TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Jane-Ly Buhvestova 206132IABB

Osapoole ja Osapoole Seose arhetüüpmustrite realiseerimine, testimine ja sobivuse hindamine ISO 13940:2015 (ContSys) mõistete deklaratiivseks spetsifitseerimiseks

Bakalaureusetöö

Juhendaja: Toomas Klementi
MSc
Kaasjuhendaja: Gunnar Piho
PhD

Tallinn 2023

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Jane-Ly Buhvestova

17.05.2023

Annotatsioon

Meditsiiniinformaatika ContSys standardis defineeritud mõistete eesmärk on saavutada masinloetav semantika, et meditsiiniinfosüsteemid toetaksid tervise järjepidevuse tagamist. Käesoleva töö eesmärk on Osapoole ja Osapoole Seose arhetüüpmustrite realiseerimisel ja testimisel ABC4HEDA-s hinnata nende sobivust ContSys mõistete deklaratiivseks spetsifitseerimiseks. Varasemalt on termineid tõlgendatud protsesside perspektiivist, mis on tihedalt seotud Osapoole arhetüüpidega – äriprotsess toetub osapooltevahelistele seostele. Hinnates ContSys termineid kitsendatult osapoole arhetüüpidena, antakse ülevaade osapoole ja selle seostena esitatavatest terminitest. Lisaks ontoloogilisele lähenemisele hinnatakse ABC4HEDA üleminekut MVC mudelilt SPA-le, seda seejuures täiendades testidega.

Vastavalt töötulemustele kinnitab käesoleva töö autor ContSysi potentsiaali üheselt mõistetava semantika loomiseks meditsiiniinfosüsteemides, juhtides tähelepanu hetkel semantilist koostalitlusvõimet pärssivad mitme arhetüübina tõlgendatavad mõisted. Käesolev töö autor peab oluliseks edasistes töödes tulemuste valideerimist meditsiinitöötajatega.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 30 leheküljel, 5 peatükki, 18 joonist, 10 tabelit.

Abstract

Realization, testing and suitability assessment of Party and Party Relationship archetype patterns for the declarative specification of ISO 13940:2015 (ContSys) concepts

The purpose of the ContSys standard is to achieve machine-readable semantics so that healthcare information systems can support the continuity of care. The aim of this work is to evaluate the suitability of the Party and Party Relationship archetype patterns in the realization and testing of ABC4HEDA for the declarative specification of ContSys concepts. Previously, the terms have been interpreted from a process perspective, closely related to Party archetypes – business processes rely on interparty relationships. By evaluating ContSys terms strictly as Party archetypes, an overview of the terms presented as Parties and Party Relationships is provided. In addition to the ontological approach, the transition of ABC4HEDA from the MVC model to SPA is evaluated, supplemented by tests.

Based on the results of this work, the author confirms the potential of ContSys to create unambiguous semantics in healthcare information systems, while drawing attention to the concepts that are currently interpreted as multiple archetypes, inhibiting semantic interoperability. The author of this work considers it important to validate the results with healthcare professionals in further studies.

The thesis is in Estonian and contains 30 pages of text, 5 chapters, 18 figures, 10 tables.

Lühendite ja mõistete sõnastik

ABC4HEDA	*Archetypes Based Computing for Health Data* ehk arhetüüpidel ja arhetüüpmustritel põhinev terviseandmete mudel
API	*Application Programming Interface* ehk rakendusliides
Arhetüüp	Süsteemi mõne olulise komponendi mudel (näiteks osapool, roll, protsess, tellimus, reegel)
ContSys	*A system of concepts for the continuity of care* (ISO 13940:2015) ehk mõistesüsteem tervishoiu ja arstiabi järjepidevuse toetamiseks
ISO	*International Organization for Standardization* ehk Rahvusvaheline Standardiorganisatsioon
MVC	*Model-View-Controller* ehk Mudel-Vaade-Kontroller
Ontoloogia	Tehniline termin tähistamaks artefakti, mis võimaldab modelleerida teadmisi mõne reaalse valdkonna kohta
Sematiline	Tähenduslik
SPA	*Single Page Application* ehk üheleheline rakendus
UML	*Unified Modeling Language* ehk ühtne visualiseerimiskeel

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

1.1 ContSys roll meditsiiniinformaatikas

Meditsiiniinformaatika on üks kiiremini kasvavaid tööstusharusid [1], mis juhib tähelepanu muuhulgas tervishoiu infosüsteemide koostalitlusvõime standarditele. Tervishoiu ja arstiabi järjepidevuse põhinõue [2] on ontoloogial põhinev masintõlgendatav semantika, mis soodustab heterogeensete infosüsteemide tõhusat koostalitlusvõimet [3] ning seeläbi tõstab tervishoiu kvaliteeti ja ohutust [2]. Semantilist koostalitlusvõimet toetab rahvusvaheline meditsiiniinformaatika standard, ContSys [4], aastast 2015, mille eestikeelne versioon publitseeriti 2023. aasta veebruaris.

ContSys standardi mõistete eesmärk on patsiendikeskse tervishoiu järjepidevuse toetamine kliiniliste protsesside keskel, võimaldades terviseteabe kasutamist ka muudel eesmärkidel (näiteks teadmiste halduses, teiseseks kasutamiseks järelkontrollis). ContSys eesmärk on olla tervishoiuorganisatsioonide kõigi tasandite koostalitlusvõime ja tervishoiu infosüsteemide arendamise vundament, et luua terviklik kontseptuaalne alus tervishoiuteenuste sisule ja kontekstile, arendades ärimudeleid, infosüsteeme või struktureeritud teavet kindlaksmääaratud kliiniliste protsesside jaoks sõltumata riiklikest, kultuurilistest ja ametialastest takistustest [2].

1.2 Probleem

Ontoloogia üks osa on ärikontekstile vastavate mõistete kujundamine [3] infosüsteemis, kusjuures selleks vajalikud terminid peaksid olema kooskõlas vastava valdkonna spetsialistide pädevusküsimustega [5], mida pakub näiteks ContSys. Domeenispetsiifilist keelt meditsiinistandardite ja protokollide deklaratiivseks täpsustamiseks üheselt mõistetaval ja masinloetaval kujul kirjeldab arhetüüpidel ja arhetüüpmustritel põhinev meta-mudel, milles hoiab andmeid töös kasutatav ABC4HEDA [6]. Arhetüübid ja arhetüüpmustrid võimaldavad tagada mudeli korduvkasutatavuse ja automatiseerituse tervishoiu järjepidevuse saavutamiseks. ABC4HEDA [7] valdkonnamudeli sobivust on hinnatud lähtuvalt LOINC [8], HL7 FHIR [9] (publitseeritud HEDA 2022 [10]

konverentsil) ja ContSys protsessimustrist [6] (publitseeritud HEDA 2022 [11] ja MODELSWARD 2023 [12] konverentsidel).

Patsiendikeskse tervishoiu tagamiseks on oluline Osapoole ja Osapoole Seose mõistmine, kuid seni pole hinnatud ABC4HEDA mudeli nimetatud arhetüüpmustrite sobivust ContSys standardis defineeritud mõistete [13] üksikasjalikuks ja selgitavaks esitamiseks meditsiiniinfosüsteemis.

1.3 Eesmärk

Käesoleva töö peamiseks eesmärgiks on hinnata ABC4HEDA mudeli sobivust ContSys standardis defineeritud terminite deklaratiivseks spetsifitseerimiseks vastavalt Osapoole (*Party*) ja Osapoole Seose (*Party Relationship*) arhetüübimustritele. Töö kaasnevaks eesmärgiks on seni mittetäielikult realiseeritud ABC4HEDA Osapoole ja Osapoole Seose arhetüüpmustri täielik realiseerimine ja testimine ning senise MVC arhitektuurilt üleminek SPA arhitektuurse toe lisamisele. Viimase puhul peetakse silmas andmete liikuvust serveri ja kliendirakenduse vahel API kaudu ning ABC4HEDA mudeli kasutatavuse hindamist SPA raamistikes. ABC4HEDA tehniline arendamine on vajalik terminite tõlgendamisel mõistmaks arhetüüpide võimalusi, mis omakorda võimaldab ContSys mõistete semantilisuse tagamisel need kerge vaevaga infosüsteemi sisestada ja järjepidevust toetav meditsiiniinfosüsteem on juba töökorras. Sellest tulenevalt tulemuseks on valdkonnapõhine Osapoole ja Osapoole Seoste deklaratiivne ja käitlusajal kasutatav spetsifitseerimise keel, mille sobivust meditsiini- ja tervishoiuvaldkonnas hinnatakse lähtuvalt ContSys standardist.

1.4 Töö struktuur

Töö on jaotatud neljaks osaks, millest käesolevas peatükis antakse ülevaade ContSysi aktuaalsusest meditsiiniinformaatikast, sellega seotud lahendamist vajavast probleemist ning eesmärgist, mida soovitakse lõputöö kirjutamise käigus saavutada. Metoodikas kirjeldatakse lühidalt projekti objekti ehk Osapoole ja Osapoole Seose arhetüüpmustrit, töö käigus kasutatud tööriistu ning tööprotsessi. Kolmas peatükk keskendub väljapakutud lahenduse esitlemisel ja selgitamisel. Viimases sisulises peatükis esitatakse analüüs ja hinnang tehtud tööle, selgitades kasutatud mustreid ja meetrikaid, ning andes ülevaate töös tehtud kitsendustest ja edasistest töödest.

