuožiūrą.
- Visiškai identišką reikšmę - turi sąvokos integracija (angl. *integration*) bei susivienijimas (angl. *unification*). Vieni projektai, pavyzdžiui, „PrIME“ naudoja integraciją, o „Yoo, et al.“ Naudoja „unifide“, kas reiškia – vieninga.
- Nėra jokio standarto, kaip turėtų būti aprašyta ir kas tiksliai yra sąvoka ar tokie modeliai kaip proceso modelis, technikos modelis.

Išskyrę šias problemas straipsnio autoriai priėjo prie išvados, kad kažkokio pobūdžio terminų bei veiksmų harmonizacija yra būtina rinkoje. Jie sukūrė harmonizuotą kelių modelių ontologiją – H2mO (angl. Harmonization of Multiple Models Ontology), kuriai iškėlė du pagrindinius tikslus:

1. Terminų, įvairių sinonimų, homonimų, neatitikimų, nesuderinamumų bei terminologinių konfliktų nustatymo bei identifikavimo.
2. Įgyvendinti tokias sąvokas, kurios naudojamos išanalizuotojo literatūroje.

Šie mokslininkai nutarė, kad norint pasiekti šiuos tikslus reikės naudoti ontologiją, kuri reprezentuotų harmonizaciją tarp kelių modelių. Ši ontologija privalo apibrėžti visas sąvokas, kurios yra aiškios ir suprantamos, nustatyti terminus bei visus apibrėžimus, kurie padėtų nustatyti ryšius tarp modelių. Savo darbe didžiąją dalį informacijos, tokios kaip apibrėžimai ar ryšiai jie ėmė ir rėmėsi kitomis ontologijomis bei kitais moksliniais darbais, kol galiausiai sudarė lentelę terminų bei jų paaiškinimų, kad būtų aiškiai suprantama konkretaus termino reikšmė ar veiksmo paskirtis.

12 lentelė. H2mO terminai

Terminas	**Tipas**	**Apibrėžimas**
Palyginimas (angl. *Comparison*)	Metodika	Aukšto lygio charakteristikų analizė tarp skirtingų modelių.
Apimties laipsnis (angl. *Degree of coverage*)	Matavimo vienetas	Apimties laipsnis rodo kiek du skirtingi modeliais yra panašūs atsižvelgiant į tam tikras charakteristikas.
Santykių laipsnis (angl. *Degree of relationship*)	Matavimo vienetas	Santykių laipsnis nurodo kiek konkreti esybė yra panaši į kitą esybę iš kito modelio, matavimas paremtas „vienas su vienu“ tipu.
Detalumas (angl. *Granularity*)	Sąvoka	Sąvoka skirta suprasti, kiek modelis yra detalus.
Harmonizacija (angl. *harmonization*)	Sąvoka	Rinkinys žingsnių, kurie turi būti atlikti vienam ar keliems modeliams siekiant pakeisti jų procesų struktūrą į bendresnę.
Harmonizacijos strategija (angl. *Harmonization strategy*)	Sąvoka	Harmonizacijos strategija yra procesas, kuris leidžia mums sistemingai judėti link vieno ar kelių modelių subendrinimo ir tiksliai nurodo eigą, ką po ko daryti.
Homogenizavimas (angl. *Homogenization*)	Metodika	Žingsnis ar rinkinys žingsnių, kurie yra reikalingi perdaryti procesų struktūrą į homogenizuotą.
Homogenizavimo struktūra (angl. *Homogenization structure*)	Sąvoka	Homogenizuotos struktūros procesai bei jų elementai palengvina kelių modelių analizę, kurie yra įgyvendinti pagal skirtingus modelius (CMMI, ISO/IEC 15504), bet pagal tą pačią struktūrą. To
Integracija (angl. *Integeration*)	Metodika	Veiksmas arba poveiksmis, siekiant sujungti du ar daugiau modelių įgyvendinant reikalingas metodikas bei veiksmus.

Kartografavimas (angl. *Mapping*)	Palyginimas	Palyginimo metodikos įgyvendinimas ir gilinimasis į rezultatus ne tik paviršutiniškai.
Matavimas (angl. *Measurement*)	Sąvoka	Apibrėžia matavimo kriterijus bei matavimo skalę.
Metodas (angl. *Method*)	Sąvoka	Metodas yra procedūra, kuri įprastai yra orientuoja į tam tikrą tikslą.
Kokybės modelis (angl. *Quality model*)	Sąvoka	Rinkinys išmatuojamų sąvokų bei ryšių tarp jų, nurodantis kokybės reikalavimus bei kokybės gerinimo principus.
Ryšys (angl. *Relationship*)	Sąvoka	Ryšys tarp nurodytų tam tikrų procesų ar net modelių.
Skalė (angl. *Scale*)	Sąvoka	Rinkinys kintamųjų su nurodytomis reikšmėmis.
Semantinė analizė (angl. *Semantic analysis*)	Analizės terminologija	Procesas, kuriuo analizuojami tam tikri veiksmai atsižvelgiant į įprastą kalbą, kad būtų paprastai suprantama žmonėms.
Sinergija (angl. *Synergy*)	Sąvoka	Sinergija įvyksta, kai būna sujungiami daugiau nei du modeliai į žymiai didesnę visumą. Sinergija naudojama, kai toks sujungimas padaro modelius, kurie yra jungiami kaip įmanoma kokybiškesniais.
Sintaksinė analizė (angl. Syntactic analysis)	Analizės terminologija	Procesas, kurio metu turimas tekstas patikrinamas gramatiškai, kad nebūtų gramatinių klaidų ir atitiktų deramą struktūrą.
Metodika (angl. *technique)*	Sąvoka	Skirtingi būdai įgyvendinti metodą.
Terminologijos analizė (angl. *Terminology analysis*)	Sąvoka	Terminologinė analizė teksto analizavimo procesas, kuris leidžia geriau suprasti modelius, jų sintaksę taip pat semantinę analizę.
Skalės tipas (angl. *Type of scale*)	Sąvoka	Natūralus ryšys tarp reikšmių skalėje.
Matavimo vienetas (angl. *Unito f measurement*)	Sąvoka	Tam tikras matas, kuris yra nustatytas pagal susitarimą.

Taip pat H2mO turi nusistatę savo ryšių lentelę, kurią taip pat pristatė visiems kartu su terminų bei metodikų lentele. Nepaisant to, kad pagrindinės sąvokos apibrėžtos lentelėje, visgi liko septynios labiausiai diskutuotinos sąvokos bei jų naudojimas buityje: harmonizacija, palyginimas, sinergija, korespondencija, integracija, subendrinimas bei kartografavimas. Nesunku pastebėti, kad nurodytame sąraše yra tų pačių sąvokų, kurios parašytos ir lentelėje.

H2mO ontologija buvo išmėginta trimis scenarijais. Pirmieji du buvo pritaikyti organizacijoms, kurios domisi bei turi pakankamai žinių apie kelių modelių suderinimą vienoje organizacijoje.

1. Pirmuoju atveju harmonizacija buvo pritaikyta tyrėjų, kurie siekė sukurti modelį Informacinių Technologijų Vyriausybei – ITG (angl. *Information Technology Government*). Kuri siekė pritaikyti tą modelį Gvatemalos bankams valdyti bei prižiūrėti. Šiuo atveju buvo pritaikyta 6 modelių harmonizacija, kurios rezultatas buvo modelis pavadintas Informacinių Technologijų Taikymo modelis bankams – ITGMSB (angl. *Information Technology Governance Model for Banking*). Šis bandymas buvo sėkmingas ir harmonizacija pavyko sklandžiai.
2. Antruoju atveju harmonizacija buvo įgyvendinta *Audisec* organizacijoje. Šioje organizacijoje teko įgyvendinti du modelius „ISO 27001“ ir „ISO 20000 part 2“. Harmonizacija susitelkė ties tokių problemų sprendimu:
    - Modelių palyginimu, skirtumų bei panašumų identifikavimu
    - Papildomumo lygio įvertinimas (Išsiaiškinimas kiek modeliai papildo vienas kitą)
    - Rūpintis bei konsultuoti organizacijas dėl ISO 20000 atsižvelgiant į ISO 27001

    Viena svarbiausių išmoktų pamokų buvo tinkamai valdyti harmonizaciją, kuri įtraukia kelis skirtingus modelius, skirtingus terminus bei sąvokas. Taip ryšiai tarp jų turi išlikti aiškūs. Harmonizacijos ontologija buvo labai naudinga ir pripažinta. Po tyrimo buvo pripažinta, kad ji labai padėjo suprantant skirtingų modelių procesus.
3. Paskutinysis atvejis buvo pritaikytas kuriant „HProcessTOOL“ žiniatinklio įrankį, kuris palengvina valdymą harmonizacijos procesų. Kuriant šį įrankį buvo naudojami H2mO terminai bei ryšiai. Tiesiog mėginta H2mO praplėsti ir atsirinkus tinkamiausius aspektus sukurti kitą įrankį.

Privalumai bei trūkumai:

Išanalizavus harmonizacijos ontologiją H2mO galima išskirti šiuos privalumus bei trūkumus:

Privalumai:

- Suvienodintas plačiausiai naudojamų terminų bei ryšių žodynas.
- Patogu vertinti modelius, kai jie kuriami remiantis tuo pačiu „karkasu“.
- Išmėginta realiai trimis skirtingais atvejais, kurie davė teigiamus rezultatus.

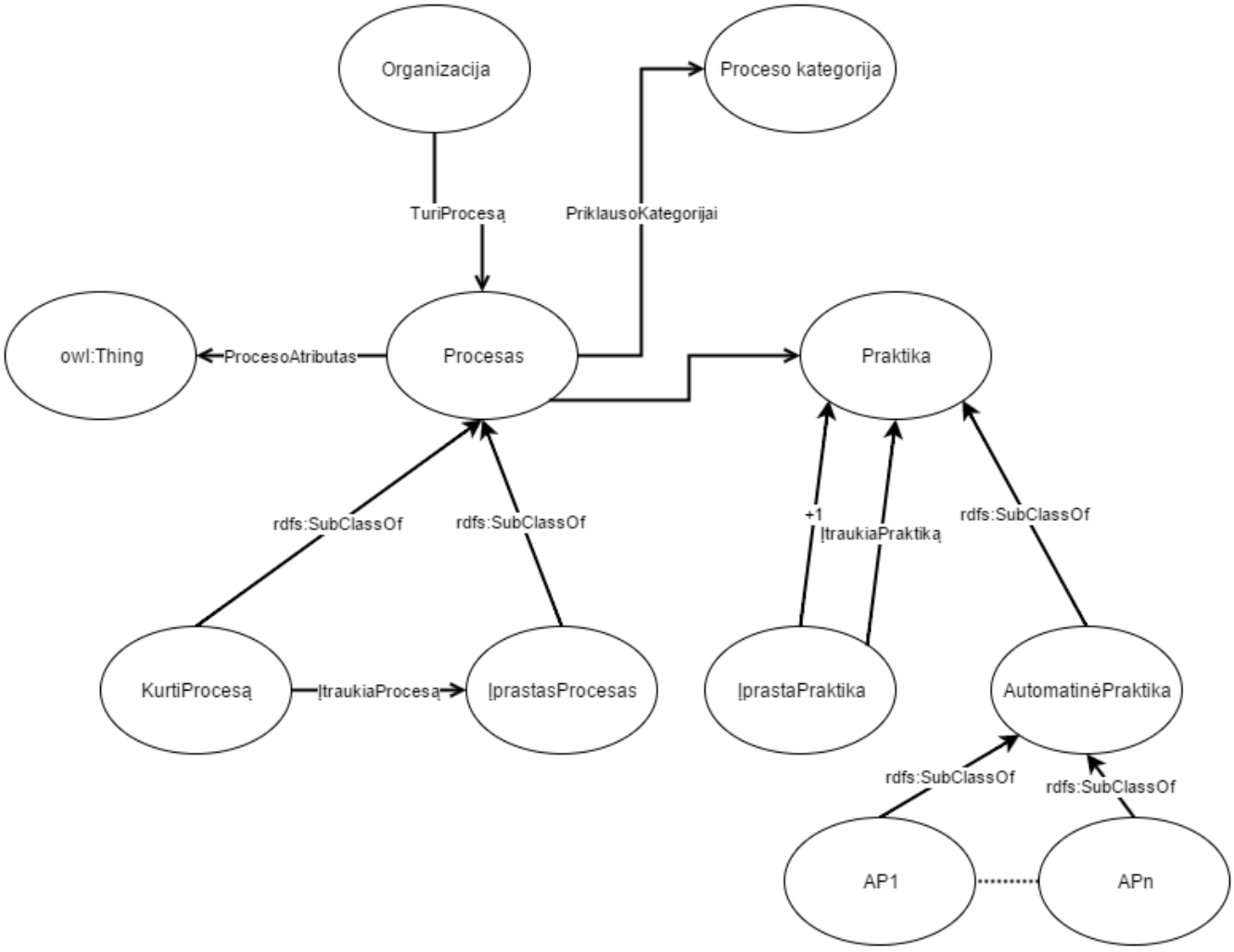
Trūkumai:

- Harmonizacijos ontologija sukurta tik žemame lygyje, su pačiomis abstrakčiausiomis sąvokomis.

3.5.2. Programinės įrangos procesų ontologija

Programinės įrangos procesų ontologija, žinoma kaip SPO (angl. *Software Process Ontology*) buvo sukurta autorių Li Liao, Yuzhong Qu bei Hareton K. N. Leung. Ontologija buvo kuriama remiantis CMMI bei ISO/IEC 15504, tačiau suprojektuota taip, kad būtų galima pritaikyti ir bet kuriam kitam modeliui. Idėja kurti SPO kilo iš tų pačių problemų, kurios kyla iš skirtingų modelių architektūrų, turinio, požiūrio į tam tikras situacijas, sąvokų, taip pat netgi tikslo. Šiek tiek daugiau nei prieš 10 metų pradedant kurti ontologijas orientuotas į organizacijų procesų gerinimo bei vertinimo modelius buvo itin retas atvejis kombinuoti semantinius tinklus bei programų sistemų inžineriją.

Pasitelkus OWL ontologiją buvo suprojektuotas bei sukurtas SPO. Tuomet paraleliai SPO buvo praplėstas, kad atitiktų CMMI bei ISO/IEC 15504, po to buvo sukurtas organizacijų procesų vertinimo prototipas.



15 pav. SPO žemiausio lygio grafai

Diagramoje pavaizduoti SPO grafai parodo, kad viskas vėlgi yra tik pačiame žemiausiame lygyje. Dėl to galima kurti ontologijas, bet jos bus labai abstrakčios ir konkrečių rezultatų pasiekti nepavyks. Pavyzdžiui Procesas yra klasė, kuri reprezentuoja super klasę visų tipų procesams modelyje. Šiuo atveju turi dvi klases „po savimi“ – „KurtiProcesą“ bei „ĮprastasProcesas“. Tokiu būdų galime atvaizduoti tiek CMMI tiek ISO/IEC 15504 ar bet kokio kito modelio bet kokio proceso sritį, tai būtų specifiniai tikslai, ar bendriniai. Remiantis tokiu principu buvo sudarytos CMMI_Onto bei ISO15504_Onto ontologijos.

Naudojantis SPO galima kurti ontologijas kiekvienam modeliui atskirai pagal tą patį karkasą, tačiau tai bus labai abstraktu ir netikslu. Taip pat SPO nepakankamai gerai išsprendžia kelių modelių įgyvendinimo organizacijoje problemą.

Privalumai bei trūkumai

Labiau įsigilinus į SPO išryškėja teigiami bei neigiami šio modelio bruožai.

Privalumai:

- Nėra labai sudėtinga iš SPO sukurti atitinkamai kiekvienam modeliui ontologiją
- OWL ontologijų įtraukimas į šią sritį yra didelis žingsnis į priekį

Trūkimai:

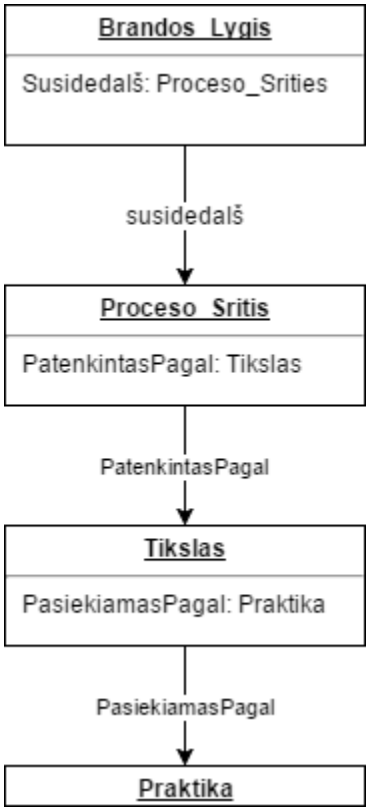
- Ontologijos gausis labai abstrakčios
- Tai nepakankamai išsprendžia problemą dėl kelių modelių palyginimo bei įgyvendinimo vienu metu
- Viskas yra žemiausiame lygyje
- Nėra pakankamai medžiagos kaip tikslingai kurti modelio ontologiją, norint po to juos palyginti

3.5.3. CMMI-DEV ontologija

CMMI-DEV ontologiją sugalvoję kurti autoriai, Gokhan Hait Soydan bei Mieczyslaw M. Kokar turėjo aiškų tikslą – optimizuoti CMMI-DEV įvertinimą įmonėms. Pagrindinė jų idėją buvo sukurti ontologiją, pagal kurią organizacijos galėtų apsirašyti savo viduje (organizacijose) vykdomus procesus ir taip gauti įvertinimą pagal CMMI-DEV. Tokiu būdų organizacija žinot savo brandos lygį. Jie taip pat nurodė galimus ir kitokius šios ontologijos panaudojimus, tokius kaip, procesų gerinimas,

procesų optimizavimas ir vienas svarbiausių, kad ontologija gali būti naudojama informacijos perdavimui apie tam tikrus procesus tarp skirtingų organizacijų.[SK12]

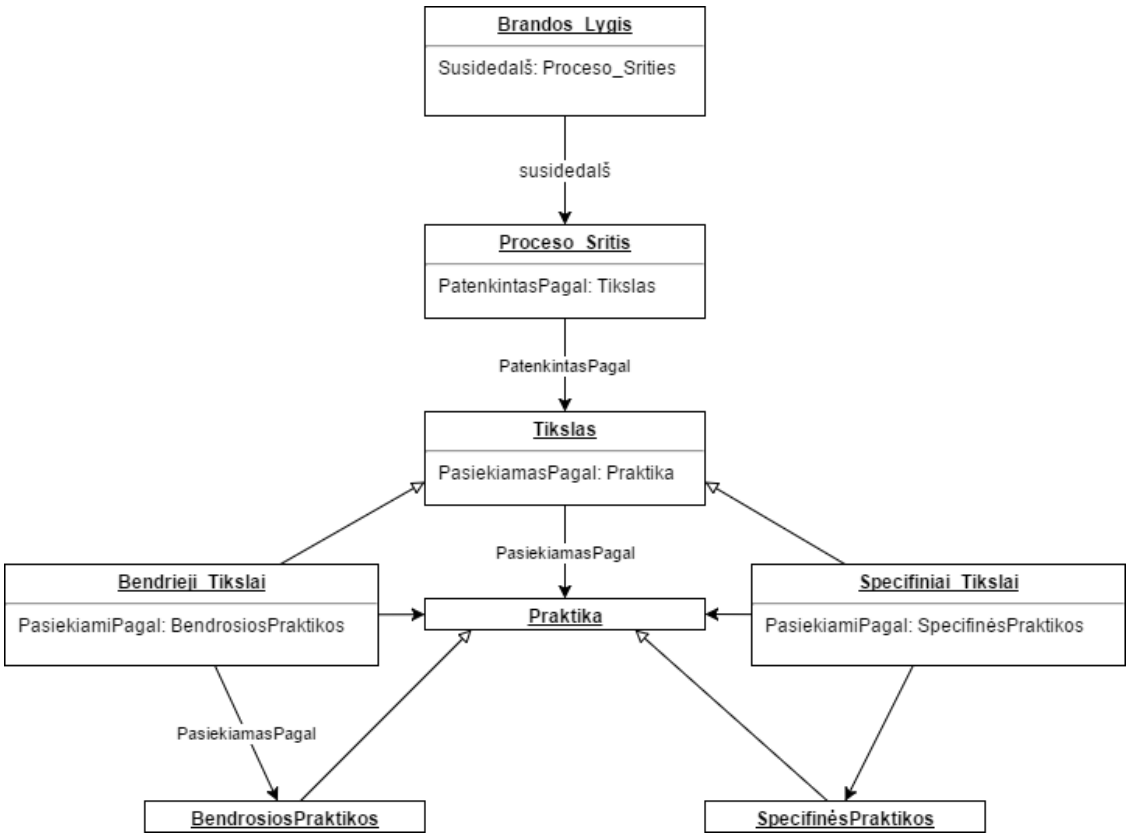
Autoriai šiame darbe vėlgi išskyrė svarbiausią problemą ir užduotį – aprašyti visas sąvokas vieningai. Žemiau pateikta 16 paveikslėlyje diagrama, kurioje OWL užrašyta ontologijos fragmentas, kuriame parodytas kelias nuo praktikos iki brandos lygio.



16 pav. CMMI-DEV Brandos lygio poklasės

Iš paveikslėlio galime suprasti pačius esminius dalykus, kurių mums reikia, kad aukščiausias lygis (Brandos) susideda iš proceso sričių, o proceso sritys yra įgyvendinamos įvykdžius tikslus. Pagal tai galima suprasti, kad ir tikslai yra įgyvendinami dėka praktikų.

Skaidant CMMI-DEV modelį galima leisti išties giliai. Autoriai savo darbe taip pat nesustojo taip abstrakčiai ir tęsė toliau. Tad pateiktame 17 paveikslėlyje žemiau galima matyti tai, kas visiškai atitinka CMMI-DEV modelį.



17 pav. CMMI-DEV ontologijos fragmentas

Kaip autoriai nurodo, tai yra 2 detalumo lygis. Kaip jau buvo minėta – autorių pristatytos diagramos visiškai atitinka CMMI-DEV. Esybė Tikslas susideda iš kitų esybių, tokių kaip, Bendrieji tikslai, bei Specifiniai tikslai. Taip buvo skaidomas visas CMMI-DEV modelis.

Toliau autoriai dirbo prie brandos lygių. Jie apibrėžė kiekvieną lygį, iš ko jis susideda, pavyzdžiui buvo nurodyta, kad 2 Brandos lygis susideda iš 1 Brandos lygio, bei tokių praktikų kaip:

- Proceso ir produkto kokybės užtikrinimas
- Projekto stebėjimas ir kontrolė
- Projekto planavimas

Sukūrę šią ontologiją jie atliko testavimus su 15 organizacijų, o testų rezultatai atitiko autorių lūkesčius.

Ši, CMMI-DEV, ontologija dar vienas darbas, kuomet ontologijų dėka bandoma dirbti su organizacijų procesų vertinimu. Vieni dirba su vertinimų atvaizdavimu, kiti mėgina automatizuoti patį įvertinimą. Padaryti jį pigesnį ir prieinama didesniam spektrui žmonių, tačiau visus tuos žmonės, kurie

prie to dirba, jungia ontologijos. Autoriai tai mato kaip neatsiejama to dalį ir matydami tame perspektyvą bei potencialą atlieka įvairiausius eksperimentus.

3.6. Ontologijų apibendrinimas

Ontologijos yra vis plačiau naudojamos, jos yra ateitis. Jos daugelyje sričių jau dabar naudojamos. Naudojant ontologijas informacinėse sistemose svarbu suprasti, kad jos turi turėti labai griežtą tvarką, Taip pat svarbus dalykas, kad ontologijas galima jungti, lyginti. Kad yra ir toks galimas variantas, kad modeliai skiriasi ir visiškai nepersikloja, o ontologijos atitinkamai persikloti gali.

4. OWL ontologijos

OWL kalba ir technologija yra paremtas ir semantinis tinklas. Kurio idėja, kad visa medžiaga internete būtų struktūrizuota, parašyta pagal tam tikras taisykles ir tai padėtų dirbtiniam intelektui apdoroti visą tą informaciją. Didžiausia internetinė enciklopedija – Wikipedia, taip pat naudojasi griežta struktūra, kad aprašytų duomenis, kuriuos po to būtų galima greitai rasti atliekant paieškas populiariausiose paieškos sistemose – google, bingo, yahoo. Kol kas toks struktūrizuotas rašymas nėra labai populiarus ir populiarėja ganėtinai lėtai. Tačiau, jeigu visi produktai, paslaugos ar šiaip informacija būtų aprašoma struktūrizuotai, tai ieškant būtų galima rasti kur kas tikslesnius duomenis.

Kiekvieną OWL žodyną sudaro tam tikros jo dalys, kurios yra žemiau aprašytos [FHH+12]. Siekiant sustiprinti supratimą, kai kur pateikti pavyzdžiai iš picos ontologijos.

Klasės – apibrėžia klasę individų su panašiomis charakteristikomis, tai gali būti tiek kitos klasės – poklasės, tiek individai su tam tikromis savybėmis. Pati aukščiausia klasė yra pavadinta „daiktu“ owl:Thing. Pačios abstrakčiausios sąvokos turėtų būti apibrėžiamos kaip klasės. Pavyzdžiui picos ontologijoje klasė yra „Pica“, o jos viduje yra įvairių klasių, pavyzdžiui „Vegetariška pica“, o vegetariškos picos klasėje jau yra konkretūs duomenys.

Individai – dar vadinami „faktais“ yra klasių nariai, jau konkrečios reikšmės. Pavyzdžiui picos ontologijoje galėtume paimti pavyzdį kaip Vegetariškos picos klasės individas – špinatų pica.

Duomenų tipų savybės – tai viena iš dviejų pagrindinių savybių kategorijų. Jos paskirti sujungti individus su konkrečiais duomenimis, kurie apibrėžiami kaip kintamieji. Paprasčiau sakant, kai jungiami individą su reikšme, kuri yra loginė, tekstinė, skaitinė – naudojame duomenų tipo savybę. Picos ontologijoje mes jau turime individą – špinatų picą. Jos duomenų tipo savybė galėtų būti „ilgis“, kuris nurodomas centimetrais.

Objekto savybė – tai antroji iš dviejų pagrindinių savybių kategorija. Jos paskirtis sujungti individus su kitais individais. Picos ontologijoje jau žinome apie klasę – Pica, įsivaizduokime, kad yra dar viena klasė – padas. Tuomet galime sujungti picą ir padą, kad „Pica turi Padą“.

Anotacijos – savybė, kuri suteikia papildomų žinių apie klasę ar individą. Pavyzdžiui: rdfs:comment. Picos ontologijos atveju, tai galėtų būti prie Vegetariškos picos klasės parašytas komentaras, kad picoje nėra mėsos.