2 Metoodika

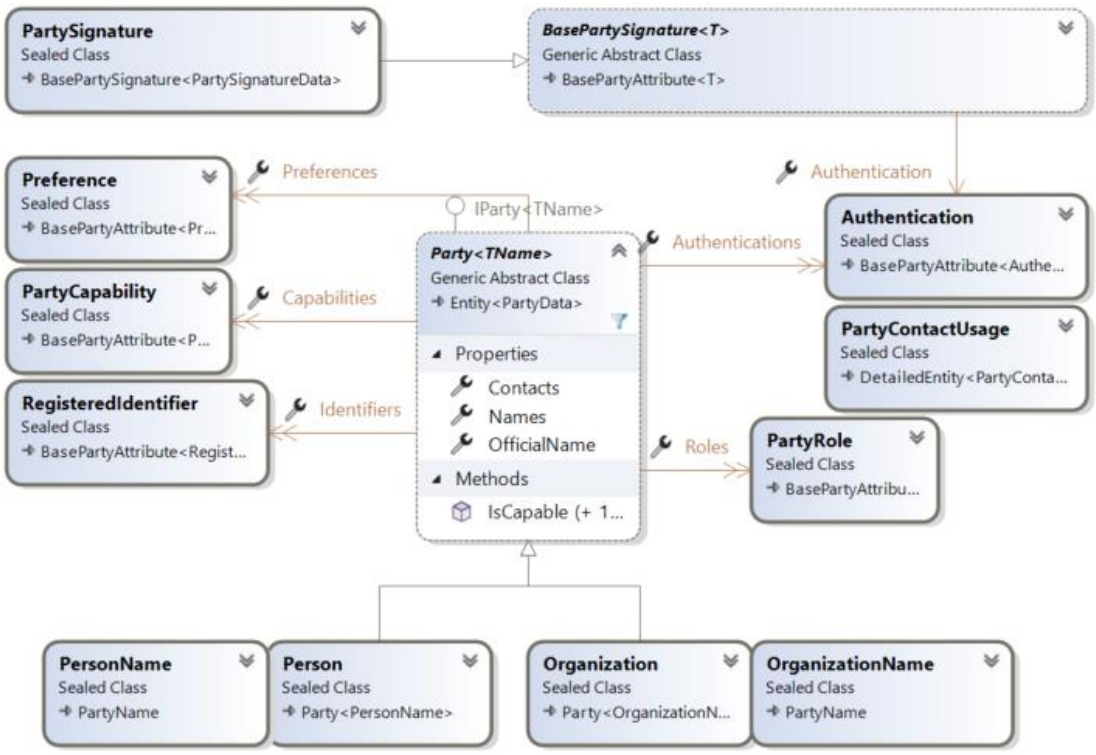
2.1 ABC4HEDA tarkvaramudel

ABC4HEDA tarkvaramudel baseerub Arlow & Neustadti mudelipõhisel arhitektuuril [14], mille kavandamisel on arvestatud Zacmani raamistikuga, mille kohaselt tuleb süsteemide projekteerimisel teada, miks, kuidas, kus, kes, millal ja miks midagi tehakse. On leitud, et Zachmani raamistiku sidumine Arlow & Neustadti arhetüüpide ja arhetüüpmustritega aitab paremini mõista ärivaldkonda, kujundada kulutõhusamat ärirakendust ja süstematiseerida seeläbi arhetüüpsete komponentide taaskasutamist rakendusesiseselt [14].

2.2 Osapoole arhetüüpmuster

Osapoole arhetüübi muster, mis esindab Arlow & Neustadti käsitluse kohaselt isikut või organisatsiooni, kirjeldab olulise teabe esitamist, säilitamist ja osapoole rolle erinevates osapooltevahelistes seostes [15]. See eeldab, et näiteks kliinilistes protsessides esitatakse osapoolega seotud teavet standardsel ja ühtsel viisil (Joonis 1). Osapoolega saab siduda eelistusi (*Preference*), pädevusi (*PartyCapability*), rolle (*PartyRole*) ning tema autentimisi (*Authentications*), eriti kui osapool soovib midagi allkirjastada (*PartySignature*).

Osapoole arhetüüpmustri konkreetsus ja kasutatavus erinevates ärivaldkondades on lihtne oma ülesehituselt ja põhimõttelt [15], kuid selle kasutuselevõtt ja juurutamine tervishoius on osutunud väljakutseks. Vastavalt ContSysi erinevatele terminitele ei ole pelgalt mõistete pealevaatamisele selge, kas tegu on või kindlasti ei ole Osapoolega.

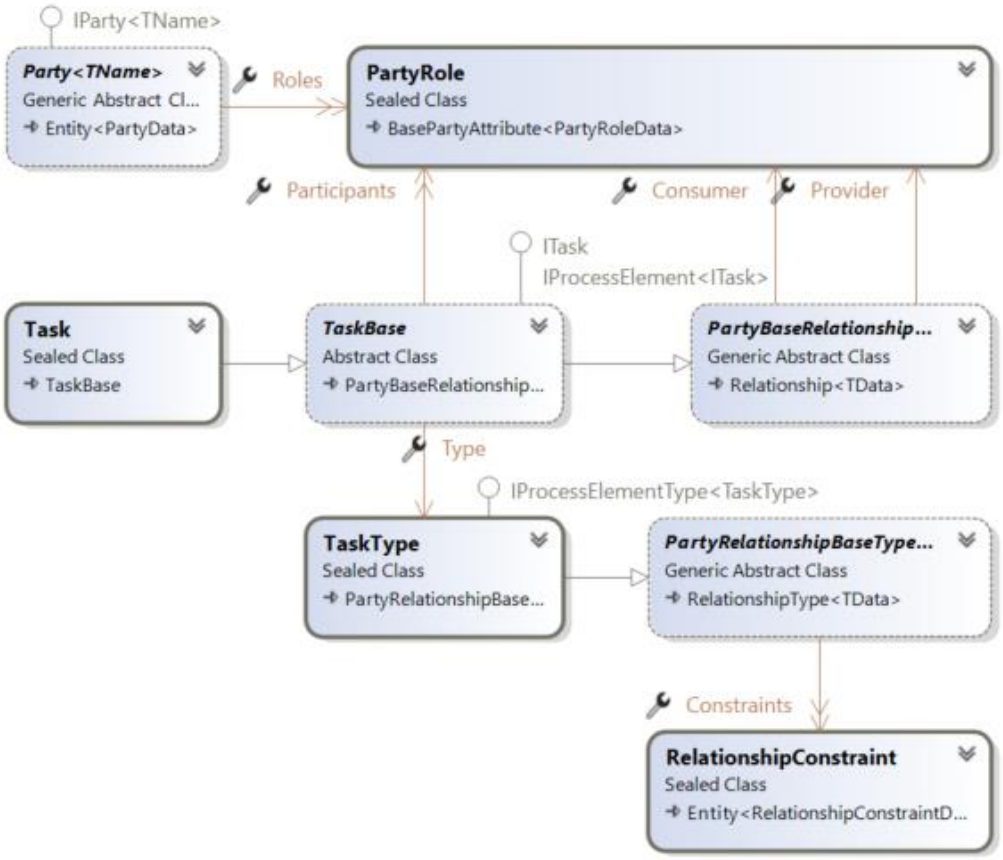


Joonis 1. Osapoole (*Party*) arhetüüpmuster.

2.3 Osapoole Seoste arhetüüpmuster

Osapooli saab modelleerida keeruliste ja abstraktsete omavaheliste seostena (*Party Relationship*). Igat osapoolt ühendab teistega teatud arv suhteid, kuid objektorienteeritud arenduse seisukohalt on vaja välja selgitada ärinõuetest lähtuvalt vaid äriinfosüsteemide jaoks olulised osapoolte seosed [15]. Näiteks verekeskuse seos haiglatega (doonori vere müümine [16]) on olulise tähtsusega, kuid doonori seos haiglaga ei ole. Kui doonor külastab sama haigla erakorralise meditsiiini osakonda, tekib nende vahel relevantne seos.

Osapoole seostele on ülesehitatud ka protsessimuster [6]. Iga protsessi tööülesande (*TaskBase*) täitmiseks on määratud konkreetsed osapoole rollid (*PartyRole*). Samaaegselt võib tööülesanne siduda osapooled (*PartyRelationship*) läbi protsessilõimede (Joonis 2). Näiteks patsient pöördub arsti poole neerupuudulikkusega, millega tegelemisel avastatakse südamepuudulikkus, mida tuleks hakata paralleelselt ravima. Sellisel juhul on raviperiood (*period of healthcare*) protsess, mis koosneb erinevatest raviprotsessiõimedest ühe terviseloo (*professional health record*) raames ning puudutab erinevaid osapooli erinevates seostes.



Joonis 2. Osapoolte Seoste (*Party Relationship*) seos ülesannetega protsessi vaates.

2.4 Kasutatud tööriistad

Projekti arendamiseks kasutati *Microsoft Visual Studio Community* 2022 17.5.5 versioonil põhinevat arenduskeskkonda, mis võimaldas käesolevas töös ABC4HEDA mudelil põhineva veebirakenduse vabavaralist arendamist C# keeles, testide jooksutamist ja käivitamist virtuaalmasinas. Kasutajaliidese arendamiseks kasutati .NET arendajate seas populaarsust koguvat *Blazor* veebiraamistiku, täpsemalt *Radzen Blazor* komponente, mis on avatud lähtekoodiga ja äriliseks kasutamiseks tasuta. Need on implementeeritud C#´s, sõltumata *JavaScript* raamistikest ega teekidest, ning kasutavad täielikult ära *Blazor* raamistikku, toetades nii serveri- kui kliendipoolset *Blazor´*it [17].

ASP.NET Core veebiraamistik võimaldas luua pilvetoega internetiühendusega rakendust [18], mille versioonihaldus toimus *Microsoft Azure´*i *Git*-teenuse *DevOps* kaudu. *Microsoft SQL Server* andmebaasi kasutades toetuti *Entity Framework Core´*ile, et võimaldada andmebaasile juurdepääsu .NET objekte kasutades [19].

2.5 Tööprotsess

Arendusprotsess toimus puhta koodi nõudeid [20] järgides, et tagada tuleviku huvides ABC4HEDA koodi läbipaistvus, lihtsasti jälgitavus ning järjepidevus. Kvaliteedi ja nõuetele vastavuse tagamisel oli oluline roll testipõhisel arendusel (*Test-Driven Development,* edaspidi TDD), mille keskseks eesmärgiks on samuti puhas töötav kood. TDD eeldab lisaks koodi dubleerimise eemaldamisele ja automaatse testi ebaõnnestumisest tingitud uue koodi kirjutamisele ka ise testide kirjutamist, mitte sõltumist kolmanda osapoole testide kirjutamise kiirusest [21]. Arvestades tervishoiu nõuete kiiresti muutuvust arendati projekti lähtuvalt ekstreemprogrammeerimise põhimõtetele, et arendusprotsess oleks avatud pidevatele võimalikele muudatustele säilitades ajaefektiivsuse [22].

3 Peamised tulemused

Lähtuvalt püsitatud eesmärgist antakse järgnevalt ülevaade ABC4HEDA edasiarenduse (sealhulgas testimise) sisulistest aspektidest ning ContSys terminite deklaratiivse spetsifitseerimise tulemustest.

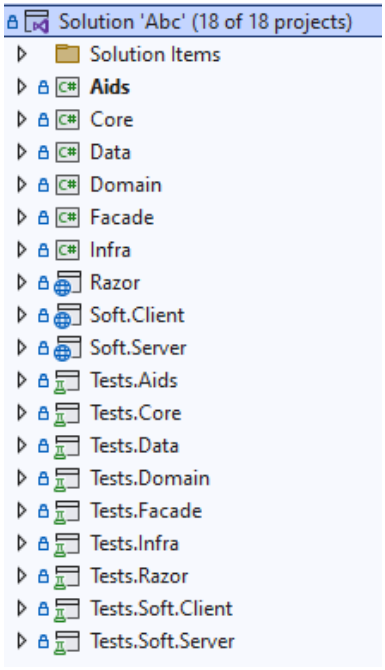
3.1 ABC4HEDA SPA arhitektuur

3.1.1 ABC4HEDA üleminek SPA arhitektuurile

Käesoleva töö üks eesmärkidest oli töö algushetkel *ASP.NET Core MVC* raamistikul baseeruva ABC4HEDA teisendamine SPA, täpsemalt *Blazor´*i, peale katsetuse eesmärgil. *Blazor* on üks SPA implementatsioonidest, nagu ka näiteks *AngularJS*.