4.1. GoodRelations elektroninės komercijos žodynas

GoodRelations yra standartizuotas žodynas, kuris taip pat vadinamas kaip „schema“, „duomenų žodynas“ ar netgi „ontologija“. Šis žodynas yra skirtas aprašyti įvairiems produktams pagal nurodytą tvarką. Įkėlus tokius duomenis į tinklalapį, juos rasti yra žymiai lengviau naudojant google, yahoo, bindex, bingo paiešką, nes yra vykdoma semantinė paieška. Kadangi daugumos paieškų sistemų naudoja sukurtą „Schema.org“ žodyną, kuris skirtas sužymėti interneto svetainėse patalpintam turiniui. GoodRelations tikslas yra sukurti struktūrą e-komercijai [HEP11]. Pavyzdžiui Volkswagen šiuo žodynu naudojasi savo tinklalapyje norėdami susisteminti paiešką.

GoodRelations žodynas yra nepriklausomas nuo industrinės srities, tinkamas tiek elektronikai, automobiliams, bilietams, įvairiausioms paslaugoms ar el. parduotuvių prekėms. Tinka ne tik galutiniai jau parduotuvei aprašyti prekes, bet nuo pačios prekės pardavimo grandinės pradžios.

Pačiu GoodRelations naudojasi daugiau nei 10 000 įvairaus dydžio parduotuvių visame pasaulyje. Yra įvairių turinio valdymo sistemų, kurios palaiko šį žodyną. Pavyzdžiui, Drupal, Magento, Pretashop, Joomla, Oxid eShop ir t.t. Žemiau pateikti kai kurie jų naudotojai ir trumpai paminėta kokiu tikslu jie naudoja GoodRelations žodyną.

Google – oficialiai rekomenduoja naudoti GoodRelations žodyną siekiant persiųsti struktūrizuotą informaciją į Google Rich Snippets. Tai yra skirta tikslesnei paieškai.

Yahoo – oficialiai rekomenduoja naudoti GoodRelations žodyną siekiant persiųsti duomenis jų SearchMonkey plėtiniui.

Best Buy - jiems GoodRelations yra pagrindinė skaitmeninio marketingo strategijos dalis. Jie publikuoja visus duomenis, įvairiausias akcijas naudodamiesi GoodRelations.

O‘Reilly – naudoja GoodRelations norėdami atlikti semantinį SEO visiems knygų pavadinimams.

Renault UK, OpenLink Software, Peek & Clppenburg, CSN Stores, Arzneimittel.de, Rakuten.de. Netgi toks automobilių pramonės gigantas kaip VolksWagen UK taip pat pasinaudojo GoodRelations teikiamu žodynu. Jų tikslas buvo naujas paieškos varikliukas tinklalapyje, kad potencialūs klientai galėtų rasti tikslesnius duomenis ir tai daryti žymiai greičiau.

Lygiai tokiu pačiu principu sudarius CMMI-DEV žodyną, vėliau organizacijų kūrimo procesų vertintojai pagal jį suvedę vertinimo duomenis galėtų lyginti skirtingus vertinimus. Tokiu būdu galima ne tik efektyviau įvertinti įmone, bet ir pasiūlyti proceso gerinimus, nes per užklausas galime rasti

silpnąsias vietas. Tai galime padaryti žymiai detaliau, nei naudojantis dabartiniais būdais. Taip pat galima konkretų vertinimą detaliau analizuoti, matant, kurie individai yra silpnesni, kurie stipresni ir pan.

4.2. Procesų vertinimo modelio žodyno idėja

Šis žodynas yra pavyzdinis ir buvo kuriamas atsižvelgiant į CMMI-DEV modelį, nes jis yra viešai prieinamas. Todėl jame nėra visų sąvokų, kurių gali reikėti. Jį ateityje galima pralėsti. Nusprendus sukurti žodyną iš jo buvo tikimasi:

- Pagal šį žodyną būtų galima aprašyti specifinio modelio (mūsų atveju CMMI-DEV 1.3) praktikas bei tikslus.
- Galimybė įvesti požymį „Įgyvendinta“, kad aprašius praktikas būtų galima matyti ar tam tikra praktika ar veiksmai joje buvo įgyvendinti.
- Aprašius specifinį modelį, būtų galima atlikti specifines paieškas nežinant daug informacijos. Taip surasti silpnas vietas.

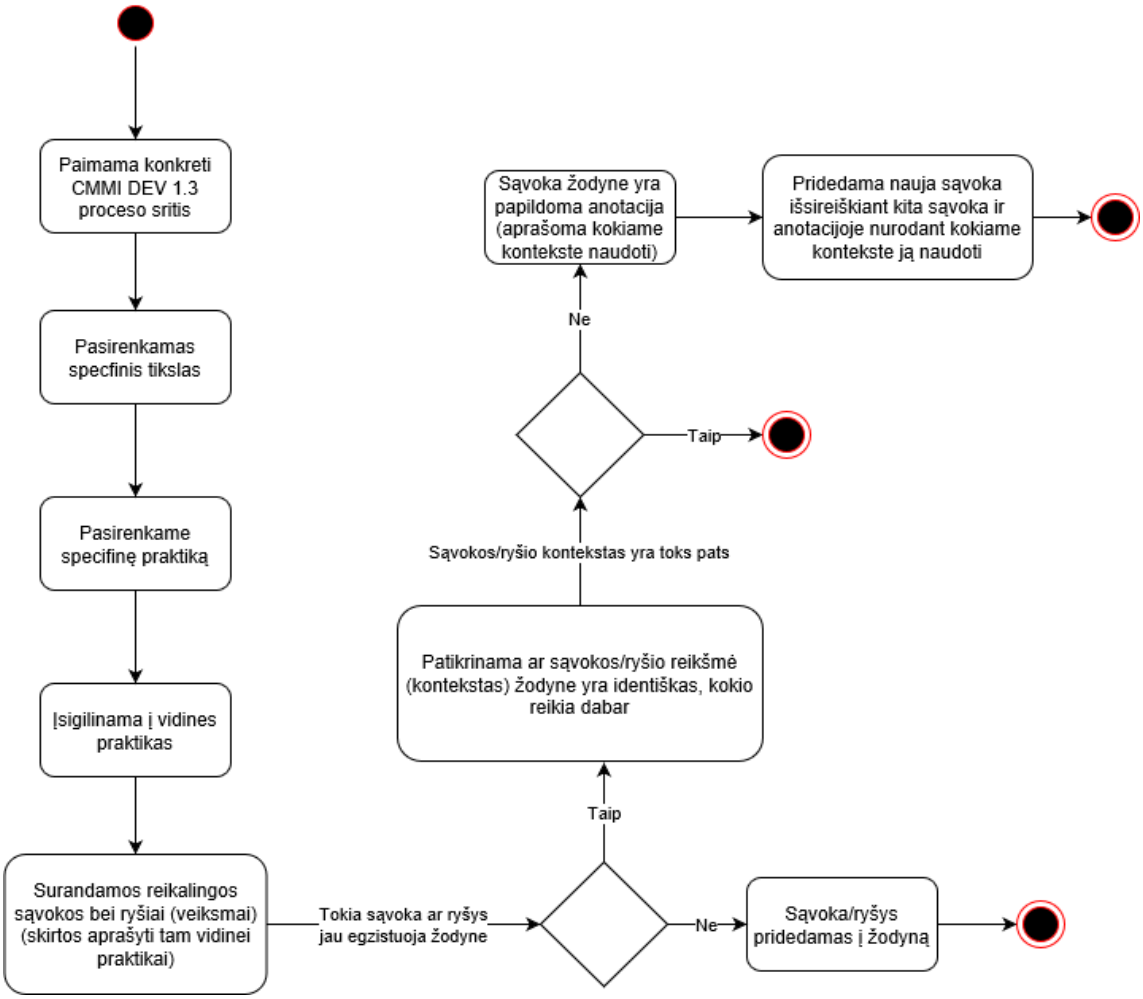
5. Procesų vertinimo modelių ontologija

Šiame darbe buvo kuriamas tik pavyzdinis žodynas, apimantis tik dalį sąvokų ir predikatų.. Kuriant šį žodyną buvo aišku, kad kai kurių individų būtinai reikės, todėl jie jau buvo sukurti iš anksto, kad žodyno naudotojui to daryti jau nereikėtų. Kuriant žodyną buvo pildomos tam tikros sritys – klasės, duomenų rūšies ypatybės, objektų ypatybės, anotacijos, individai.

5.1. Ontologijos kūrimo procedūra

Žodynas buvo kuriamas vykdant tokią darbų seką:

- Buvo paimama konkreti CMMI-DEV 1.3 proceso sritis
    - Gilinamasi į kiekvieną tikslą bei jo praktikas.
        - Atrenkamos sąvokos bei ryšiai (veiksmai), kurie reikalingi žodynui.



18 pav. Ontologijos kūrimo metodikos veiklos diagrama

5.2. Klasės

Dalis klasių, kurios buvo sukurtos žodyne:

Tiekėjas (angl. *Supplier*) – ši klasė yra tėvinė, po kuria sugula visos įmonės, kuri gamina PĮ klasės.

Architektų komanda (angl. *ArchitectureTeam*) – architektų komandos klasė, kurioje yra gali būti vaikinių klasių, arba jau patys individai.

Programuotojai (angl. *Developer*)– programuotojų klasė, kuri apibrėžia būtent programuotojus. Joje galima kurti poklases. Kadangi šiuo metu dar neaišku ar jų reikės – jos sukurtos nebuvo.

Klientas (angl. *Client*) – kliento klasė, ši klasė yra tėvinė iš kliento pusės, po ja rikiuojasi visos klasės, kurios yra reikalingos iš kliento pusės.

Produktas (angl. *Product*) – produkto klasė. Po šia klase rikiuojasi įvairios klasės, kurios susijusios su kuriamu produktu. Pavyzdžiui žemiau aprašyta dokumentų klasė.

Dokumentai (angl. *Documents*) – dokumentų klasė. Šioje klasė yra taip pat kuriamos kitos klasės, kad būtų lengviau klasifikuoti, pavyzdžiui, dokumentų yra įvairių tipų – įsipareigojimai (angl. *commitments*), reikalavimai (angl. *requirements*).

Įsipareigojimai (angl. *Commitments*) – įsipareigojimai. Šioje klasėje kuriami individai – dokumentai, kurie yra susiję su įsipareigojimais. Pavyzdžiui – įsipareigojimai laikytis nustatyto tvarkaraščio, kad būtų matyti, jog abi pusės įsipareigoja laikytis susitarimo laiku pristatyti darbus.

Reikalavimai (angl. *Requirements*) - reikalavimų klasė. Šioje klasėje kuriami įvairiausi individai, kurie yra susiję su reikalavimų dokumentais. Pavyzdžiui: produkto reikalavimai (angl. P*roduct requirements*), kriterijai reikalavimams (angl. *Requirements criteria*), reikalavimų pakeitimai (angl. *Requirements changes*)

Architektų komanda (angl. *ArchitectOwnerTeam*) - architektų klasė, kurioje galima kurti žemesnio lygio klases ir detaliau klasifikuoti architektus.

Analitikas (angl. *Analyst*) – analitiko klasė, kurioje yra kuriami individai – analitikai.

Integruotojas (angl. *Integrator*) – integruotojo klasė. Šioje klasėje kuriami individai bei poklasės, kurios apibrėžia asmenis atsakingus už produkto diegimą kliento aplinkoje. Taip pat, jeigu reikia apjungti dvi sistemas, už tai atsakingi taip pat integruotojai.

5.3. Duomenų rūšies ypatybės

Buvo sukurtos tik 4 tokios ypatybės, kurios pateiktos žemiau:

yraParuoštas (angl. *isPrepared*) – loginė ypatybė, kurios galimos reikšmės yra TRUE ir FALSE. Ši duomenų rūšies ypatybė galėtų būti naudojama su dokumentais. Pavyzdžiui siekiant pažymėti projekto planą paruoštu.

yraPabaigtas (angl. *isFinished*) – ypatybė, kuri nurodo ar kažkas yra pabaigtas. Galimos reikšmės TRUE ir FALSE. Galėtų būti naudojamas su projekto vykdymo etapais.

yraPasirašytas (angl. *isSigned*) – ypatybė, kuri nurodo, kad kažkas yra pasirašytas. Galimos reikšmės TRUE ir FALSE. Galėtų būti naudojamas su dokumentais ir įvairiais įsipareigojimais.

terminas (angl. *term*) – ypatybė, kuri nurodo terminą. Galimos reikšmės tik skaičiais. Galima naudoti norint nurodyti projekto arba konkretaus etapo terminą.

5.4. Objektų ypatybės

Žemiau pateikiame keletą jų, kad būtų galima susidaryti tikslesnį įspūdi, kam jos skirtos.

Paruošia (angl. *Prepares*). Ši ypatybė naudojama labai dažnai. Ypatingai dažnai su įvairiais trilypiais ryšiais, kurie apima dokumentus.

Identifikuoja (angl. *Identify*). Ši savybė nurodo, kad kažkoks individas kažką konkrečiai identifikuoja, tai gali būti naudojama tiek su projekto rizikomis, pavyzdžiui: projekto vadovas identifikuoja projekto rizikas.

Analizuoja (angl. *Analys*). Savybė, kuri daugiausiai naudojama prie analitiko (angl. *Analyst*) individo. Naudojama siekiant nurodyti, kad kažkuris individas kažką tiksliai analizuoja.

Vertina (angl. *Assess*). Savybė, kuri gali būti naudojama su daugeliu individu. Pradedant nuo projekto vadovo, kuris galėtų vertinti reikalavimus produktui. Kiek jie yra realūs.