SPA raamistikus töötab rakendus ühe veebilehena, mille puhul on kogu rakenduse esitluskiht serverist välja võetud ja seda hallatakse brauserist. SPA laeb eelnevalt üks kord alla veebilehe struktuuri, et vältida serverist uue lehe hankimiseks vajaminevat serveripäringut, võimaldades dünaamiliselt „ümber joonistada“ teatud ekraaniosi, mille värskendust näeb kasutaja koheselt. Pärast esmast lehe laadimist saadetakse ja võetakse serverist vastu ainult andmeid [23].

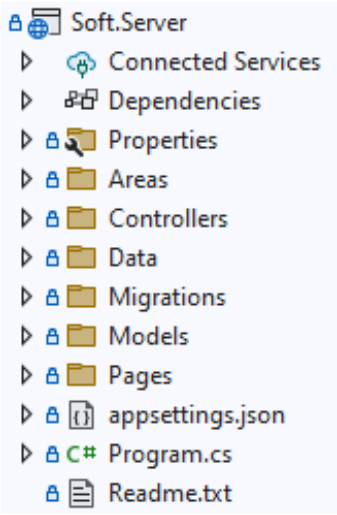
Projekt toetub üheksakihilise arhitektuuri struktuurile, mille eesmärk on hoida lahus andme-, loogika- ning esitluskiht. Andmekiht koosneb *Data* kihis asuvate andmeklassidest, mis ei sõltu ühestki raamistikuspetsiifilisest baasklassist (*Plain Old CLR Objects* ehk POCO) ning *Infra* kihis andmetele ligipääsudest, kus moodustatakse nende andmete põhjal ka andmebaasitabelid. Esitluskiht koosneb vaadetest (*Facade*), millest saadud informatsioon saadetakse vastavalt äriloogikale (*Domain*) andmebaasi talletamiseks. *Aids* ja *Core* tagavad kogu projektis läbivalt kasutatavate tööriistade hoiustamise ning *Razor* kuvatavate lehekülgede CRUD toimingute võimaldamise eest. Kasutajaliidese osad *Soft.Server* (kus asuvad kontrollerid) vastutab andmepäringute töötlemise eest (*back-end*) ning *Soft.Client* nende andmetel põhineva veebilehe võimalikult kasutajasõbraliku kuvamise (*front-end*) eest (Joonis 3).



Joonis 3. ABC4HEDA arhitektuur.

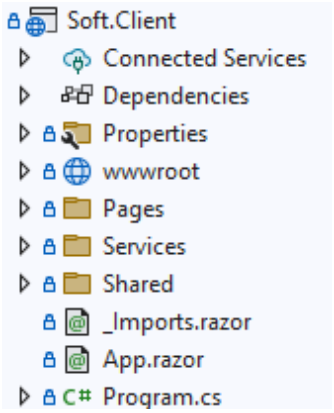
3.1.2 Blazor implementeerimine käesolevas töös

SPA olemusest tingitud lahtiühendatud esitluskiht võimaldab hallata kliendi- ja serveripoolt teineteisest eraldiseisvatena, mistõttu võeti *Blazor´*i võimalustest parim võimalik kombinatsioon – kliendipool vastab *Blazor WebAssembly* ning server *Blazor Server* raamistikule. Serveripooles (Joonis 4), milles käivitatakse *ASP.NET Core*'i rakendus, asuvad kontrollerid tegelevad andmete juurdepääsu tagamisele vastavalt API kaudu edastatud kasutajaliidese värskendustele või sündmustele [24].



Joonis 4. *Blazor Server* raamistik ABC4HEDA projektis.

Kliendipool (Joonis 5) käitub töölauarakendusena käivitudes brauseris *WebAssembly* raamistikul põhineval .NET käitussajal (runtime) [24], kus asuvad *Razor* lehed *Blazor* komponentidega. Selliselt ülesehitatult kliendirakendus võimaldab säilitada lehe staatilist visuaalsust ja on võimeline dünaamiliselt värskendama ainult andmeid API kaudu.



Joonis 5. *Blazor WebAssembly* raamistik ABC4HEDA projektis.

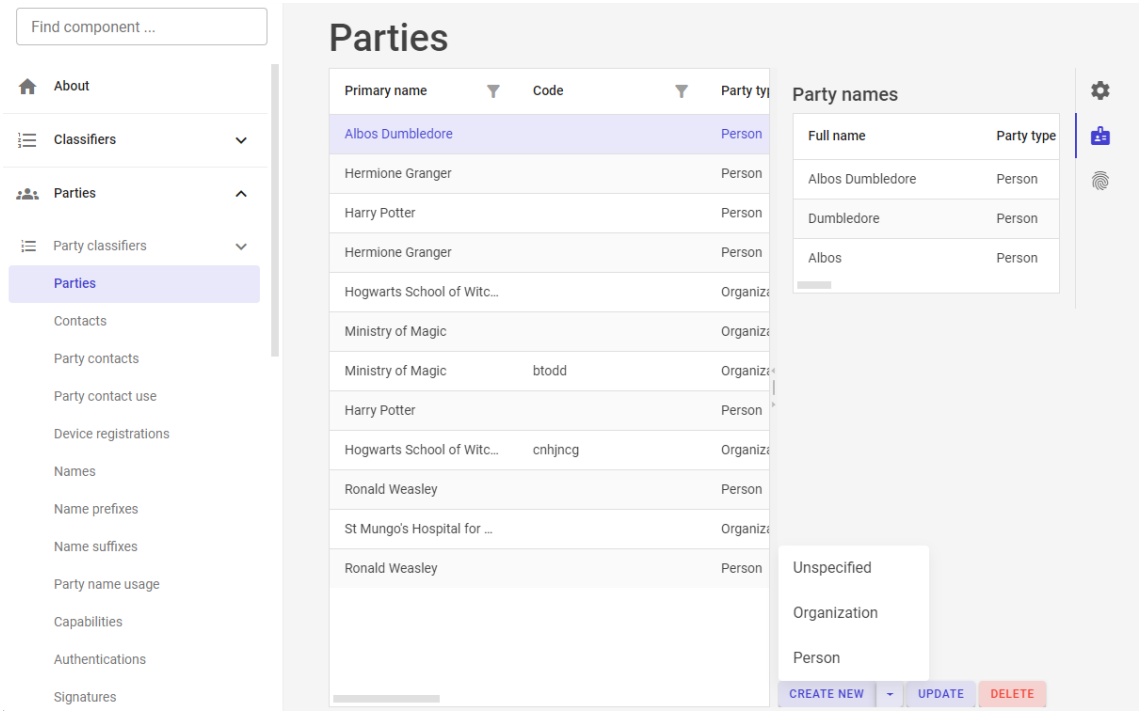
Komponentidest kasutatakse näiteks rippmenüüd (*DropDown*), kus *@parties* on list kõikidest osapooltest (Joonis 6), mille annab *getKeyValues* kasutades API´t, et serveri kaudu vastavad andmed saada. *@bind-Value* tähistab kasutaja konkreetset valikut listis.

```
@using Abc.Facade.Parties

@inherits EditItem<PartyContactView>

<EditDetailed Item="@Item" TDetailed="PartyContactView"
IsAdminView="@IsAdminView">
    <Editors>
        <EditForeignKey @bind-Value="Item.PartyId" Items="@parties" />
        <EditForeignKey @bind-Value="Item.ContactId" Items="@contacts" />
    </Editors>
</EditDetailed>

@code {
    internal IEnumerable<KeyValue> parties { get; set; } =
    Array.Empty<KeyValue>();
    internal IEnumerable<KeyValue> contacts { get; set; } =
    Array.Empty<KeyValue>();

    protected internal override async Task doOnInitializedAsync() {
        parties = await getKeyValues("parties");
        contacts = await getKeyValues("contacts");
    }
}
```

Joonis 6. *Blazor* komponendi realiseerimine vaates *PartyContactView*.

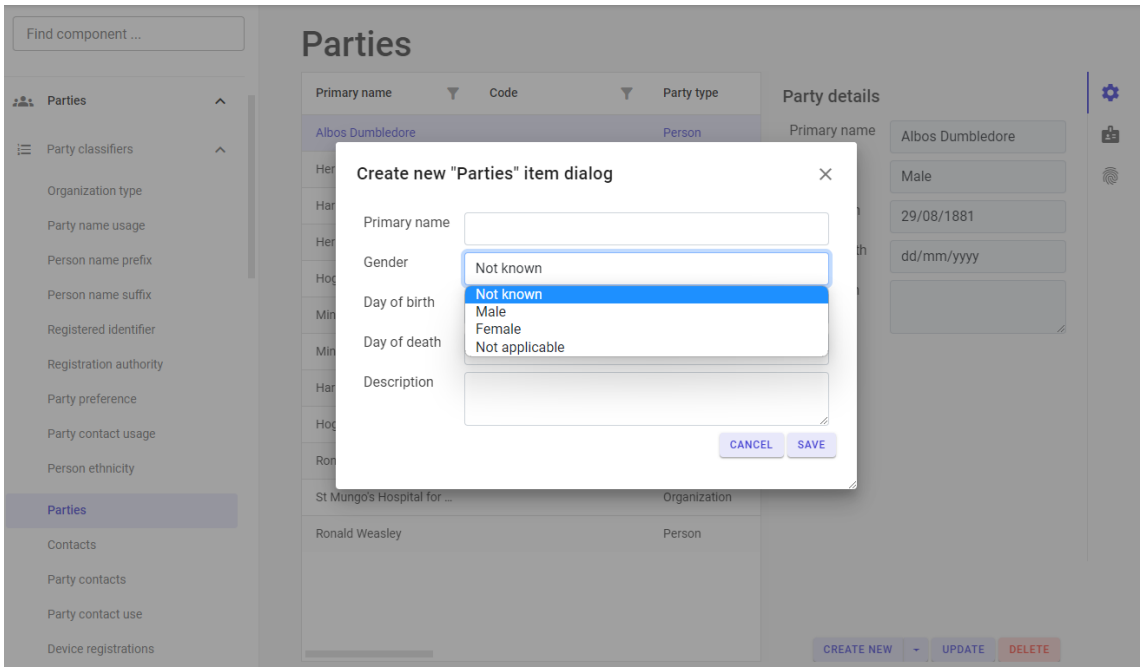
ABC4HEDA-s arhetüüpide realiseerimisel kasutati samasugust kasutajaliidese disaini kõikide arhetüüpide implementeerimisel. Põhilise osa moodustab tabelivaade kõikide registreeritud arhetüüpide väärtustest, kus aktiivse arhetüübi valikul on selle kõrval detailivaade. Samuti on võimalik menüüst valida detailivaate asemel arhetüübi seosed teiste arhetüüpidega, nagu näiteks rollitüübi (*PartyRoleType*) puhul (Lisa 1) on võimalik kuvada rollitüübile kehivat reeglistikku (*RuleSet*).

Samas osapoole (Party) arhetüübi kasutajaliideses oli lisaks standardsetele osadele ka erifunktsioon uue osapoole lisamisel. Nimelt osapoolt lisades saab esmalt teha konkreetse valiku, millist osapoolt lisada tahetakse ning vastavalt sellele kuvatakse lisamise kastis sellele osapooletüübile omased väljad (Joonis 7).



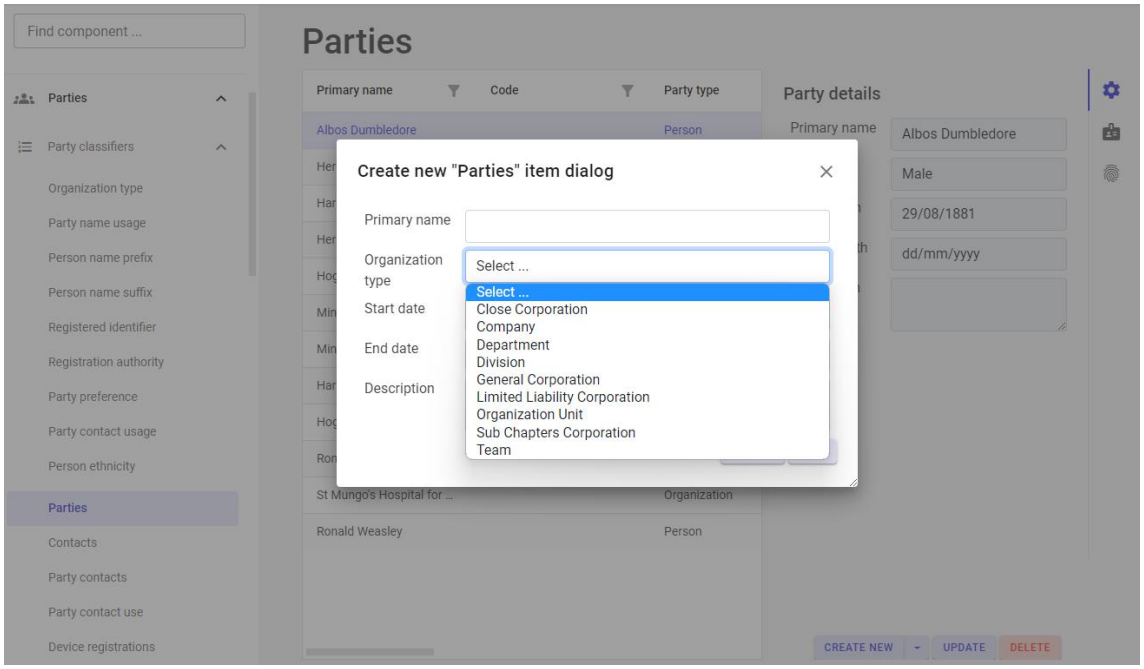
Joonis 7. *Party* vaade kasutajaliides.

Isiku (Person) lisamisel (Joonis 8) on vajalik isiku nimi (*primary name*), sugu (*gender*), sünnipäev (*day of birth*). Vajadusel saab määrata surmakuupäeva (*day of death*) ja anda isiku kohta lisainformatsiooni (*description*).



Joonis 8. Uue isiku (*Person*) lisamine.

Samas uue organisatsiooni (*Organization*) lisamisel (Joonis 9) on oluline vaid organisatsiooni nimi (*primary name*), liik (*organization type*) ja loomiskuupäev (*start date*). Sarnaselt isikule saab lisada kirjelduse (description) ja lõpetamiskuupäeva (*end date*).



Joonis 9. Uue organisatsiooni (*Organization*) lisamine.

Valides kindlaks määramata (*Unspecified*) osapoole lisamise (Lisa 2) on sarnaselt eelnevatele vaja selle nime, vajadusel kirjeldust, kuid lisaks ka kehtivusaja määratlemist (*Valid from/Valid to*) ja osapooletüüpi (*Party type*), mis vaikimis on *Unspecified.*

3.1.3 Implementeeritud arhetüüpide testimine

Osapoole ja Osapoole Seose arhetüüpide realiseerimisel testiti andme-, fassaadi-, vaate- ja domeenikihti kuuluv analoogiliselt juba varasemalt testitud protsessimustrile [8] [6]. Käesoleva töö kirjutamise ajal arendati juurde *Editors* [25] ja kontrollerite (*Controlller*) testid. Viimase lahendusest antakse järgnevalt põgus ülevaade.

Kontroller vastutab kasutajaliidese ja rakenduse vahelise koostöö eest. Vastavalt kasutajapäringule brauserist tagab kontroller päringule sobiva informatsiooni saamise/salvestamise vooloogika rakenduses. *Test4Controller* on abstraktne klass, mis vajab kontrolleri ülesannetest lähtuvalt sisendiks (Joonis 10) testitavat kontrollerit (*TClass*), andmeklassi (*TData*), domeeniklassi ehk äriloogika klassi (*TObject*) ning vaate ja vaatevabriku klassi (*TView, TViewFactory*).

```
public abstract class Tests4Controller<TClass, TData, TObject, TView,
TViewFactory> :
    DomainTests4Sealed<TClass, SentryController<TData, TObject, TView,
TViewFactory>>
    where TData : BaseData, new()
    where TObject : class, IBaseEntity<TData>
    where TView : BaseView, new()
    where TViewFactory : AbstractViewFactory<TData, TObject, TView>, new()
    where TClass : class {
    private int skip;
    private int take;
    private string filter;
    private string fixedFilter;
    private string fixedValue;
    private string orderBy;
```

Joonis 10. Tüübiargumentidega baasklass *Tests4Controller* kontrollerite testimiseks.

Klassisiseselt on defineeritud privaatsed atribuudid, mis deklareeritakse vastava meetodiga *TestInitialize* (Joonis 11).

```
[TestInitialize]
public override void TestInitialize() {
        base.TestInitialize();
        skip = random<int>();
        take = random<int>();
        filter = random<string>();
        fixedFilter = random<string>();
        fixedValue = random<string>();
        orderBy = random<string>();
}
```

Joonis 11. Juhuslike väärtuste initsialiseerimine.

Testmeetodiga *IsCorrectNameTest* testitakse võrdleva vastusepõhise testimustrile põhinedes, kas kontrolleri ja sellele vastava vaate nimed on samad. Arhetüübi realiseerimisel kasutatakse kokkuleppeliselt ja koodistandardit arvestades läbivalt klasside nimetamisel sama nimetust, mille alusel valib kontroller sobiva kasutajapäringu töötlemiseloogika kõikide kihtide klasside seast. *RepoIsNotNullTest* ja *ViewFactoryIsNotNullTest* kontrollivad, kas spetsifitseeritud kontrolleri tüüpargumendid on deklareeritud korrektselt (Joonis 12).

```
[TestMethod]
public void IsCorrectNameTest() {
        var controllerName = GetClass.Name<TClass>("Controller");
        var viewName = Strings.ToPlural(GetClass.Name<TView>("View"));
        areEqual(viewName, controllerName);
}
[TestMethod] public void RepoIsNotNullTest() =>
    isInstanceOfType<IRepo<TObject>>(controller.repo);
[TestMethod] public void ViewFactoryIsNotNullTest() =>
    isInstanceOfType<TViewFactory>(controller.viewFactory);
```

Joonis 12. *IsCorrectNameTest* ja *RepoIsNotNullTest* implementeerimine.

GetAllTest testiga kontrollitakse, kas kontrolleri *Get* meetod kutsub välja kontrolleriga seotud repositooriumi (*repo*) *Get* meetodi, kas tulemused on samad ja kas toimub korrektne teisendamine *Tview* klassiks (Joonis 13).

```
[TestMethod]
public async Task GetAllTest() {
        var count = controller.repo.Get().Count();
        areNotEqual(0, count);
        var l = await controller.Get();
        isInstanceOfType<IEnumerable<TView>>(l);
        areEqual(count, l.Count());
}
```

Joonis 13. *GetAllTest* implementeerimine.

RepoOsNullGetAllTest testib, kas kontroller töötab korrektselt ka siis, kui repositoorium (*repo*) ise on *null*. *RepoIsNotNullGetAllTest* testib, kas kontroller töötab eesmärgipäraselt, kui repositoorium ise ei ole *null*, kuid see on tühi (Joonis 14).

```
[TestMethod]
public async Task RepoIsNullGetAllTest() {
        var l = await controller.get((IRepo<TObject>)null);
        isInstanceOfType<IEnumerable<TObject>>(l);
        areEqual(0, l.Count());
}
[TestMethod]
public async Task RepoIsNotNullGetAllTest() {
        var l = await controller.get(controller.repo);
        isInstanceOfType<IEnumerable<TObject>>(l);
        areNotEqual(0, l.Count());
}
```

Joonis 14. *RepoIs(Not)Null* testide implementeerimine.

ToViewsTest eesmärk on testida, kas *toViews* teisendab valdkonna objektide listi korrektselt vaate objektide listiks (Joonis 15).

```
[TestMethod]
public void ToViewsTest() {
        var lObj = controller.repo.Get();
        var lView = controller.toViews(lObj);
        isInstanceOfType<IEnumerable<TView>>(lView);
        areEqual(lObj.Count(), lView.Count());
}
```

Joonis 15. *ToViewsTest* implementatsioon.

EmptyTest eesmärk on võrrelda, et kas *SentryController´*i meetod *Empty* loob korrektsete etteantud objektidega tühja listi. *CountTest* testib, kas *Count* meetodis jõuavad sisendparameetrid õigetesse väljadesse (Joonis 16).

```
[TestMethod]
public void EmptyTest() {
        isInstanceOfType<IEnumerable<TData>>(SentryController<TData, TObject,
TView, TViewFactory>.empty<TData>());
        isInstanceOfType<IEnumerable<TObject>>(SentryController<TData,
TObject, TView, TViewFactory>.empty<TObject>());
        isInstanceOfType<IEnumerable<TView>>(SentryController<TData, TObject,
TView, TViewFactory>.empty<TView>());
}
[TestMethod]
public async Task CountTest() {
        var count = await controller.Count(filter, fixedFilter, fixedValue);
        areEqual(filter, controller.repo.DynamicLinqFilter);
        areEqual(fixedFilter, controller.repo.FixedFilter);
        areEqual(fixedValue, controller.repo.FixedValue);
        areEqual(0, count);
}
```

Joonis 16. *EmptyTest* implementeerimine.