Dokumentuoja (angl. *Documents*). Tiksli ypatybė, kuri skirta nurodyti, kad kažkas yra dokumentuojama.

Palaiko (angl. *Maintains*). Savybė, kuri nurodo, kad kažkas kažką palaiko. Ši savybė naudojam tokiame kontekste: projektų vadovas palaiko nesuderinamumo dokumentą (reiškia, kad jis pastoviai jį pildo ir yra už tai atsakingas, tai nėra vienatinis darbas).

Įgauna (angl. Obtains). Savybė, kuri buvo pridėta dėl konkretaus atvejo – projekto vadovas gauni įsipareigojimus projektui.

Peržiūri (angl. *Reviews*). Ši savybė naudojama tuomet, kai norima pasakyti, kad kažkas peržiūri tam tikrą dokumentą.

Pasirašo (angl. *Signs*). Pasirašymas turėtų būti naudojamas su kažkokių įsipareigojimų, sutarčių pasirašymų.

Stebi (angl. *Monitors*). Stebėjimas nėra vienetinis darbas, jis tęsiasi viso projekto laiku. Kaip pavyzdys galėtų būti – projekto vadovas stebi projekto eigą.

Užtikrina (angl. *Ensures*). Naudojama norint pabrėžti, jog kažkas yra užtikrinama.

5.5. Anotacijos

Žemiau yra pora pavyzdžių, kaip naudojamos anotacijos žodyne:

Komentaras (angl. *Commet*) – aprašant klasę ar kokį nors individą, prie jo parašome komentarus, kad būtų galima suprasti kokiame kontekste norima naudoti tam tikra sąvoka. Ši anotacija ypatingai padėtų žmonės susigaudyti pirmą kartą naudojantis šiuo žodynu.

Įgyvendinta (angl. *Implemented*) – aprašant tam tikrą trilypį ryši, kuris priklauso kažkokiai praktikai, pažymime tai įgyvendinta ar ne.

5.6. Individai

Žemiau pateikiame dalį sukurtų individų, kurie gali būti naudojami ir toliau aprašant praktikas, taip pat jų gali atsirasti ir daugiau.

KlientoAnalitikas (angl. *ClientAnalyst*) – analitikas, kuris pažymėtas kaip būtent analitikas iš kliento pusės. Pagrindinės šios rolės veiklos bus siejamos su darbų priežiūra, ar jie atliekami tikslingai. Sudaromi reikalavimai, galbūt reikalavimų pakeitimų registravimas ir iš kliento pusės.

KlientoDirektorus (angl. *ClientDirector*) – direktorius iš kliento pusės, tai asmuo, kuris yra viršiausias ir atsako už projektą(produktą, kurį bando įsigyti). Šis asmuo pasirašo visus reikiamus dokumentus, visus įsipareigojimus.

Architektas (angl. *Architect*) – tai rolė, kuri yra atsakinga už techninę projekto dalį. Turi suprojektuoti visą būsimą sistemą. Tiek dėl reikalavimų pačiai aplinkai, kurioje bus diegiama sistema, tiek programavimo kalbą, karkasą ir t.t..

Analitikas (angl. *Analyst*) – rolė, kuri daugiausiai bendrauja su klientu. Yra atsakinga už įvairias dokumentacijas, tokias kaip – reikalavimų kriterijų sudarymą, turi išsiaiškinti tiksliai ko nori klientas. Suprasti kliento poreikius.

Programuotojas (angl. *Developer*) – rolė, kuri atsakinga už programavimo darbus ir papildomai jau priklausomai nuo organizacijos.

Testuotojas (angl. *Tester*) – asmuo, kuris yra atsakingas už programinės įrangos testavimą. Taip pat įeina ir tokios veiklos kaip testavimo plano sudarymas, pačių testų rašymas, komunikavimas su programuotojais, dokumentavimas visą ko, kas susiję su testavimu ir klaidų taisymu. Taip pat testavimo darbų atlikimas jau kliento aplinkoje įdiegus programinę įrangą.

Integruotojas (angl. *Integrator*) – asmuo, atsakingas už programinės įrangos integravimą kliento aplinkoje, bei pilną jos paleidimą.

Projekto vadovas (angl. *ProjectManager*) – tai viena esminių rolių, kuri turi begalę įvairiausių pareigybių. Tai įvairiausios pareigybės, kurios susiję su projekto valdymu. Pavyzdžiui įvairiausi projekto planų rengimai, rizikų nustatymas, rizikų valdymas ir t.t.

Evoliuciniai priėmimo reikalavimai (angl. *EvolutionAcceptanceRequirements*) - šis individas aprašo reikalavimus, kuriuos turi pasiekti projektas norėdamas atsiskaityti darbus tam tikrame etape.

Kriterijai Reikalavimams (angl. *RequirementsCriteria*) – individas, kuris reiškia kriterijus reikalavimams. Šie kriterijai yra skirti klientui, kad pagal juos sudarytų reikalavimus projektui. Tai tam tikros gairės.

Išoriniai įsiapreigojimai (angl. *ExternalCommitments*). Naudojamas kai kalbama apie kažkokius įsipareigojimus klientui, galbūt subrangovams.

Vidiniai įsipareigojimai (angl. *InternalCommitments*). Naudojamas kai kalbama apie vidinius įsipareigojimus pačioje įmonėje.

Projekto įsipareigojimai (angl. *ProjectCommitments*). Aiški rolė, kuri nurodo įsipareigojimus projektui. Juos turi pasirašyti ir įsipareigoti abi pusės – tiek programinės įrangos kūrėjai, tiek programinės įrangos užsakovai.

Veiklos žurnalas (angl. *ActivitiesLog*). Naudojamas, kai norima apibrėžti, kad kažkas pildo, galbūt seka veiklos žurnalą ir stebi visą darbų seką.

Privatumas (angl. *Privacy*). Naudojamas, kai norima apibrėžti privatumą.

Projekto istoriniai duomenys (angl. *ProjectHistoricalData*). Naudojamas, kai norima kalbėti apie istorinius duomenis, juos stebėti, analizuoti.

Projekto rizika (angl. *ProjectRisk*) – jas turi identifikuoti projekto vadovas. Į rizikas įeina tiek vidiniai veiksniai iš įmonės vidaus – išeina darbuotojas iš darbo ir pan. Taip pat ir išoriniai veiksniai, pavyzdžiui, subrangovas vėluoja atlikti tam tikrus darbus ir dėl to visas projektas gali vėluoti, kas galimai labai atsilieptų finansiškai.

Reikiami įgūdžiai (angl. *RequiredSkills*) – reikiami įgūdžiai. Reikiami įgūdžiai apibrėžia įgūdžius, kurie reikalingi būtent tam projektui. Tai apima tiek patirtį, tiek programavimo kalbas, įrankius, karkasus ir dar daugelį kitų kriterijų.

Techninis požiūris (angl. *TechnicApproach*). Skirtas apibrėžti technines savybes, kažkokius techninius kriterijus.

Pagrindiniai etapai (angl. *MajorMilestones*). Būtinas individas, kadangi bet kurioje įmonėje projektas turi kažkokius etapus.

Projekto planas (angl. *ProjectPlan*). Taip pat būtinas individas, kurio prireiks aprašant bet kurios įmonės procesus.

Preliminarus tvarkaraštis (angl. *ScheduleAssumptions*). Šis individas naudojamas siekiant aprašyti konkrečius darbus kiekviename projekto etape.

Produkto reikalavimai (angl. *ProductRequirements*). Taip pat privalomas individas, kadangi įmonės užsako, kad būtų kuriamas tam tikra programinė įranga, tad natūralu, kad ji turi atitikti tam tikrus reikalavimus.

Reikalavimų pakeitimai (angl. *RequirementsChanges*). Reikalavimų pakeitimai turi būti registruojami. Kaip pakito nuo projekto pradžios. Taip pat kiekvienas pakeitimas turi būti išanalizuotas ir įvertintas, kaip stipriai tai paveiks visą sistemą. Atsižvelgiama į visus kriterijus, pavyzdžiui, ar pakeitus reikalavimą, kuris yra pradžioje, ar bus įmanoma įgyvendinti tuos reikalavimus, kurie bus įgyvendinami projekto pabaigoje.

Reikalavimų atsekamumo matrica (angl. *RequirementsTraceabilityMatrix*) – reikalavimų atsekamumo matricą pildo projektų vadovas, taip pat analitikas.

Produkto komponentas (angl. *ProductComponent*) – galimas variantas, kad kuriamas produktas turės kažkokių komponentų.

6. Procesų vertinimo modelių žodyno pritaikymas praktiškai

Tai yra pavyzdys žodyno, kuris apima CMMI-DEV 2 brandos lygio specifines praktikas. Atliekant literatūros analizę nebuvo įmanoma rasti panašių žodynų, kad jie būtų pakankamai žemo lygio, nes visi buvo tik abstrakčiausiame lygyje. Šis žodynas yra daugiau pavyzdinis, kurio tikslas pristatyti idėją ir pateikti gaires tolimesniems darbams. Todėl kuriant žodyną gali atsirasti neatitikimų, arba aprašant praktikas, jos gali būti interpretuotos ne visiškai teisingai. Yra sukurta UML diagrama, kurios labai maža dalis (dalis iškarpyta, nes netilptų į lapą ir nepavyktų nieko perskaityti popieriniame variante). Ji pateikta 1 priede. Žodynas apima tik šias proceso sritis:

- Reikalavimų valdymas REQM (angl. *Requirements Management*)
- Projekto planavimas PP (angl. *Project Planning*)
- Projekto priežiūra bei kontrolė PMC (angl. *Project Monitoring and Control*)
- Matavimai ir analizė MA (angl. *Measurement and Analysis*)
- Konfigūracijos valdymas CM (angl. *Configuration Management*)
- Procesų ir produktų kokybės užtikrinimas (PPQA) (angl. *Process and Product Quality Assurance*)
- Tiekėjo sutarčių valdymas (SAM) (angl. *Supplier Agreement Management*)

6.1. Trilypiai ryšiai

Aprašant praktikas buvo naudojami trilypiai ryšiai. Loginių ryšių išsidėliojo dvigubai daugiau, nei aprašytų. Aprašant praktikas buvo naudojamasi SEI (Software Engineering Institue) paruošta technine ataskaita „CMMI® for Development, Version 1.3“[SEI10].

6.1.1. Trilypiai ryšių kūrimas

Trilypių ryšių kūrimas vyksta tokia seka:

- ➢ Pasirenkama proceso sritis (pavyzdžiui reikalavimų valdymas REQM (angl. *Requirements Management*)
  - ➢ Matome, kad ši proceso sritis turi 1 specifinį tikslą –valdyti reikalavimus (angl. *Manage Requirements*). Tad šį specifinį tikslą ir pasirenkame.
    - ➢ Toliau matome, kad šis specifinis tikslas turi 5 specifines praktikas. Pasirenkame pirmąją – suprasti reikalavimus (angl. *Understand Requirements*). Matome, kad ji turi 4 vidines praktikas (angl. *Subpractices*). Pasirenkame ir aprašome visas jas.

- „Išskirti kriterijus tinkamiems reikalavimams, kurie būtų skirti reikalavimų tiekėjams“. Iš šios praktikos apibūdinimo galime daryti prielaidą, kad analitikas iš organizacijos, kuri kuria programinę įrangą paruošia reikalavimų kriterijų dokumentą. Ir trilypis ryšys turėtų atrodyti taip:
  - Analitikas paruošia ReikalavimųKriterijus (angl. *Analyst prepares RequirementsCriteria*).
- „Sukurti kriterjus, pagal kuriuos galima būtų sukurti evoliucinius reikalavimus. Priėmimo reikalavimus.“. Daroma prielaida, kad kažkas turi sukurti kriterijus, pagal kuriuos pasibaigus etapui būtų vertinami darbai. Ar per etapą pavyko pasiekti užsibrėžtus tikslus ir esamas rezultatas atitinka užsibrėžtus tikslus. Trilypis ryšys galėtų atrodyti taip:
  - Analitikas paruošia PriėmimoReikalavimųKriterijus (angl. *Analyst prepares AcceptanceRequiremensCriteria*).
- „Išanalizuoti reikalavimus, kad būtų užtikrinama, ar parašyti reikalavimai atitinka jiems pateiktus kriterijus“. Kai klientai parašo reikalavimus norimam produktui (programinei įrangai), analitikas turėtų išanalizuoti reikalavimus, patikrinti ar jie atitinka reikalavimų kriterijus. Sukurtas trilypis ryšys galėtų atrodyti taip:
  - Analitikas analizuoja Reikalavimus (angl. *Analyst analyze Requirements*).
- „Tinkamai suprasti reikalavimus, suderinti juo su užsakovu“. Manoma, kad čia siekiama, jog neliktų kažkokių nesusipratimų ir tiek kliento, tiek iš tiekėjo pusės būtų tinkamas reikalavimų supratimas. Todėl trilypius ryšius galima parašyti kelis:
  - Analitikas supranta reikalavimus (angl. *Analyst understands Requirements*).
  - Analitikas komunikuoja su užsakovo analitiku (angl. *Analyst communicate ClientAnalyst*).

Žemiau lentelėje yra pateikti visi trilypiai ryšiai, kurie buvo sudaryti aprašant praktikas.