Sarnaselt *CountTest´*iga testitakse *GetTest´*is, kas *Get* meetodisse antud parameetrid jõuavad õigetesse väljadesse (Lisa 3), ning *RepoIsNullGetTest* eesmärk on võrrelda, kas argumentidena antud väärtused jõuavad õigetesse kontrolleri väljadesse, kui repositoorium on *null*, ning *RepoIsNotNullGetTest* testib sama, kuid *repo* ei ole null, vaid tühi (Lisa 4). *GetCountRepoIs(Not)Null* testi eesmärk on kontrollida, kas kontrolleri vastavad muutujad saavad ettenähtud väärtused (Lisa 5).

PageSizeTest kontrollib, kas kontroller saab lehekülge suuruse (*PageSize*) repositooriumist. *KeyValuesTest* testib, kas kontrolleri *KeyValues* arvutab *SelectItem* listi õigesti (Joonis 17).

```
[TestMethod]
public async Task PageSizeTest() {
        var count = await controller.PageSize();
        areEqual(controller.repo.PageSize, count);
}
[TestMethod]
public async Task KeyValuesTest() {
        var l = await controller.KeyValues(filter);
        var all = controller.repo.Get();
        foreach (var e in l) {
            var o = all.FirstOrDefault(x => x.Id == e.Key);
            isNotNull(o);
            var v = getItemValue(o);
            areEqual(v, e.Value);
        }
}
```

Joonis 17. *PageSize* ja *KeyValues* testide implementeerimine.

Sellise *Test4Controller´*i kasutamine võimaldab testida kõiki kontrollereid väikese vaevaga, andes tüübiargumentidele väärtused kasutades näidisrepositooriume (*mocked repos*), nagu näiteks *responsibilitiesRepo* (Joonis 18).

```
[TestClass]
public class ResponsibilitiesControllerTests :
    Tests4Controller<ResponsibilitiesController,
        ResponsibilityData, Responsibility, ResponsibilityView,
ResponsibilityViewFactory> {
    protected override ResponsibilitiesController createObject() =>
new(responsibilitiesRepo);
}
```

Joonis 18. *ResponsibilitiesControllerTests* kasutades baastestklassi *Test4Controller*.

3.2 ContSys mõisted Osapoole ja Osapoole Seoste arhetüüpmustrite keeles

3.2.1 *Party* arhetüübile vastavad ContSys mõisted

Osapoole arhetüüp, mis on tarkvarasüsteemis inimese või organisatsiooni mudel, on identifitseeritav ja adresseeritav üksus, millel võib olla juriidiline staatus ja autonoomne kontroll (vähemalt osade) tegevuste üle [15]. Näiteks automatiseeritud meditsiiniseadet võib teatud juhtudel, nagu näiteks verelaboriseade [26], mis on võimeline teatud määral autonoomselt analüüse tegema, pidada osapooleks (Tabel 1).

Tabel 1. *Party* arhetüübile vastavad ContSys terminid.

Party arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
Automated medical device	Automatiseeritud meditsiiniseade
(Health)care personnel	Meditsiinipersonal
(Health)care third party	Tervishoiu kolmas osapool
Healthcare actor	Tervishoiu osapool
Healthcare party	Tervishoiu osapool
Individual provider	Meditsiinipersonal
Party	Osapool

3.2.2 *Person* arhetüübile vastavad ContSys mõisted

Isiku (*Person*) arhetüüp esindab informatsiooni inimese kohta, mille eristamine osapoolest on tervishoiu infosüsteemides kriitilise tähtsusega (Tabel 2). Isikule omaselt saab valikuliselt lisada sünniaja [15], mis on oluline näiteks retseptiravimi määramisel. Näiteks Klabax antibiootikumi, mis sisaldab klaritromütsiini, ei annustata alla 12-aastastele lastele [27].

Tabel 2. *Person* arhetüübile vastavad ContSys terminid.

Person arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
Person (term)	Isik

3.2.3 *BodyMetrics* arhetüübile vastavad ContSys mõisted

Kehameetrika (*BodyMetrics*) arhetüüp võimaldab salvestada teavet inimese biomeetriliste mõõdikute kohta (kehakaal, silmade värv, pikkus jne) [15]. Tervisega seotud seisundid on, kas ise kehameetrika näitaja, näiteks vererõhk või peavalu (*observed condition*), või selle seisundiga seotakse hulk biomeetrilisi näitajaid, näiteks äge müokardiinfarkti (*professionally assessed condition*) üheks sümptomiks on peavalu. Sümptomid on seotud terviseprobleemiga (*health problem*, *health issue*), nagu näiteks rinnavähk, mistõttu saab neid salvestada tervisemurena, mis on terviseloo (*professional health record*) alus. Terviselugu koosneb aga kliinilistest põhjendustest (*entry*), mida saab samuti kehameetrikat puudutavate andmete salvestamisena tõlgendada. Diagnoosiga (*working diagnosis*) seotakse arvesse võetav seisund (*considered condition*) täheldatud seisunditega, mistõttu on diagnoosi määramise aluseks biomeetrilised andmed. Näiteks eelmainitud äge müokardiinfarkt on nii tervishoiu töötaja poolt hinnatud terviseseisund kui ka diagnoos (Lisa 6).

3.2.4 *Organization* arhetüübile vastavad ContSys mõisted

Organisatsiooni (*Organization*) arhetüüp esindab haldus- ja funktsionaalset struktuuri. Kui inimesi on ainult ühte tüüpi, siis organisatsioone on palju erinevaid [15]. Lisaks tervisekindlustusfondile, saab ka perekonda (*healthcare supporting organization*) organisatsioonina käsitleda [2] (Tabel 3).

Tabel 3. *Organization* arhetüübile vastavad ContSys terminid.

Organization arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
(Health)care organization	Tervishoiuorganisatsioon
(Health)care supporting organization	Tervishoiu tugiorganisatsioon
Healthcare delivery organisation	Tervishoiuorganisatsioon
Organization	Organisatsioon

3.2.5 *Preferences* arhetüübile vastavad ContSys mõisted

Preferences arhetüüp esindab osapoole väljendatud valikut või eelistust, sageli võimalike või pakutavate valikute hulgast. Eristatakse osapoole personaalseid (nt toitumisharjumused) kui ka osapoole rolli mängiva osapoole eelistusi, mis tekivad osapoolele konkreetses kontekstis. Igal eelistusel on eelistuse tüüp (*PreferenceType*), mis määrab eelistuse nimetuse, kirjelduse ja valiku (*PreferenceOptions*). Iga eelistus määrab täpselt ühe valiku loetletud valikute hulgast selles eelistuse tüübis (Tabel 4) [15]. Näiteks inimene saab täita digiloos doonorluse (*PreferenceType*) tahteavalduse, mida saab tõlgendada (potentsiaalse) patsiendi tahtena (*subject of care desire*). Küll aga tuleb teha valik elundite, vere või surnukeha loovutamise tahteavalduste vahel [28] (*PreferenceOptions*). Allkirjastades vereülekande tahteavalduse, siis sellest saab konkreetne *Preference*.

Tabel 4. *PreferenceType* arhetüübile vastavad ContSys terminid.

PreferenceType arhetüüp	
ContSys mõiste	**ContSys mõiste**
Dissent	Abist keeldumine
Subject of care desire	Patsiendi tahe

3.2.6 *PartyRole* arhetüübile vastavad ContSys mõisted

PartyRole arhetüüp kajastab osapoole rolli semantikat konkreetses osapoole seostes (*Party Relationship*), mida kasutatakse teabe salvestamiseks, mis puudutab osapoole rolli ennast, mitte teavet osapoole või osapoole seose kohta. Näiteks perekonnas õe-õe suhtes mängib üks naisosapool vanema õe rolli ning teine noorema õe rolli. Osapoolte rollide üksikasjaliku semantika huvi korral on vajalik andmeid salvestada üksikasjalike

kirjelduste, kohustuste, eelistuste ja muu sellise kohta. Enamik osa sellest informatsioonist on ühine kõigi teatud *PartyRole* arhetüüpide puhul, mistõttu kogu teabe salvetsamiseks kasutatakse *PartyRoleType* arhetüüpi, mille teabele saab viidata *PartyRole*'i eksemplarides [15].

Roll on seos rollitüübi ja inimese vahel, kus inimene on konkreetne isik ja tüüp võib olla patsient, arst, hooldusõde. Sellisel juhul saab roll koosneda erinevatest rollitüüpidest vastavalt vajadusele ning on tagatud, et arst võib teatud tingimustel patsiendirollis olla. Kasutades rolli piiranguid (*PartyRoleConstraints*) saab kirjeldada, mis tüüpi osapooled võivad, milliseid osapoole rolle mängida. Iga *PartyRoleType*'iga on seotud üks või mitu *PartyRoleConstraint*'i. Näiteks kui teatud *PartyRole* tüübil on *PartyRoleConstraint*, mille osapoole tüüp on "*Organization*", võib iga organisatsiooni tüüpi osapool seda tüüpi *PartyRole*'i kasutusele võtta. Lähtudes sellisest interpretatsioonist on ContSys standardis rollidele viitavad terminid kirjeldatavad rollitüüpidena (Tabel 5), mis seotakse arhetüüpide implementeerimisel osapoolega või selle tüübiga läbi rolliobjekti või rollipiiranguobjekti.

Tabel 5. *PartyRoleType* arhetüübile vastavad ContSys terminid.

PartyRoleType arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
(Health)care professional	Tervishoiutöötaja
(Health)care provider	Tervishoiuteenuse osutaja
Client	Patsient
Health professional	Tervishoiutöötaja
Health(care) (service) provider	Tervishoiuteenuse osutaja
Informal/other carer	Muu abiosutaja
Next of kin	Lähedane isik
Organization role	Organisatsiooni roll
Patient	Patsient
Person role	Isiku roll
Role	Roll
Service user	Patsient
Subject of (health)care	Patsient
Subject of care agent/proxy	Patsiendi esindaja

3.2.7 *PartyRelationship* arhetüübile vastavad ContSys mõisted

PartyRelationship arhetüüp kajastab semantilist seost osapoolte vahel, milles kummalgi osapoolel on konkreetne roll (Tabel 6) [15]. Näiteks volitus loob üldiselt osapooltevahelise seose, kus üks volitab teist millegi eest vastutama. Varasemates töödes on organisatsioonimudelit käsitletud kui osapoole seost [7],

Tabel 6. *PartyRelationship* arhetüübile vastavad ContSys terminid.

PartyRelationship arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
Authorization by law	Seadusega antud volitus
(Health)care employement	Töösuhe tervishoius
(Healthcare) contact	Tervishoiukontakt
Healthcare communication	Suhlus
Organizational pattern	Organisatsiooni mudel

Sarnaselt osapoole rollile, saab osapoole seoste üksikasjalike kirjeldusi ja nõudeid teabena salvestada osapoole seosetüübi (*PartyRelationshipType*) arhetüüpi kasutades ning seejärel osapoole seoste eksemplaris sellele viidata [15]. Näiteks võib patsiendil olla mitu aktiivset eesootavat visiiti erinevate arstidega ning sellisel juhul on üheks seosetüübiks vastuvõtt kardiloogiga, teine uroloogiga. Samuti võib raviperioodi jooksul toimuda mitu tervishoiukontakti, mistõttu esmast tervishoiukontakti (*initial contact*) saab lugeda üldiste tervishoiukontaktide tüübiks (Tabel 7).

PartyRelationshipConstraints arhetüübiga saab sõnastada reeglid kirjeldamaks, millised osapoole rollid võivad millistes osapoolte seostes olla. Näiteks hetkel kehtivate õigusaktide kohaselt võivad abielluda mees ja naine [28]. Igal osapoole seosetüübil on üks või mitu osapoole seosepiirangut. Valikulise reeglistiku (*RuleSet*) nõuded määratlevad tingimuste komplekti, mis peavad olema täidetud enne, kui kaks konkreetset osapoole rolli saavad osaleda teatud tüüpi osapooltevahelises seoses [15].

Volituste erinevad tüübid on kirjeldatavad osapoole seosetüübina. Seadusest tulenevalt on näiteks alaealise lapse eestkostja täisealine isik (*PartyRelationshipConstraints*) [29] omades õigust eestkostet pakkuda (PartyRelationshipType). Selline volitusõigusepõhine seos lõppeb lapse täisealiseks saamisel [29] ning selline seosetüüp (eestkoste õigus) nende vahel enam ei kehti – saab kehtida muu seosetüüp.

Tabel 7. *PartyRelationshipType* arhetüübile vastavad ContSys terminid.

PartyRelationshipType arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
Appointment	Korraldatud kokkusaamine kellegagi
(Health)care mandate/ commission	Tegevusvolitus tervishoius
(Health)care period mandate	Ravivolitus
Clinical process mandate	Kliinilise protsessi läbiviimise volitus
Commission to export personal information	Isikliku teabe väljastamise volitus
Continuity facilitator commission/mandate	Ravi järjepidevuse korraldamise volitus
Demand commission/mandate	Nõudemandaat
Healthcare activity commission/mandate	Tervishoiutegevuse volitus
Healthcare appointment	Tervishoiutöötaja vastuvõtt
Initial contact	Esmane tervishoiukontakt
Mandate to export personal information	Isikliku teabe väljastamise volitus

3.2.8 *Responsibility* arhetüübile vastavad ContSys mõisted

Osapoolte rollile on võimalik määrata vastutused (*Responsibility*), mis kujutavad endast tegevuse kirjeldust, mida *PartyRoleType*'ilt võidakse oodata. Iga seda tüüpi osapoole rolli omav osapool peab võtma endale kõik kohustuslikud kohustused ja võib võtta osa või kõik valikulised kohustused, mis sellele rollile määratud on (Tabel 8) [15].

Vastutuse võtmist osapoole rolli alusel saab modelleerida määratud vastutusena (*AssignedResponsibility*). Määratud vastutusel võib olla valikuline kehtivusperiood, mis algab ja kestab kuni ning soovi korral võib selle allkirjastada osapoole allkirjaga (*PartySignature*), mis registreerib vastutusemääraja. Igal vastutusel võib olla ka vastutuse nõuete kogum, mis määratleb piirangud, mis peavad olema täidetud enne, kui vastutus osapoole rollile määratakse [15]. Näiteks kui tervishoiuteenusel osalemiseks tagatakse juurdepääs isikuandmetele, siis kaasneb sellega isikuandme kaitse seaduse kohaselt vastutus kohelda neid seaduspäraselt [30]. Selline juurdepääs tagatakse ka arstiõppe üliõpilastele, aga ainult neile, kes on läbinud õppekavas olevad neljanda kursuse kohustuslikud ained [31] (vastutuse määramise piirang). Ka organisatsiooni rollidele saab

määrata vastutusi, näiteks haiglale on määratud tervishoiuteenuse osutamise kohustus (*healthcare commitment*).

Tabel 8. *Responsibility* arhetüübile vastavad ContSys terminid.

Responsibility arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
(Health)care commitment	Tervishoiuteenuse osutamise kohustus
Commitment	Kohustus

3.2.9 *Capability* arhetüübile vastavad ContSys mõisted

Pädevuse (*Capability*) arhetüüp kirjeldab, mida osapool on võimeline tegema. Näiteks hooldusõiguslikul vanemal või füüsilisest isikust eestkostjal on pädevus anda nõusolek lapse lühiajalise asendushooldusteenuse osutamiseks katkematult kuni 90 päeva [32]. Osapoole võimeid talletatakse reeglistikku (*RuleContext*) kirjeldatavate nõuetena osapoole rollidele ja vastutusele, sest rollidele/vastutusele vastavad pädevused peavad olema täidetud enne, kui osapool saab mängida teatud tüüpi osapoole rolli. Näiteks hooldusõe rolliks on vajalik vastav kvalifikatsioon. See on nõue, mis tagab pädevuse hooldusõena töötamiseks (Tabel 9).

Tabel 9. *Capability* arhetüübile vastavad ContSys terminid.

Capability arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
Care professional entitlement	Tervishoiutöötajana tegutsemise õigus
Consent competence	Nõusolekuvõime
Dissent	Abist keeldumine
Healthcare professional entitlement	Tervishoiutöötajana tegutsemise õigus
Subject of care desire	Patsiendi tahe

4 Analüüs ja järeldused

Järgnevalt analüüsitakse Osapoole ja Osapoole Seose arhetüüpide sobivust ContSys mõistete deklaratiivseks spetsifitseerimiseks, lisades seejuures hinnangu ABC4HEDA arendusmuudatuste tulemustele.

4.1 Osapoole ja Osapoole Seoste SPA raamistiku analüüs

ABC4HEDA arhitektuur on disainitav nii MVC kui ka SPA mustreid jälgides, kuid nende rakendamine ja eesmärgid on mõnevõrra erinevad. Veebirakenduse arendamisel SPA raamistikus koondatakse kogu funktsionaalsus ühte lehte, võimaldades värskendada dünaamiliselt kasutajaliidest vastavalt kasutaja tegevustele. See tähendab seda, et lehti ei laeta uuesti, kui kasutaja rakenduses navigeerib, vaid andmeid uuendatakse vaid teatud rakenduse leheosas. Sellest tingituna on SPA raamistikul mitmeid eeliseid MVC ees.

SPA tagab kasutajasõbralikkuse. Kuna SPA ei vaja veebilehtede navigeerimise vahel lehtede uuesti laadimist, on kasutajakogemus sujuvam ja kiirem. Eriti oluline on see rakenduse kasutamisel mobiiliseadmetes, kus võrguühendus võib olla ebaühtlane.

SPA vähendab koormust serveris. Pärast veebilehe esmakordset laadimist esitatakse kasutajapäring serverile vaid andmete värskendamiseks, mitte kogu lehe uuesti laadimiseks, mis võib parandada rakenduse jõudlust ja muuta arendust paindlikumaks. SPA on testitav otse brauserist ilma, et peaks iga kasutajapäringuga serveri poole pöörduma.

Blazor on *Microsoft´*i poolt loodud SPA raamistik, mis eristub teistest SPA raamistikest mitmel viisil. *Blazor* võimaldab luua veebirakendusi C# keeles, mis on .NET arendajatele tuttavam ja mõistetavam kui *JavaScript*. See võib arendusprotsessi lihtsamaks ja kiiremaks muuta. *Blazor* toetab täielikult .NET raamistikku. Blazor´iga saab kasutada kõiki .NET ökosüsteeme (sealhulgas tööriistu, *library´*d ja keeli) [33], võimaldades arendajatel kasutada olemasolevaid teadmisi ja oskusi.

Blazor toetab nii serveri- (*server-side*) kui ka kliendipoolset (*client-side*) renderdamist, mis annab arendajatele paindlikuse rakenduse arhitektuuri valikul. Blazor võimaldab C# koodi jooksutada brauseris *WebAssembly* abil. Võrreldes laadimisaega, on *Blazor Server* kiirem kui *WebAssembly*, kuna see töötab serveri poolel ja seetõttu on allalaadimise maht väiksem. Seevastu *Blazor WebAssembly* rakendus töötab täielikult brauseris kasutades kasutajaseadmeressursse ja pakub täielikku ühelehelise rakenduse kasutajakogemust suurema algkoormuse korral. Kui rakendus on kliendi poolel täielikult käivitatud, toetab see võrguühenduseta stsenaariumi, kuna kõigi kasutajaliidese värskenduste jaoks pole vaja serveri poole pöörduda. Samas *Blazor Server* võimaldab tundlikku teabe turvalisemat salvestamist, kuna kasutajatel pole andmetele otsest juurdepääsu [34].

4.2 Osapoole ja Osapoole Seoste disainimustrid

Implementeerides ABC4HEDA-s arhetüüpe C# keelt eriti toetava *Blazor´*iga, arvestati kodeerimisel objekt-orienteeritud keelele omaselt tarkvara disainimustreid, mis nõuavad objektide struktureeritust. Arhetüüpide implementeerimisel arvestati avatud-suletud põhimõttega näiteks rollide haldamisel, kus olemasolevat rolli Arst ei muudeta (suletud), kuid vajadusel lisatakse uus rolli Perearst muutmata juba olemasolevat rollitüüpi Arst (avatud).

Single Responsibility põhimõtte kohaselt peab klass või moodul omama ainult ühte vastutust, mistõttu on vaid üks põhjus klassi muutmiseks [20]. Projekti arhitektuurivaates on nimetatud põhimõtet rakendatud *Data, Domain* ja *Facade* kihtides, kus iga kiht vastutab vaid ühe konkreetse ülesande eest. Andmekiht vastutab andmete kättesaamise eest andmebaasist, domeenikiht äriloogika eest kasutades andmeid, mis tulevad andmebaasist ning vaatekiht andmete kuvamise eest kasutajaliides. ContsSys terminite spetsifitseerimine arhetüüpmustrites baseerub justnimelt *Single Responsibility* põhimõtte rakendamisel. Meditsiinivaldkonna mudeli jaoks oleks vaja, et igal mõistel oleks oma kindel koht selles mudelis kindla selgitusega ning samaaegselt selgitus ei kattuks teiste terminite selgitustega. Probleemseks osutub nende mõistete spetsifitseerimine, mida on võimalik tõlgendada mitmeti – näiteks nii osapoolena kui tootena. Sellisel juhul ei ole ContSys terminil mudeli implementeerimisel ühte kindlat vastutusala.