13 lentelė. Proceso sritys aprašytos trilypiais ryšiais

Nr.	Pavadinimas	Trilypis ryšys
1.	**Proceso sritis**: reikalavimų valdymas REQM (angl. *Requirements Management*)	
1.1	**Specifinis tikslas:** Valdyti reikalavimus (angl. *Manage Requirements*)	
1.1.1.	Suprasti reikalavimus (angl. *Understand Requirements*)	Analyst prepares RequirementsCriteria

1.1.1.	Suprasti reikalavimus (angl. *Understand Requirements*)	Analyst prepares EvolutionAcceptanceRequirements
1.1.1.	Suprasti reikalavimus (angl. *Understand Requirements*)	ClientAnalyst prepares EvolutionAcceptanceRequirements
1.1.1.	Suprasti reikalavimus (angl. *Understand Requirements*)	ClientAnalyst prepares ProductRequirements
1.1.2.	Gauti įsipareigojimus į reikalavimus (angl. *Obtain Commitment to Requirements*)	ProjectManager assess Requirements
1.1.2.	Gauti įsipareigojimus į reikalavimus (angl. *Obtain Commitment to Requirements*)	ProjectManager signs ProjectCommitments
1.1.2.	Gauti įsipareigojimus į reikalavimus (angl. *Obtain Commitment to Requirements*)	ClientDirector signs ProjectCommitments
1.1.3.	Tvarkyti pakeitimus (angl. *Manage Requirementss Changes*)	Analyst documents RequirementsChanges
1.1.3.	Tvarkyti pakeitimus (angl. *Manage Requirementss Changes*)	ProjectManager evaluates RequirementsChanges
1.1.4.	Palaikyti abipusį atsekamumą tarp reikalavimų ir produkto (angl. *Maintain Bidirectional Traceability of Requirements*)	Analyst maintains RequirementsTraceabilityMatrix
1.1.4.	Palaikyti abipusį atsekamumą tarp reikalavimų ir produkto (angl. *Maintain Bidirectional Traceability of Requirements*)	ProjectManager review ProjectPlans
1.1.4.	Palaikyti abipusį atsekamumą tarp reikalavimų ir produkto (angl. *Maintain Bidirectional Traceability of Requirements*)	ProjectManager review ActivitiesLog
1.1.4.	Palaikyti abipusį atsekamumą tarp reikalavimų ir produkto (angl. *Maintain Bidirectional Traceability of Requirements*)	ProjectManager identify Inconsistency
1.1.5.	Užtikrinti atitikimą tarp reikalavimų ir produkto (angl. *Ensure Alignment Between Project Work and Requirements*)	ProjectManager review Requirements
1.1.5.	Užtikrinti atitikimą tarp reikalavimų ir produkto (angl. *Ensure Alignment*	ProjectManager review Product

	Between Project Work and Requirements)	
2.	**Proceso sritis:** Projekto planavimas (angl. *Project Planning*)	
2.1.	**Specifinis tikslas:** Nustatyti matavimus (angl. *Establish Estimates*)	
2.1.1.	Nustatyti projekto apimtį (angl. *Estimate the Scope of the Project*)	ProjectManager creates WBS
2.1.1.	Nustatyti projekto apimtį (angl. *Estimate the Scope of the Project*)	ProjectManager identify ProductComponents
2.1.2.	Nustatyti įverčius (angl. *Establish Estimates of Work Product and Task Attributes*)	ProjectManager determines TechnicalApporach
2.1.2.	Nustatyti įverčius (angl. *Establish Estimates of Work Product and Task Attributes*)	Architect determines TechnicalApporach
2.1.2.	Nustatyti įverčius (angl. *Establish Estimates of Work Product and Task Attributes*)	Analyst determines TechnicalApproach
2.1.3.	Nustatyti projekto gyvavimo ciklo fazes. (angl. *Define Projec Lifecycle Phases*)	ProjectManager determines ProjectLifeCycle
2.1.4.	Įvertinti laiką ir kainą (angl. *Estimate Effort and Cost*)	Analyst analyze ProjectHistoricalData
2.2.	**Specifinis tiklsas:** Sukurti projekto planą (angl. *Develop a Project Plan*)	
2.2.1.	Nustatyti tvarkaraštį ir biudžetą (angl. *Establish the Budget and Schedule*)	ProjectManager identify MajorMilestones
2.2.1.	Nustatyti tvarkaraštį ir biudžetą (angl. *Establish the Budget and Schedule*)	ProjectManager identify ScheduleAssumptions
2.2.1.	Nustatyti tvarkaraštį ir biudžetą (angl. *Establish the Budget and Schedule*)	ProjectManager identify Constraints
2.2.2.	Identifikuoti Projekto rizikas (angl. *Identify Project Risks*)	ProjectManager identify ProjectRisks
2.2.2.	Identifikuoti Projekto rizikas (angl. *Identify Project Risks*)	Analyst identify ProjectRisks
2.2.3.	Saugumas (angl. *Security*)	ProjectManager ensures Security
2.2.4.	Planuoti projekto resursus (angl. *Plan the Project‘s Resources*)	Projectmanager determines ProjectRequirements
2.2.4.	Planuoti projekto resursus (angl. *Plan the Project‘s Resources*)	ProjectManager identify RequiredSkills

2.2.5.	Planuoti reikiamas žinias ir įgudžius (angl. *Plan Need Knowledge and skills*	ProjectManager prepares RequiredSkillsDocument
2.2.6.	Planuoti suinteresuotų šalių įtraukimą. (angl. *Plan Stakeholder Involvement*)	ProjectManager plans StakeHolderInvolvment
2.2.7.	Parengti projekto planą (angl. *Establish the Project Plan*)	Projectmanager prepares OverallProjectPlan
2.3.	**Specifinis tikslas:** Gauti įsipareigojimus planui (angl. *Obtain Commitment to the Plan*)	
2.3.1.	Peržiūrėti planus, kurie įtakos projektui (angl. *Review Plans That Affect the Project*)	ProjectManager record ProjectReviews
2.3.2.	Suderinti reikiamus ir turimus resursus (angl. *Reconcile differences between estimates and available resources*)	ProjectManager revise schedul
2.3.2.	Suderinti reikiamus ir turimus resursus (angl. *Reconcile differences between estimates and available resources*)	ProjectManager revise Requirements
2.3.3.	Gauti įsipareigojimus planui (angl. *Obtain Plan Commitment*)	ProjectManager reviews InternalCommitments
2.3.3.	Gauti įsipareigojimus planui (angl. *Obtain Plan Commitment*)	ProjectManager reviews ExternalCommitments
2.3.3.	Gauti įsipareigojimus planui (angl. *Obtain Plan Commitment*)	ProjectManager reviews ProjectCommitments
3.	**Proceso sritis:** Projekto stebėjimas ir kontrolė PMC (angl. *Project Monitoring and Control*)	
3.1.	**Proceso tikslas:** Stebėti projektą ir projekto planą (angl. *Monitor the Project Against the Plan*)	
3.1.1.	Stebėti projekto planavimo parametrus (angl. *Monitor Project Planning Parameters*)	ProjectManager monitors ProjecProgress
3.1.1.	Stebėti projekto planavimo parametrus (angl. *Monitor Project Planning Parameters*)	ProjectManager monitors Plans
3.1.2.	Stebėti įsipareigojimus (angl. *Monitor commitments*)	ProjectManager monitors Commitments
3.1.3.	Stebėti projekto rizikas (angl. *Monitor Project Risks*)	ProjectManager monitors Risks
3.1.4.	Stebėti duomenų valdymą (angl. *Monitor Data Management*)	DataAnalyst monitors DataManagement
3.1.5.	Stebėti suinteresuotų šalių įsitraukimą (angl. *Monitor Stakeholder Involvement*)	ProjectManager monitors StakeHolders

3.1.6.	Stebėti projekto progresą (angl. *Conduct Progress Reviews*)	ProjectManager reviews Performance
3.1.7.	Stebėti projekto etapus (angl. *Conduct Milestone Reviews*)	ProjectManager reviews Milestone
3.2.	**Proceso tikslas:** Valdyti koregavimo veiksmus (angl. *Manage Corrective Action to Closure*)	
3.2.1.	Analizuoti klaidas (angl. *Analyze Issues*)	Analyst collects Issues
3.2.1.	Analizuoti klaidas (angl. *Analyze Issues*)	Analyst analyze Issues
3.2.1.	Analizuoti klaidas (angl. *Analyze Issues*)	Analyst determines CorrectionActions
3.2.2.	Imtis koregavimo veiksmų (angl. *Take Corrective Action*)	Analyst prepares CorrectiveActionsPlan
3.2.3.	Valdyti koregavimo veiksmus (angl. *Manage Corrective Actions*)	Analyst monitors CorrectionActions
4.	**Proceso sritis:** Matavimas ir Analizė MA (angl. Measurement *and Analysis*)	
4.1.	**Proceso tikslas:** Suderinti matavimo ir analizės veiklas (angl. *Align Measurement and Analysis Activities*)	
4.1.1.	Nustatyti matavimo tikslus (angl. *Establish Measurement Objectives*)	Analyst documents MeasurementObjectives
4.1.1.	Nustatyti matavimo tikslus (angl. *Establish Measurement Objectives*)	Analyst prioritize MeasurementObjectives
4.1.1.	Nustatyti matavimo tikslus (angl. *Establish Measurement Objectives*)	ProjectManager review MeasurementObjectives
4.1.1.	Nustatyti matavimo tikslus (angl. *Establish Measurement Objectives*)	Analyst maintain TraceabillityOfMeasurementObjectives
4.1.2.	Nurodyti matavimus (angl. *Specify Measures*)	Analyst Identify Mesures
4.1.2.	Nurodyti matavimus (angl. *Specify Measures*)	ProjectManager review Measures
4.1.3.	Nurodyti duomenų rinkimo ir talpinimo procedūras (angl. *Specify Data Collection and Storage Procedures*)	TechnicalAnalyst identify ExistingData
4.1.3.	Nurodyti duomenų rinkimo ir talpinimo procedūras (angl. *Specify Data Collection and Storage Procedures*)	Analyst identify Measures

4.1.3.	Nurodyti duomenų rinkimo ir talpinimo procedūras (angl. *Specify Data Collection and Storage Procedures*)	TechnicalAnalyst specifies DataCollectionProcedure
4.1.3.	Nurodyti duomenų rinkimo ir talpinimo procedūras (angl. *Specify Data Collection and Storage Procedures*)	TechnicalAnalyst creates DataCollectionProcedure
4.1.3.	Nurodyti duomenų rinkimo ir talpinimo procedūras (angl. *Specify Data Collection and Storage Procedures*)	TechnicalAnalyst starts DataCollectionProcedure
4.1.4.	Nurodyti analizės procedūras (angl. *Specify Analysis Procedures*)	Analyst specifies Analyses
4.1.4.	Nurodyti analizės procedūras (angl. *Specify Analysis Procedures*)	Analyst selects DataAnalysisMethods
4.1.4.	Nurodyti analizės procedūras (angl. *Specify Analysis Procedures*)	Analyst selects DataAnalysisTools
4.1.4.	Nurodyti analizės procedūras (angl. *Specify Analysis Procedures*)	Analyst update Measures
4.1.4.	Nurodyti analizės procedūras (angl. *Specify Analysis Procedures*)	Analyst update MeasurementObjectives
4.2.	**Proceso tikslas:** Pateikti matavimo rezultatus (angl. *Provide Measurement Results*)	
4.2.1.	Gauti matavimo duomenis (angl. *Obtain Measurement Data*)	Analyst obtains MeasurementData
4.2.1.	Gauti matavimo duomenis (angl. *Obtain Measurement Data*)	Analyst generates DataForMeasurement
4.2.2.	Analizuoti matavimo duomenis (angl. *Analyze Measurement Data*)	Analyst conducts InitialAnalyses
4.2.2.	Analizuoti matavimo duomenis (angl. *Analyze Measurement Data*)	Analyst prepares InitalAnalysesResults
4.2.2.	Analizuoti matavimo duomenis (angl. *Analyze Measurement Data*)	Analyst improves InitialAnalyses
4.2.3.	Saugoti duomenis ir rezultatus (angl. *Store data and results*)	Analyst ensures Data
4.2.3.	Saugoti duomenis ir rezultatus (angl. *Store data and results*)	Analyst stores Data
4.2.3.	Saugoti duomenis ir rezultatus (angl. *Store data and results*)	Analyst protect Data

4.2.4.	Aptarti rezultatus (angl. *Communicate Results*)	ProjectManager informs StakeHolders
4.2.4.	Aptarti rezultatus (angl. *Communicate Results*)	Analyst assists StakeHolders
5.	**Proceso sritis:** Konfigūracijos valdymas (angl. *Configuration Management*)	
5.1.	**Proceso tikslas:** Nustatyti bazines linijas (angl. *Establish baselines*)	
5.1.1.	Nustatyti konfigūracijos elementus (angl. *Identify configuration items*)	TechnicalAnalyst select ConfigurationItems
5.1.1.	Nustatyti konfigūracijos elementus (angl. *Identify configuration items*)	TechnicalAnalyst specifies CharacteristicsforConfigurationItems
5.1.1.	Nustatyti konfigūracijos elementus (angl. *Identify configuration items*)	TechnicalAnalyst identifies OwnersOfConfigurationItems
5.1.1.	Nustatyti konfigūracijos elementus (angl. *Identify configuration items*)	TechnicalAnalyst specifies RelationShipsAmongConfigurationItems
5.1.2.	Sukurti konfigūracijos valdymo sistemą (angl. Establish a Configuration Management System)	Developer establish ConfigurationSystemControl.
5.1.2.	Sukurti konfigūracijos valdymo sistemą (angl. Establish a Configuration Management System)	Developer provides Authorization
5.1.2.	Sukurti konfigūracijos valdymo sistemą (angl. Establish a Configuration Management System)	Developer stores ConfigurationItems
5.1.2.	Sukurti konfigūracijos valdymo sistemą (angl. Establish a Configuration Management System)	Developer retreves ConfigurationItems
5.1.2.	Sukurti konfigūracijos valdymo sistemą (angl. Establish a Configuration Management System)	Developer controls ConfigurationSystemVersions
5.1.2.	Sukurti konfigūracijos valdymo sistemą (angl. Establish a Configuration Management System)	TechnicalAnalyst creates ConfigurationManagmentReports
5.1.3.	Sukurti arba įgyvendinti pradines linijas (angl. *Create or Release Baselines*)	Analyst creates Baselines
5.1.3.	Sukurti arba įgyvendinti pradines linijas (angl. *Create or Release Baselines*)	Analyst documents Baselines
5.2.	**Proceso tikslas:** Sekti ir kontroliuoti pakeitimus (angl. *Track and control changes*)	