Liskov Substitution põhimõte sätestab, et alamklassid peavad olema asendatavad enda ülemklassiga rakendust seejuures rikkumata [35]. Näiteks piiratud teovõimega

nakkushaigele arstiabi andmine võib toimuda kas tema seadusliku esindaja nõusolekul või vältimatu abi korral esindaja nõusoleku puudumisel [36]. Arstiabi andmine selles kontekstis toetub tegevusvolitusele tervishoius (*healthcare mandate*), mis nõuab kuni ühte seadusega antud volitust (*authorization by law*) [2]. Sellisel juhul on Liskovi printsiibi kohaselt seadusega antud volitus ülemklass ning seadusliku esindaja nõusolek ja vältimatu arstiabi korral selle puudumine ülemklassi alamklassid. Arstiabi andmine sisuliselt ei sõltu sellest, kummast seadusega tulenevast õigusest lähtutakse, kuna mõlemad täidavad seadusest tuleneva volituse rolli. Lisaks sellele tuleb meditsiiniinfosüsteemis arvestada vajadusega, et arst võib ise arstiabi patsiendina vajada, ei tohi olla Liskovi printsiibi kohaselt osapoolt Patsient ja osapoolt Arst. Sellisel juhul on osapool Isik, kellele saab määrata nii arsti kui patsiendi rolli vastavalt olukorra tingimustele. Isiku sisuline pool ei muutu tema poolt täidetava rolli tõttu.

Kui volitus on osapoolte seosetüüp, siis ContSysi mõiste nõudemandaat (*demand mandate/commission*), mis hõlmab õigust ja kohustust nõuda tervishoiutegevust, on kõikide tervishoiu tegevusvolituste osapoolte seose baastüüp. Sellest tuletades erinevaid tegevusvolitusi (volitustüüpe) on igal selliselt tuletatud volitusetüübil unikaalne liides (*interface*). See allub *Interface Segregation* nõudele, kuna ühe liidese, mis tagaks ligipääsu kõikidele volitusetüüpidele, asemel kasutatakse iga volitusetüübi jaoks eraldi liidest. *Gang of Four* disainimustritest [36] on projektis endiselt kasutusel äriloogika kihis (näiteks *PartyFactory*) abstraktsete objektide üksuste loomine [6].

4.3 Testimine ja testimise mustrid

Ühiktestimisel kasutatud *Microsoft Unit Testing* raamistikuga tagati järjepidev koodihaldus, et kohe vähendada koodi funktsionaalset ebakõla, kui tehti uusi koodimuudatusi. Põhiliselt toetuti testide kirjutamisel abstraktsiooni mustrile [37], luues abstraktseid testklasse, andes spetsifitseerimisel nendele sisendiks tüüpargumentidega konkreetsed klassid või kirjutati need üle virtuaalsete (*virtual*) meetoditena. Puhta koodi nõudeid jälgides on kontrollerite testimine lahendatud ühe abstraktse baastestina *Test4Controller*, mida kõik kontrollerid pärivad, andes sellele vastava arhetüübiga seotud olulised klassid. Nii oli võimalik vähendada oluliselt analoogilise koodi kordust ja tagada organiseeritult jälgitav ülesehitus.

4.4 Osapoole ja Osapoole Seoste sobivus valdkonnapõhiseks spetsifitseerimiseks

ContSys mõistete valideerimisel selgus, et nii mõnigi termin on tõlgendatav mitme arhetüüpmustri kontekstis ehk ei võimalda semantilist koostalitlusvõimet (Tabel 10). See takistab ühese meditsiinivaldkonna mudeli spetsifitseerimist. Raviperiood seob erinevaid osapooli (patsient ja arstid) kindlas ajavahemikus, mis viitab Osapoolte Seosele. Raviperioodi käigus võib patsient kokkupuutuda erinevate meditsiinitöötajate, tervisenähtude ja nendega seotud raviplaanidega, mistõttu on raviperiood tõlgendatav protsessina, täpsemalt protsessilõimena, kus inimene tuleb tervisekaebusega tervishoiuasutusse ja on seotud seni kuniks temaga lõpetatakse kõik seosed (pole enam ka kodusel ravil, ei pea käima ülevaatusel jne). Epikriis suletakse tavaliselt juhul, kui inimene saab kas terveks, sureb ära või haigusjuhtum on muutunud krooniliseks [38]. Teades, et protsess põhineb osapooltevahelisel seosel, on tõlgendused põhjendatud.

Abist keeldumine on inimese eelistus näiteks siis, kui inimene on digiloos allkirjastanud mitteelustamise tahteavalduse. Kui patsient on allergiline *amoxicillin´*i vastu, ei ole ta võimeline seda sisaldavaid ravimeid tarvitama, mis võib klassifitseeruda abist keeldumise alla – patsiendina eelistaks õige antibiootikumi tarvitamist, aga allergilisi reaktsioone arvestades ei ole selleks võimeline (*Capability*). Automatiseeritud meditsiiniseadme puhul võib tegemist olla osapoolega, nagu eelnevalt mainitud, kuid kui kõik käsklused annab sisendiks inimene, on tegu tootetüübiga [25]. Sama kehtib ContSys mõistega *nõudemandaat*, mis ühest küljest on osapooltevaheline seosetüüp (määrab volituse aluse). Teisest küljest võib tegu olla raviasutuse teenusetüübiga, nagu ka ravivolituse puhul [25].

Tabel 10. Semantiliselt ebakõlas ContSys terminid.

ContSys mitmeti tõlgendatavad mõisted		
ContSys mõiste	**ContSys mõiste eestikeelne vaste**	**Arhetüüpide valikuvariant**
Appointment	Korraldatud kokkusaamine kellegagi	PartyRelationshipType
		Service
Automated medical device	Automatiseeritud meditsiiniseade	Party
		ProductType
Demand commission	Nõudemandaat	PartyRelationshipType
		ServiceType

ContSys mitmeti tõlgendatavad mõisted		
ContSys mõiste	**ContSys mõiste eestikeelne vaste**	**Arhetüüpide valikuvariant**
Dissent	Abist keeldumine	Capability
		Preference
(Health)care period mandate	Ravivolitus	PartyRelationshipType
		ServiceType
Period of healthcare	Volitustega kaetud raviperiood	PartyRelationship
		Process

Protsessimustri spetsifitseerimisel [6] tõlgendati mõningaid termineid osapoolena, mille suhtes on käesoleva töö autoril eriarvamus (Lisa 7). Varsemalt on tervishoiuteenuse osutamise kohustust (*healthcare commitment*) valideeritud kui osapoole allkirja (*PartySignature*) põhjendusel, et allkirjastamisel osapool võtab omaks kohustuse, mis on teise osapoole poolt talle määratud. Küll aga allkirjaga kinnitatakse selle kehtivus ja tõepärasus. Tervishoiuteenuse osutamise kohustus oma mõistelt on tõlgendatav siiski kohustuse (*Responsibility*) arhetüübina. Seni osapoole rollitüübina käsitletud tervishoiu rahalised vahendid (*healthcare funds*) on käesoleva töö autori hinnangul ebatäpselt spetsifitseeritud, kuna ContSys selgituse alusel on tegu ressurssiga, mida võivad osapooled kasutada teenuste maksmiseks [2]. Samas töös hinnati tervishoiuressurssi (*healthcare resource*) osapoolena (*Party*), mille alusel peaks tervishoiu rahalised vahendid (teatud liiki tervishoiuressurss) olema osapoole tüüp, mitte rollitüüp. Antud kontekstis ei ole rahalised vahendid osapoole rollitüüp ega osapooletüüp, sest näiteks Tervisekassa ei saa võtta endale rolliks raviraha. Roll saab olla ravirahastamine, milleks kasutatakse ressurssi ehk raviraha. Raviraha ei ole ka teatud autonoomsete omadustega, mistõttu ei saa seda lugeda ka osapooleks. Varasemalt isikuna (*Person*) defineeritud meditsiinipersonali (*healthcare personnel*) klassifitseerib käesoleva töö autor osapoolena (*Party*). Personal koosneb paljudest isikutest ja personal ei ole tõlgendatav üksikisikuna. Samuti ei ole meditsiiniseade (*medical device*) osapool, sest eristatakse automatiseeritud seadet tavalisest seadmest, millel võib olla teatud osapoole omadused, ning mistõttu selliselt tõlgendatult ei ole automatiseerimata seade osapoolena käituv. Samuti lükatakse ümber, et tervishoiu osapool (*healthcare actor*) on osapoole roll (PartyRole). Termini

kohaselt on tegu organisatsiooni või isikuga, kes osaleb tervishoius [2] – vastab osapoole (*Party*) definitsioonile.

4.5 Tööteema aktuaalsus erinevates publikatsioonides

ContSys standardi laialdane kasutuselevõtt on tervishoiu infosüsteemide modelleerimisel arenemisjärgus, kuid sellegipoolest on Osapoole ja Osapoole Seoste arhetüüp ning ContSys erinevate infosüsteemide implemeneteerimisel on leidnud kajastatust. ContSys kontseptsiooni toel on püütud näiteks imikute toitmise eest hoolitsemisprotsessi (*the process of care of infant feeding*) infomudelina määratleda, kasutades kliiniliste mõistete tõlgendamisel lisaks ISO EN13606 standardit. Mudel kirjeldab ContSysi terminoloogial põhinevat protsessi kontseptsioone ja asetust üksteise suhtes [39].

ContSysi kui ka HL7 [9] kontseptsioonide kohandamisel väljatöötatud ka EHR (*Electronic Health Record*) süsteemi kontseptuaalne mudel, millel põhinev tarkvara arhitektuur on ABC4HEDA-le sarnaselt jaotatud kihtideks. Artiklis on antud üksikasjalik ülevaade, kuidas reaalsest maailmast kogutud informatsiooni muuta infosüsteemis kasutatavaks teadmiseks olemi (*entity*) koha. Sellise *mis-millal-kus-kes-kuidas-miks* andmestruktuuri loomisel mängis olulist rolli UMLS sõnastik, mis määratleb üle ühe miljoni biomeditsiinilisi mõiste ja hõlmab semantiliste seoste võrgustikku nende mõistete vahel, hõlbustades arvutisüsteemide arendamist [40].

Meditsiiniinfosüsteemide kavandamisel on oluline arvestada tervishoiu nõuete ja keskkonna pidevate muutustega. Sellest tulenevalt on leitud, et kui eesmärk on ehitada infosüsteem, mis on tulevikukindel ja koostalitlusvõimeline, tuleks lisaks äriloogika eraldamisele programmikoodist (kihiline arhitektuur) modelleerida ainult need mõisted, mis teadaolevalt kehtivad kõikidel juhtudel ja on ajas muutumatud. Artikli autorid on seisukohal, et seda täidavad kõik Osapooled, kellel on alati juriidiline identiteet ja rollid. Arhetüüpsemantikat defineeritakse artiklis pigem keele kui mudelina, mis parandab käitlusaja paindlikkust, kuid iseloomustab infosüsteemi projekteerimisaja keerukust. Samas ei ole objektorienteeritud kirjanduses piisavalt käsitletud arhetüüpide ja valdkonnaspetsifilise ontoloogia vahelist piirangulist seost [41].