5.2.1.	Sekti pakeitimų užklausas (angl. *Track change requests*)	Architect prepares ChangeRequestDatabase
5.2.1.	Sekti pakeitimų užklausas (angl. *Track change requests*)	Analys analyze ImpactOfChangeRequests
5.2.1.	Sekti pakeitimų užklausas (angl. *Track change requests*)	Analyst categorizes ChangeRequests
5.2.1.	Sekti pakeitimų užklausas (angl. *Track change requests*)	Analyst prioritzes ChangeRequests
5.2.1.	Sekti pakeitimų užklausas (angl. *Track change requests*)	Analyst tracks ChangeRequests
5.2.2.	Valdyti konfigūracijos valdymo pakeitimus (angl. *Control changes to configuration items*)	Analyst reviews ChangeRequests
5.3.	**Proceso tikslas:** Nustatyti integravimą (angl. *Establish Integrity*)	
5.3.1.	Sukurti konfigūracijos valdymo įrašus (angl. *Establish configuration management records*)	Analyst seeks VersionsOfBaselines
5.3.1.	Sukurti konfigūracijos valdymo įrašus (angl. *Establish configuration management records*)	Analyst describes differencesBetweenBaselines
5.3.2.	Atlikti konfigūracijos auditą (angl. *Perform configuration audits*)	TechnicalAnalyst assess IntegrityOfBaselines
5.3.2.	Atlikti konfigūracijos auditą (angl. *Perform configuration audits*)	TechnicalAnalyst review ConfigurationManagementRecords
5.3.2.	Atlikti konfigūracijos auditą (angl. *Perform configuration audits*)	Analyst tracks ActionItems
6	**Proceso sritis:** Procesų ir produktų kokybės užtikrinimas (angl. *Process and product quality assurance*)	
6.1.	**Proceso tikslas:** Objektyviai vertinti procesus ir darbo produktus (angl. *Objectively evaluate processes and work products*)	
6.1.1.	Objektyviai įvertinti procesus (angl. *Objectively evaluate processes*)	ProjectManager creates workEnvironment
6.1.1.	Objektyviai įvertinti procesus (angl. *Objectively evaluate processes*)	ProjectManager creates EvaluationsCriteria
6.1.1.	Objektyviai įvertinti procesus (angl. *Objectively evaluate processes*)	ProjectManager identifies NoncompliancesForProcess
6.1.1.	Objektyviai įvertinti procesus (angl. *Objectively evaluate processes*)	ProjectManager improves Process

6.1.2.	Objektyviai įvertinti darbo produktus (angl. *Objectively evaluate work products*)	ProjectManager selects WorkProducts
6.1.2.	Objektyviai įvertinti darbo produktus (angl. *Objectively evaluate work products*)	ProjectManager follows WorkProductsEvolution
6.1.2.	Objektyviai įvertinti darbo produktus (angl. *Objectively evaluate work products*)	ProjectManager creates CriteriaForWorkProducts
6.1.2.	Objektyviai įvertinti darbo produktus (angl. *Objectively evaluate work products*)	ProjectManager identifies NoncompliancesForProducts
6.1.2.	Objektyviai įvertinti darbo produktus (angl. *Objectively evaluate work products*)	ProjectManager improves Process
6.2.	**Proceso tikslas:** Pateikti objektyvias įžvalgas (angl. *Provide objective insight*)	
6.2.1.	Komunikuoti ir išspręsti neatitikimus (angl. *Communicate and resolve noncompliance issues*)	StakeHolders review ResultsOfEvolutions
6.2.1.	Komunikuoti ir išspręsti neatitikimus (angl. *Communicate and resolve noncompliance issues*)	Analyst resolves NoncompliancesForProducts
6.2.1.	Komunikuoti ir išspręsti neatitikimus (angl. *Communicate and resolve noncompliance issues*)	Analyst resolves NoncompliancesForProcess
6.2.1.	Komunikuoti ir išspręsti neatitikimus (angl. *Communicate and resolve noncompliance issues*)	ProjectManager ensures NonCompliancesResolution
6.2.2.	Tvarkyti įrašus (angl. *Establish records*)	Analyst records ProcessAssuranceActivities
6.2.2.	Tvarkyti įrašus (angl. *Establish records*)	Analyst records ProductQualityAssuranceActivities
6.2.2.	Tvarkyti įrašus (angl. *Establish records*)	Analyst analyzes HistoricalDataOfQualityAssuranceActivities
7.	**Proceso sritis:** Tiekėjo sutarčių valdymas (SAM) (angl. *Supplier Agreement Management*)	
7.1.	**Proceso tikslas:** Nustatyti tiekėjo sutartis (angl. *Estatblish Supplier Agreements*)	
7.1.1.	Nustatyti įsigijimo tipą (angl. *Determine acquisition type*)	ProjectManager determines AcquisitionType
7.1.2.	Pasirinkti tiekėjus (angl. *Select susppliers*)	ProjectManager prepares cirteriaForSuppliers

7.1.2.	Pasirinkti tiekėjus (angl. *Select susppliers*)	ProjectManager identifies PotensialSuppliers
7.1.2.	Pasirinkti tiekėjus (angl. *Select susppliers*)	ProjectManager evaluates RisksOfSupplier
7.1.2.	Pasirinkti tiekėjus (angl. *Select susppliers*)	ProjectManager selects Supplier
7.1.3.	Paruošti sutartį tiekėjams (angl. *Establish supplier agreements*)	Analyst prepares RequirementsForSupplier
7.1.3.	Paruošti sutartį tiekėjams (angl. *Establish supplier agreements*)	Analyst documents SupplierAgreement
7.1.3.	Paruošti sutartį tiekėjams (angl. *Establish supplier agreements*)	Projectmanager ensures SupplierAgreement
7.2.	**Proceso tikslas:** Įgyvendinti sutarties su tiekėjais sąlygas (angl. *Satisfy supplier agreements*)	
7.2.1.	Įgyvendinti įsipareigojimus nurodytus sutartyje su tiekėjais (angl. *Execute the supplier agreement*)	ProjectManager monitors SupplierProgress
7.2.1.	Įgyvendinti įsipareigojimus tiekėjams (angl. *Execute the supplier agreement*)	ProjectManager review WorkProductFromSuppliers
7.2.1.	Įgyvendinti įsipareigojimus tiekėjams (angl. *Execute the supplier agreement*)	ProjectManager follows RisksOfSupplier
7.2.2.	Priimti užsakytą produktą (angl. *Accept the acquired product*)	ProjectManager defines ProductAcceptanceProcedures
7.2.2.	Priimti užsakytą produktą (angl. *Accept the acquired product*)	ProjectManager obtain AgreementsFromStakeHolders
7.2.2.	Priimti užsakytą produktą (angl. *Accept the acquired product*)	ProjectManager verifies AcquiredProduct
7.2.2.	Priimti užsakytą produktą (angl. *Accept the acquired product*)	Analyst documents ResultsOfAcceptanceTests
7.2.3.	Užtikrinti perėjimą prie produktų iš tiekėjų (angl. *Ensure transition of products*)	Analyst prepares PlanForTransition
7.2.3.	Užtikrinti perėjimą prie produktų iš tiekėjų (angl. *Ensure transition of products*)	SystemAdministrator prepares SystemToTransition
7.2.3.	Užtikrinti perėjimą prie produktų iš tiekėjų (angl. *Ensure transition of products*)	Developer ensures Transition

6.2. Duomenų paieška

Konkrečių duomenų paieškai naudojome SPARQL užklausas. OWL kalba aprašytos ontologijos mums leidžia naudotis įvairius ryšius, išplaukiančius iš taisyklių. Todėl tai yra labai patogu ir naudinga. Būtent savo ontologijoje pažymime, kad tam tikros praktikos užpildo tam tikrą proceso sritį, todėl įgyvendinus tas praktikas, gauname, kad proceso sritis yra įgyvendinta. Naudodamiesi ryšiais ir paveldėjimais galime išsitraukti labai daug informacijos labai greitai ir mums nereikia žinoti daug detalių.

Aprašius šį žodyną galime greitai ir paprastai surasti mums rūpimus faktus apie žodyną. Pavyzdžiui norime surasti viską su kuo yra susijęs analitikas (angl. *Analyst*). Tam įvykdome užklausą, kuri pavaizduota 1 pav. ir gauname rezultatus, kurie atvaizduoti 2 lentelėje.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX cmmi: <http://www.semanticweb.org/aivaras/ontologies/2018/0/cmmi#>
SELECT ?predicate  ?object
        WHERE { cmmi:Analyst ?subject  ?a }
```

19 pav. SPARQL užklausos kodas

Įvykdžius šią užklausą mums pateikiami tokie rezultatai:

14 lentelė. SPARQL užklausos rezultatai

Kas	Ką veikia	Ką
Analitikas (angl. *Analyst*)	Identifikuoja (angl. *Identify)*	Užduočių priklausomybę (angl. *TaskDependencies)*
Analitikas (angl. *Analyst*)	Analizuoja (angl. *Analys)*	Projekto istorinius duomenis (angl. *Project Historical Data*)
Analitikas (angl. *Analyst*)	Identifikuoja (angl. *Identify)*	Pagrindinius etapus (angl. *Major Milestones*)
Analitikas (angl. *Analyst*)	Paruošia (angl. *Prepares*)	Evoliucinius priėmimo reikalavimus (angl. *Evolution Acceptance Requirements*)

Analitikas (angl. *Analyst*)	Nurodo (angl. *Determines*)	Techninius reikalavimus (angl. *Technic Approach*)
Analitikas (angl. *Analyst*)	Paruošia (angl. *Prepares*)	Reikalavimų kriterijus (angl. *Requirements Criteria*)
Analitikas (angl. *Analyst*)	Dokumentuoja (angl. *Documents*)	Reikalavimų pakeitimus (angl. *Requirements Changes*)
Ir t.t.	Ir t.t.	Ir t.t.

Jeigu norėtume surasti viską kas priklauso konkrečiai praktikai iš specifinės proceso srities, atliktume viską su paprasta paieška nurodydami, kad norime surasti visus duomenis, kurie priklauso specifinei praktikai.

Vertintojas žino kokios yra proceso sritys, kokie specifiniai tikslai ir specifinės praktikos. Žemiau pateiksime išnagrinėtus kelis variantus.

1. Žodyno naudotojas žino tiksliai specifinę praktiką, kuri jį dominą. Pavyzdžiui galime paimti projekto planavimo PP (angl. *Project Planning*) 2 specifinio tikslo – sukurti projekto planą (angl. *Develop a Project Plan*). Šio specifinio tikslo pasirenkame 1 specifinę praktiką – nustatyti biudžetą bei tvarkaraštį (angl. *Establish the Budget and Schedule*).

   20 pav. matome suformuluota SPARQL užklausą.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX cmmi: <http://www.semanticweb.org/aivaras/ontologies/2018/0/cmmi#>

SELECT Distinct ?subject   ?predicate ?object
        WHERE { ?subject ?class cmmi:SP_2.1_Establish_the_Budget_and_Schedule .
                ?object ?class cmmi:SP_2.1_Establish_the_Budget_and_Schedule .
                ?subject ?predicate?object
}
```

20 pav. SPARQL užklausa skirta surasti nurodytos specifinės praktikos trilypius ryšius

Atlikus tokią paiešką gauti rezultatai atvaizduoti 15 lentelėje apačioje.

15 lentelė. SPARQL užklausos rezultatai

Subjektas	Savybė	Objektas
Projekto vadovas (angl. *Project manager*)	Identifikuoja (angl. *Identify*)	Pagrindinius etapus (angl. *Major Milestones*)
Projekto vadovas (angl. *Project manager*)	Identifikuoja (angl. *Identify*)	Numatomas tvarkaraštis (angl. *ScheduleAssumptions*)
Projekto vadovas (angl. *Project manager*)	Identifikuoja (angl. *Identify*)	Apribojimus (angl. *Constraints*)
Analitikas (angl. *Analyst*)	Identifikuoja (angl. *Identify*)	Užduočių priklausomybė (angl. *TaskDependencies*)

2. Taip pat galime pasirinkti, kad mums išvestų apskritai visus trilypius ryšius, kurie priklauso tam tikrai proceso sričiai. Pavyzdžiui jeigu pasirinktume, kad mums išvestų visus trilypius ryšius, kurie atspindėtų kokie veiksmai turi būti atliekami organizacijoje, kad būtų užpildytas Reikalavimų valdymas (angl. *Requirements Management*). Iš pradžių turėtume atlikti 3 pav. pavaizduotą SPARQL užklausą.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX cmmi: <http://www.semanticweb.org/aivaras/ontologies/2018/0/cmmi#>

SELECT Distinct ?subject  ?x ?object
        WHERE { ?x ?class cmmi:SG_1_Manage_Requirements .
                        ?subject ?x ?object.
}
```

21 pav. SPARQL užklausa skirta surasti trilypius ryšius padengti REQM proceso sritį.

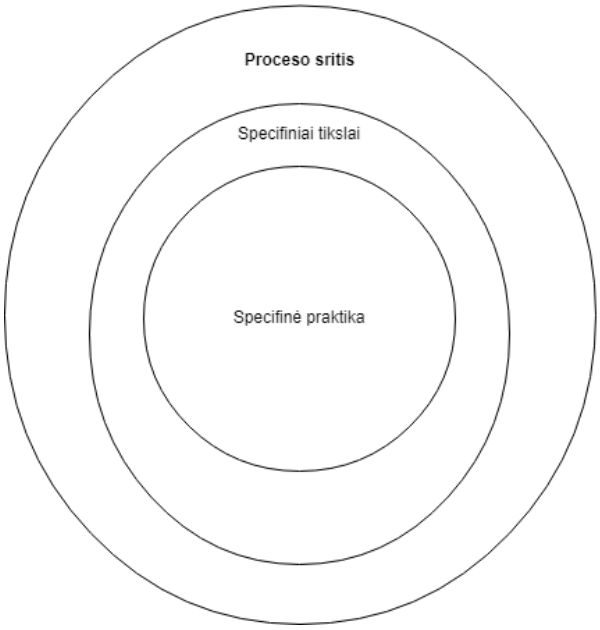
Atlikus tokią užklausą – gautume visus trilypius ryšius, kurie buvo sukurti aprašant šios proceso srities praktikas. Gautume identiškus rezultatus, kaip ir 13 lentelės pradžioje pateikti trilypiai ryšiai.

7. Praktinis pritaikytos ontologijos panaudojimus

Pasinaudojus žodynu ir apsirašius tam tikrą procesų vertinimo modelį - žymiai paprasčiau vertinti įmonės brandą. Šiame skyriuje pateiktos gairės, kaip įmonėje galėtų būti vertinama programų kūrimo procesų branda. Kaip pavyzdį naudosime CMMI modelį.

7.1. Įmonės vertinimas įprastu būdu pagal CMMI-DEV

Iki šiol įmonės vertinime buvo ganėtinai daug subjektyvumo. Pavyzdžiui, vertinant tam tikrą praktiką buvo vertinama kiek ji yra įgyvendinta, tačiau tokį įvertinimą atlikti yra labai sudėtinga. Net jeigu ir vertintojas yra labai patyręs ir kompetentingas, tokiame vertinime yra nemažai subjektyvumo. Kadangi praktikos yra vertinamos abstrakčiai.



22 pav. Įprasto (aukštesnio lygio) vertinimas

Aukščiau pateiktame 22 paveikslėlyje matome, jog proceso sritis sudaryta iš specifinių tikslų, kuriuos išpildo specifinės praktikos. Būtent tos praktikos ir yra įprastai vertinamos. Kaip jau buvo minėta anksčiau, CMMI vertinime yra skiriama procentai. Pagal juos yra matuojama kiek gerai įmonėje vykdomi konkretūs procesai. Būtent šioje vietoje ir yra sudėtinga įvertinti, nes nėra leidžiamasi dar giliau. Specifinės praktikos turi vidines praktikas. Kurių gali būti nebūtinai viena, o netgi dviženklis skaičius. Žemiau pateiktoje 16 lentelėje parodyta kaip vyksta vertinimas įprastai.

16 lentelė. Įprastas vertinimas įmonėje

GVM.1	**Enterprise Governance**						Purpose
							Outcomes
Attributes	Attribute Rating	Practice	Practice Rating	Project	Scale 0..100	NPLF	Evidence Detail
PA 1.1	81	GVM.1.BP1	33	P1	33	P	
				P2			
				P3			
		GVM.1.BP2	100	P1	100	F	
				P2			
				P3			
		GVM.1.BP3	67	P1	67	P	
				P2			
				P3			
		GVM.1.BP4	100	P1	100	F	
				P2			
				P3			
		GVM.1.BP5	67	P1	67	P	
				P2			
				P3			
		GVM.1.BP6	100	P1	100	F	
				P2			
				P3			
		GVM.1.BP7	100	P1	100	F	
				P2			
				P3			
		GVM.1.BP8	100	P1	100	F	
				P2			
				P3			
		GVM.1.BP9	67	P1	67	P	
				P2			
				P3			
		GVM.1.BP10	100	P1	100	F	
				P2			
				P3			

Lentelėje viskas vyksta taip, jog įmonėje yra vertinami keli projektai. Vertintojas pasvarstęs kiek čia procentų duoti pavyzdžiui projekto planavimui, priskiria procentus, kiek jam atrodo yra teisinga. Kompetentingas vertintojas tikrai gali gerai įvertinti. Tačiau tai vis tiek nėra labai konkretu ir objektyvu.

7.1.1. Konkretus pavyzdys ir iškylančios problemos

Proceso sritis: Konfigūracijos valdymas PP (angl. *Project planning*)

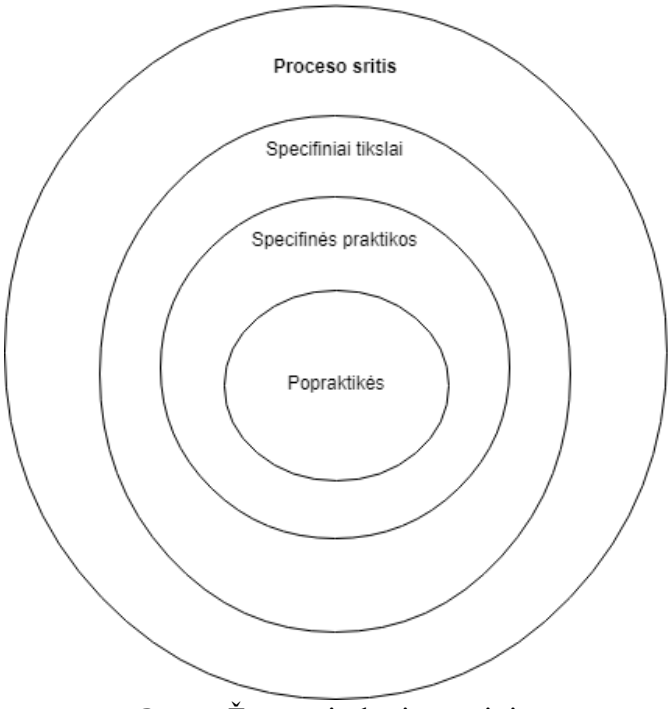
Žemiau 17 lentelėje pateiktas pavyzdys, kaip yra vertinama ir kokie iškyla klausimai žiūrint įverčius.

17 lentelė. Galimai iškylantys klausimai

Specifinis tikslas	Specifinė praktika	Įvertis	Kylantys klausimai/pamąstymai
SG 1. Nustatyti matavimus	SP 1. Nustatyti projekto apimtį	68%	• Kodėl tiek procentų? • Kas blogai? • Gal kažko neįgyvendiname apskritai?
	SP 2. Nustatyti įverčius	45%	

7.1. Galimas vertinimas

Vertinant įmonės procesų brandą žemesniame lygyje, būtų aišku kur daromos klaidos, ko trūksta. Pats vertinimas taptų objektyvesniu. Kadangi nėra sustojama praktikų lygyje. Yra gilinamasi į kiekvieną praktiką labiau, nei iki šiol. Žiūrima iš kokių popraktikių susideda pati praktika. Jos visos yra aprašomos ir žymimos tik įgyvendinta/neįgyvendinta. Tai padėtų sudaryti objektyvų vertinimą. Žemiau pateiktame paveikslėlyje matome papildomą lygį, kuriam skiriamas labai didelis dėmesys.



23 pav. Žemesnio lygio vertinimas

Gilinantis į šį lygį 18 lentelėje žemiau pateikiame pavyzdį kaip tai galėtų atrodyti.

18 lentelė.Galimai kylantys pasvarstymai vertinant žemesniu lygiu.

Specifinis tikslas	Specifinė praktika	Praktikos popraktikė	Įgyvendinta/ neįgyvendinta	Klausimai/pamąstymai
SG 1. Nustatyti matavimus	SP 1. Nustatyti projekto apimtį	Paruoši WBS	Taip	• WBS paruoštas
		Parengti projekto paketus, kad būtų galima nustatyti priklausomybes, atsakomybes, sudaryti tvarkaraštį.	ne	• Projekto paketai nėra paruošti
		Identifikuoti papildomai reikalingus produktus.	Taip	• Papildomai reikalingi produktai yra identifikuoti
		Identifikuoti darbo produktus, kurie galėtų būti perpanaudoti	Taip	• Darbo produktai nustatyti, aišku

				ką galima perpanaudoti.

Matome, kad pats vertinimas nuleistas iki tokio lygio, kur nereikia svarstyti kiek čia procentų parašyti. Todėl pats vertinimas tampa žymiai paprastesniu. O ir pašaliniams tampa aišku kas įgyvendinta ir kas ne.

Žemiau 19 lentelėje pateikiame pavyzdį, kaip galėtų būti įgyvendinamas pats įvertinimas įmonėje.

19 lentelė. Pavyzdys, kaip galėtų atrodyti vertinimas.

Specifiniai tikslai	**Galutinis įvertis**	**Specifinė praktika**	**Specifinės praktikos įvertis**	**Trilypis ryšys**	**Įvertis (True/False)**
PP 1		SP 1.1.	50	T1	T
				T2	F
		SP 1.2.	66	T1	T
				T2	F
				T3	T
		SP 1.3.	100	T1	T
	54	SP 1.4.	0	T1	F
PP 2		SP 2.1.	75	T1	T
				T2	T
				T3	T
				T4	F
		SP 2.2.	50	T1	F
				T2	T
		SP 2.3.	100	T1	T
		SP 2.4.	0	T1	F
		SP 2.5.	100	T1	T
		SP 2.6.	100	T1	T
	75	SP 2.7.	100	T1	T
P3	29.3	SP 3.1.	0	T1	F

				T1	F
		SP 3.2.	50	T2	T
				T1	F
				T2	T
		SP 3.3.	33	T3	F

Taip pat toks vertinimas yra labai patogus įmonėje, nes patį įmonė gali save įsivertinti ir žinoti kokį lygi atitinka. Kadangi pats svarbiausias dalykas, kad nereikia spėlioti kiek čia procentų praktika yra įgyvendinta. Yra labai aiškiai apibrėžti veiksmai, kurie gauna įverti vykdoma/nevykdoma. Tai suteikia galimybę viską vertinti objektyviai.

Rezultatai ir išvados

Šiame darbe buvo sukurta procesų vertinimo modelių ontologijos metmenis, kurią naudojantis galima aprašyti konkrečius procesų vertinimo modelius. Jeigu ontologijoje nėra reikiamų sąvokų ar ryšių – ją galima praplėsti.

Darbo metu buvo pasiekti šie rezultatai:

- Sukurta procesų vertinimo modelių ontologija;
- Ontologija pritaikyta praktiškai su CMMI-DEV 1.3 modelio tam tikromis procesų sritimis;
- Buvo atliktos įvairios SPARQL paieškos, norint greitai ir efektyviai sužinoti įvairios informacijos apie praktikas ir jų išpildymą;
- Pateiktos įžvalgos, kaip tai galėtų palengvinti programų kūrimo procesų vertinimą įmonėse.

Gavus šiuos rezultatus galima formuluoti šias išvadas:

- Sukūrus ontologiją, ji mums leidžia aprašyti konkrečius procesų vertinimo modelius.
- Įmanoma pagal sukurtą bendrą procesų vertinimo ontologiją aprašyti konkrečius procesų vertinimo modelius.
- Ontologija būtų viešai prieinama, todėl bet kuris vertintojas galėtų ja naudotis, esant reikalui praplėsti.

Procesų vertinimo modelių ontologija mums leidžia įgyvendinti įvairius procesų vertinimo modelius. Juos aprašius naudojantis šia ontologija, galime naudotis SPARQL užklausomis. Šių užklausų dėka galima lengvai surasti kokie veiksmai turi būti atliekami su konkrečia praktika. Lengvai surasti silpnąsias vietas ir sužinoti kurioje vietoje įmonė yra silpna. Tai palengvina vertintojų darbą. Vertinimas tampa objektyvesnis.

Šaltiniai

[BGH02] Sean Bechhofer, Carole Goble, Ian Horrocks, „Requirements of Ontology Languages“, 2002 m.

[CM01] EXTREME CHAOS 2001, 2001.
[žiūrėta 2018-05-01]. Prieiga per internetą:
http://www.cin.ufpe.br/~gmp/docs/papers/extreme_chaos2001.pdf

[CM13] CHAOS MANIFESTO 2013, 2013.

[žiūrėta 2018-05-01]. Prieiga per internetą:
https://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf

[CM15] CHAOS MANIFESTO 2015, 2015.

[žiūrėta 2018-05-01]. Prieiga per internetą: https://www.infoq.com/articles/standish-chaos-2015

[ES07] J. Euzenat and P. Shvaiko. Ontology matching. Springer, 2007.

[FHH+12] Conrad Bock, Achile Fokoue, Peter Haase, Rinke Hoekstra, Ian Horrocks, Alan Ruttenberg, Uli Sattler, Michael Smith „OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax (Second Edition)“, 2012. [žiūrėta 2018-05-05]. Prieiga per internetą: https://www.w3.org/2012/pdf/REC-owl2-syntax-20121211.pdf

[GSO+11] William Greenly, Charles Sandeman-Cralk, Yago Otero, John Streit „Contextual Search for Volkswagen and the Automotive Indusrty“, 2011. [žiūrėta 2018-05-05]. Prieiga per internetą: https://www.w3.org/2001/sw/sweo/public/UseCases/Volkswagen/Volkswagen.pdf

[GST11] Sema Gazel, Ebru Akcapinar Sezar, Ayca Tarahan. An Ontology Based Infrastructure To Support CMMIBased Software Process Assessment, 2011.

[Gru09] Tom Gruber. Encyclopedia of Database Systems, Ling Liu and M. Tamer Özsu (Eds.), Springer-Verlag, 2009.

[Gua98] Nicola Guarino „Formal Ontology and Information Systems“, 1998

[Gui05] Giancarlo Guizzardi. Ontological foundations for structural conceptual models, 2005, pp. 65-70.

[HEP11] Martin Hepp „GoodRelations: An Ontology for Describing Products and Services Offers on the Web“, 2011. [žiūrėta 2018-05-05]. Prieiga per internetą: http://www.heppnetz.de/files/GoodRelationsEKAW2008-crc-final.pdf

[Hum93] Watts S. Humphrey. Introduction to Software Process Improvement, 1993.

[žiūrėta 2018-05-01]. Prieiga per internetą: http://repository.cmu.edu/cgi/viewcontent.cgi?article=1170&context=sei

[Hor11] Matthew Horridge, A Practical Guide To Building OWL Ontologies Using Protege 4 and CO-ODE Tools Edition 1.3, 2011. [žiūrėta 2017-05-15]. Prieiga per internetą: https://mariaiulianadascalu.files.wordpress.com/2014/02/owl-cs-manchester-ac-uk_-eowltutorialp4_v1_3.pdf

[HW04] K. Hyde and D. Wilson. Intangible benefits of CMM-based software

process improvement, 2004, pp.217–228.

[ISO07] „ISO/IEC 15504-7:2007 Information technology Process assessment Part 7: Assessment of Organizational Maturity” 2007.

[JTC12] Jungtinis techninis komitetas ISO/IEC JTC 1. Information technology. Process assessment. An exemplar software life cycle process assessment model, 2012.

[žiūrėta 2018-05-01]. Prieiga per internetą:

[LÖ08] Ling Liu and M. Tamer Özsu (Eds.), „Encyclopedia of Database Systems“ 2008.

http://www.iso.org/iso/catalogue_detail.htm?csnumber=60555

[Mcb04] Brian McBride. Handbook on ontologies. The Resource Description Framework (RDF) and its Vocabulary Description Language RDFS, 51-52 psl. [žiūrėta 2017-05-15]. Prieiga per internetą: http://link.springer.com/chapter/10.1007/978-3-540-24750-0_3

[MCK04] Graham Klyne, Jeremy J. Carrroll, Brian McBride. RDF 1.1 Concepts and Abstract Syntax, 2004 m. [žiūrėta 2018-05-01]. Prieiga per internetą:

http://www.w3.org/TR/rdf11-concepts/

[Mcq12] P. A. McQuaid. Software disasters—understanding the past, to improve the future. Journal of Software: Evolution and Process, 2012 , pp.459–-470.

[Mea67] S. H. Mealy. Foundation of data modeling. 1967 m.

[Mit17] Antanas Mitašiūnas, Programų kūrimo proceso vertinimas ir gerinimas, Vilnius, 2017

[Miz11] Riichiro Mizoguchi, „Tutorial on ontological engineering”, 2011 m. [žiūrėta 2018-05-01]. Prieiga per internetą:

http://www.unipamplona.edu.co/unipamplona/portalIG/home_23/recursos/general/06032011/onto_parte1.pdf

[ORG15] Lorena Otero-Cerdeira, , Francisco J. Rodríguez-Martínez , Alma Gómez-Rodríguez. Ontology matching: A literature review 2015, 2015.

[Pau09] Mark C. Paulk „A History of the Capability Maturity Model for software“, 2009. [žiūrėta 2018-05-01]. Prieiga per internetą: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.216.199&rep=rep1&type=pdf

[Pel14] Stasys Peldžius „Programų kūrimo procesų vertinimas, naudojant keletą procesų vertinimo modelių”, 2014 m.

[PPG+10] C. Pardo, F. J. Pino, F. García, M. P. Velthius, and M. T. Baldassarre, “Trends in Harmonization of Multiple Reference Models,” in 5th International Conference, ENASE 2010, Athens, Greece, July 22-24, 2010, Revised Selected Papers, 2010, no. 4, pp. 61–73.

[PR12] Stasys Peldžius, Saulius Ragaišis, Framework for Usage of Multiple Software Process Models, Software process improvement and capability determination: 12th International Conference, SPICE 2012, Palma, Spain, May 29-31, 2012, p. 210-221.

[PR13] S. Peldzius, S. Ragaisis. Usage of Multiple Process Assessment Models. Software Process Improvement and Capability Determination: 13th International Conference, SPICE 2013, Bremen, Germany, June 04 - 06, 2013: Proceedings. Series: Communications in Computer and Information Science p. 223-234.

[PWC+93] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, Charles V. Weber, „Capability Maturity Model for Software V1.1“, 1993 m. [žiūrėta 2018-05-01]. Prieiga per internetą: http://www.sei.cmu.edu/reports/93tr024.pdf

[Rag07] Saulius Ragaišis, Programų sistemų inžinerija, Mokymo medžiaga, Vilnius, 2007.

[SEI02] CMMI 1.1 Continous Representation, Software Engineering Institute, 2002, 724 p.

[SEI10] CMMI® for Development, Version 1.3, 2010. [žiūrėta 2018-05-01]. Prieiga per internetą: http://www.sei.cmu.edu/reports/10tr033.pdf

[SMN+11] Robert Stanley, Bruce McManus, Raymond Ng, Erich Gombocz, Jason Eshleman, Charles Rockey „Applied Semantic Knowledgebase for Detection of Patients at Risk of Organ Failure through Immune Rejection“, 2011. [žiūrėta 2018-01-05]. Prieiga per internetą: https://www.w3.org/2001/sw/sweo/public/UseCases/IOInformatics/IOInformatics.pdf

[SK12] Gokhan Halit Soydan, Mieczyslaw M. Kokar. A Partial Formalization of the CMMI-DEV—A Capability Maturity Model for Development, 2012.

[SP91] S. Spaccapietra and C. Parent. Conflicts and correspondence assertions in interoperable databases, 1991, pp.49–54.

[WWWC03] World Wide Web Consortium, OWL Web Ontology Language Reference, 2004 m. [žiūrėta 2018-05-01]. Prieiga per internetą: http://www.w3.org/TR/2004/REC-owl-ref-20040210/

[WWWC04b] World Wide Web Consortium. RDF 1.1 Concepts and Abstract Syntax, 2014 m. [žiūrėta 2018-05-01]. Prieiga per internetą: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225

Sąvokų apibrėžimai

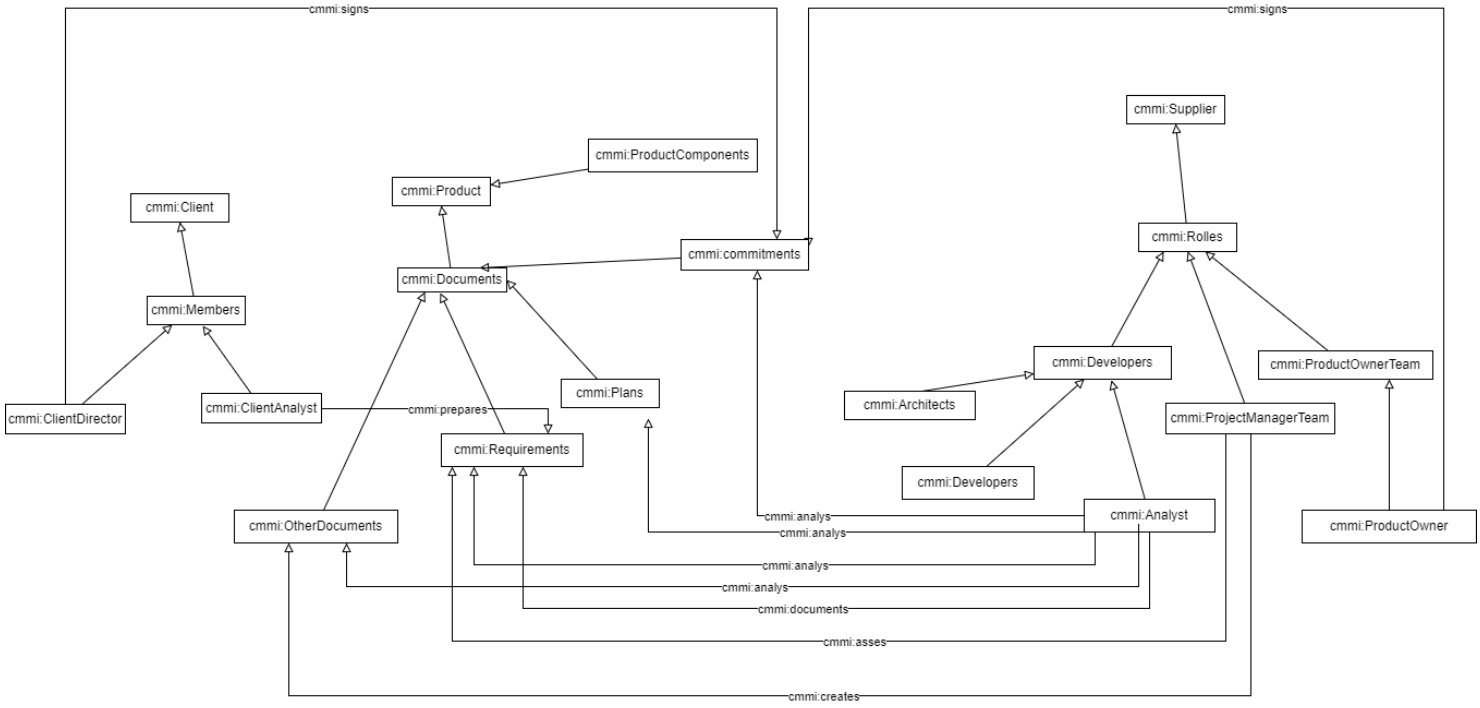
Procesų vertinimo modelis	Modelis, skirtas organizacijos procesų gebėjimo ar visuminio proceso brandos vertinimui.
Proceso branda	Charakteristika, nusakanti, kiek visuminis procesas yra valdomas, apibrėžtas, matuojamas, kontroliuojamas ir nuolatos gerinamas.
Proceso gebėjimo lygis	Įvertis diskrečioje skalėje, nusakantis proceso gebėjimą.
Proceso brandos lygis	Aiškiai apibrėžta pakopa visuminio proceso brandos evoliucijoje.
Proceso gebėjimas	Charakteristika, nusakanti rezultatų, kuriuos galima gauti taikant procesą, pasiskirstymą

Santrumpos

OWL	Žiniatinklio ontologijų kalba (angl. *Web Ontology Language*)
CMMI-DEV	Integruotas gebėjimo brandos modelis (angl. *Capability Maturity Model Integration*).
ISO/IEC 15504-7	Brandos vertinimo modelis (angl. *Assessment of Organizational Maturity*)
SPI	Programinių procesų gerinimas (angl. *Software Process Improvement*)
TPAM	Tarpinis procesų vertinimo modelis
PAM	Procesų vertinimo modelis

Priedai

1 priedas. Svarbiausia dalis CMMI žodyno UML klasių diagramos



2 priedas. Sugeneruoto kodo iškarpa

```
<owl:ObjectProperty rdf:about="http://www.semanticweb.org/cmmiDEVOntology#pre-
pares">
        <rdfs:domain rdf:resource="http://www.semanticweb.org/cmmiDEVOntol-
ogy#ClientAnalyst"/>
        <rdfs:domain rdf:resource="http://www.semanticweb.org/cmmiDEVOntol-
ogy#IntegrationLead"/>
        <rdfs:range rdf:resource="http://www.semanticweb.org/cmmiDEVOntol-
ogy#ClientRequirements"/>
        <rdfs:range rdf:resource="http://www.semanticweb.org/cmmiDEVOntol-
ogy#IntegrationPlan"/>
    </owl:ObjectProperty>

    <!--  Classes     -->

    <!-- http://www.semanticweb.org/cmmiDEVOntology#Analyst -->

    <owl:Class rdf:about="http://www.semanticweb.org/cmmiDEVOntology#Analyst">
        <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/cmmiDEVOn-
tology#DeveloperTeam"/>
    </owl:Class>

    <!-- http://www.semanticweb.org/cmmiDEVOntology#Architects -->

    <owl:Class rdf:about="http://www.semanticweb.org/cmmiDEVOntology#Archi-
tects">
        <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/cmmiDEVOn-
tology#DeveloperTeam"/>
    </owl:Class>

    <!-- http://www.semanticweb.org/cmmiDEVOntology#ClientAnalyst -->

    <owl:Class rdf:about="http://www.semanticweb.org/cmmiDEVOntology#ClientAn-
alyst">
        <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/cmmiDEVOn-
tology#Members"/>
    </owl:Class>

    <!-- http://www.semanticweb.org/cmmiDEVOntology#ClientCompany -->

    <owl:Class rdf:about="http://www.semanticweb.org/cmmiDEVOntology#Client-
Company"/>

    <!-- http://www.semanticweb.org/cmmiDEVOntology#ClientRequirements -->

    <owl:Class rdf:about="http://www.semanticweb.org/cmmiDEVOntology#ClientRe-
quirements">
        <rdfs:subClassOf rdf:resource="http://www.semanticweb.org/cmmiDEVOn-
tology#Documents"/>
```