Mudelipõhist arhitektuuri (MDA) on kasutatud ka maahaldus rakendustarkvara (LADM) arendamisel, kus lisaks Osapoole Rollile on kasutusele võetud Osapoole Grupp (*Group*

Party), et eristada osapoolterühma eraldiseisva osana. Sellises kontekstis on võimalik liikmete (*Party Member*) liikmelisuse informatsiooni dokumenteerida [42]. Tuues paralleele käesoleva tööga, on võimaik näiteks osapoolena tõlgendatud meditsiinipersonali käsitleda osapoolegrupina, sellisel juhul on osapooleks näiteks haigla, kuhu kuulub muuhulgas meditsiinipersonalile ka IT-personal jne. Kui selles kontekstis oleks meditsiinipersonal siiski osapool, on selle osapoolegruppideks näiteks erinevad haiglaosakonnad, kes kõik on meditsiinitöötajad (EMO, kardioloogia, uroloogia jne).

Teisalt, mudelipõhisel arhitektuuril põhineva katastrisüsteemi arendamisel peeti oluliseks eristada isikut (*Person*), kas mõistega *NaturalPerson* või *NonNaturalPerson* (üldiselt organisatsioon, valitus, ettevõte), mille moodustavad isikugrupid (*GroupPerson*). *GroupPerson´*i moodustavad selle käsitluse kohaselt vähemalt kaks isikut ning sinna võib kuuluda kõiki tüüpi isikud, k.a mõni muu GroupPerson. Isiku ja isikugrupi vaheline seos on väljapakutud liikemte assotsiatsiooniklassina (*Members*), kus iga liikme kohta oleks atribuudid *osalus grupis* ja *liikmelisuse periood* [43]. Sellisel juhul on meditsiinipersonal *GroupPerson*, koosnedes isikutest aga olles osa organisatsioonist. Sellisel juhul oleks *NonNaturalPerson´*i vaates isiku tüübiks juriidiline isik. Samas uurimistöös tuuakse välja, et katastrimudeli kasutamise üks eesmärkideks lisaks efektiivsuse tagamisel on pakkuda välja üks ühine andmevahetusmudel erinevate katastrisüsteemide vahel. Seda eesmärki saab üle tuua ka tervishoidu, näiteks hetkel erinevate maakondade haiglate analüüsid on kättesaamatu mõne muu maakonna haiglas. See mõjutab tugevalt arstiabi efektiivset kvaliteeti, kuna näiteks igal visiidil peab praegu inimene erinevate teenuseosutajate juures sama terviseinfot (nt vereproovide vastused) korduvalt jagama [44].

4.6 Kitsendused ja piirangud töös

Oluliselt kitsendas töötulemuste ammendavust ContSys mõistete spetsifiseerimisel meditsiiniliste teadmiste puudumine. Käesoleva töö autor tõlgendas mõisteid nii laiaulatusliku objektiivsuse ja täpsusega kui vähegi võimalik koostöös meditsiiniinformaatika taustaga kaasjuhendajaga. Küll aga tuleb arvestada autori piiratud teadmisi tervishoius ning jurisprudentsi oskust meditsiini ja tervishoiu seaduste tõlgendamisel arhetüüpide kategoriseerimiseks.

Tööst jäeti välja enamik Osapoole ja Osapoole Seose arhetüüpmustrite arhetüübid sobivate spetsifitseeritavate ContSys mõistete puudumise tõttu. Mõistetes käsitletud organisatsioonid võivad olla loodud eesmärgiga teenida raha teenuste/kaupade müügiks [15], kuid iga termini mõistet saab kasutada ka selle vastandina, mistõttu on ContSys mõistete puhul esindatud kindlasti organisatsiooni arhetüüp, aga mitte alati ettevõtte oma. Näiteks tervishoiu tugiorganisatsioon saab olla nii koduhooludust pakkuv organisatsioon (tasuline teenuse pakkumine), aga ka perekond [2]. Samuti on arhetüüpide seas unikaalseid identifikaatoreid esindavad arhetüübid (*RegisteredIdentifier*, *PartyIdentifier*), mis on ContSys valdkonnamõistete jaoks liiga spetsiifilised.

Tööst jäeti välja Osapoole ja Osapoole Seose arhetüüpidega kaudses seoses olevate ContSys terminite analüüs, peaasjalikult peamiselt otseselt protsessidega seotud mõisted (nt varem mainitud raviperiood). Arhetüüpmustrid on üksteisega pidevas sümbioosis, mistõttu käesoleva töö autor säilitas terminite spetsifitseerimisel võimalikult konkreetse ülevaate terminitest tööks valitud Osapoole ja Osapoole Seoste arhetüüpide tähenduses.

4.7 Edasised tööd

Eelneva analüüsi käigus selgus, et praeguse ContSys standardi mõistetel on teataval määral tõlgendamisruumi, mistõttu on ühene terminite spetsifitseerimine arhetüüpmustreid kasutades raskendatud. Mõistete selgitused sõltuvad nende lugejate tõlgendamisoskustest, mitte kindlatest ja üheselt mõistetavates faktidest. See on tingitud ContSysi loomiseprotsessist, kus kaasati põhiliselt meditsiinitaustaga inimesed. Samas tuleb toonitada, et tegu on alles esimese eestikeelse versiooniga ning juba tegeletakse täiendatud versiooni loomisega.

Käesoleva töö eesmärk oli proovida tõlgendada ContSys mõisteid nii palju Osapoole ja Osapoole Seose arhetüüpide keeles kui autor koos juhendajatega antud tingimustes oskasid. Töö ei ole ainuõige ContSys mõistete tõlgendamise allikaks, vaid peamiselt sellele tähelepanu pööramiseks. Sellest lähtuvalt on tehtud töö edasine samm töötulemuste valideerimine meditsiinitöötajatega. Koos omaalaspetsialistidega tervishoius saab praegust mudelit täiendada nii, et tagataks semantiline koostalitlusvõime kõikides tervishoiuasutustes. Järgmiste sammudena näeb autor ette ka projekti täieulatusliku testimist ning tööde tulemuste valideerimisel võib hinnata, kas nimetatud UMLS võiks anda lisandväärtust ContSys mõistete deklaratiivsel spetsifitseerimisel.

5 Kokkuvõte

ContSysi eesmärk on olla masinloetava semantilise koostalitlusvõime nurgakivi meditsiiniinfosüsteemide arendamisel. Arhetüüpidel põhineva ABC4HEDA sobivust oli varem hinnatud ContSys mõistete tõlgendamisel protsessivaatest, kuid Osapoolte ja Osapoolte Seose, millele protsess ülesehitatud on, sobivus oli määramata. Hindamaks ContSys mõistete modelleeritavust Osapoole ja Osapoole Seoste arhetüüpidena realiseerides arhetüübid *Visual Studio´*s *Blazor´*it kasutades, sooviti saavutada valdkonnapõhine spetsifitseerimise keel tervishoiusüsteemide järjepidevuse saavutamiseks. Arhetüüpmustrite realiseerimisel kinnitati SPA arhitektuuri sobivust senise MVC raamistikul põhineva ABC4HEDA suhtes ning lisati seni puudunud kontrollerite testimine.

ContSysi terminite spetsifitseerimisel osapoole arhetüüpmustrites osutus oodatust keerulisemaks, mistõttu käesoleva tööga ei saavutatud täielikult üheselt mõistetavat ja masinloetavat valdkonnapõhist Osapoole ja Osapoole Seose deklaratiivset ja käitlusajal kasutatavat spetsifitseerimise keelt. Autori väljapakutud mõistete arhetüüpsed liigitused küll kinnitavad ContSysi potentsiaali olla patsiendikeskse tervishoiu järjepidevust toetava infosüsteemi kontseptuaalset mudelit kirjeldav standard, kuid praeguse versiooniga ei ole tagatud terminite semantiline koostalitlusvõime infosüsteemi arendamiseks.

Terminite arhetüüpidena spetsifitseerimine meditsiinivaldkonna spetsiifiliste teadmisteta on aluseks mitmeti tõlgendavusele erinevate arhetüüpide vaates, mistõttu on edasiste tööde oluliseim osa tulemuste valideerimine meditsiinitöötajatega, et saavutada üheselt mõistetav ontoloogia ContSys standardi puhul.

Kasutatud kirjandus

[1] T. Benson, G. Grieve, Principles of Health Interoperability: FHIR, HL7 and SNOMED CT, Springer, 2016.

[2] Eesti Standardimis- ja Akrediteerimiskeskus, „Terviseinformaatika, Mõistesüsteem tervishoiu ja arstiabi järjepidevuse toetamiseks, EVS-EN ISO 13940:2016,“ 2023.

[3] J.Bermudez, A.Goni, A.Illarramendi ja M.I..Bagues, “Interoperation among agent-based information systems through a communication acts ontology,” *Elsevier,* 2007.

[4] International Organization for Standardization, “ISO 13940:2015, Health informatics— System of concepts to support continuity of care,” 2015.

[5] V.Presutti ja A.Gangemi, “Content Ontology Design Patterns as Practical Building Blocks for Web Ontologies,” Springer Link, 2008.

[6] T. Sõerd, „Meditsiiniliste protsesside spetsifitseerimine arhetüüpmustrites vastavalt rahvusvahelistele standarditele ja kokkulepetele,“ Tallinn University of Technology Press, Tallinn, 2022.

[7] G.Piho, „Archetypes based techniques for development of domains,,“ Tallinn University of Technology Press, Tallinn, 2011.

[8] K. M. Raavel, „LOINC terminoloogia kasutusele võtmine arhetüüpmustritel põhineval ABC4HEDA alusmudelil,“ Tallinn University of Technology Press, Tallinn, 2022.

[9] R. Randmaa, Äriarhetüüpidel põhinevate metamudelite ja FHIR ressursside semantilise koostalitluse hindamine, Tallinn: Tallinn University of Technology Press, 2022.

[10] R. Randmaa, I. Bossenko, T. Klementi, G. Piho and P. Ross, “Evaluating business meta-models for semantic interoperability with FHIR resources,” in *HEDA-2022: : The International Health Data Workshop, June 19-24, 2022,* Bergen, Norway, 2022.

[11] T. Sõerd, K. Kankainen, G. Piho, T. Klementi and P. Ross, “Specification of Medical Processes in Accordance with International Standards and Agreements,” in *HEDA-2022: The International Health Data Workshop, June 19-24, 2022,* Bergen, Norway, 2022.

[12] T. Sõerd, K. Kankainen, G. Piho, T. Klementi and P. Ross, “Specification of Medical Processes in Accordance with International Standards and Agreements,” in *MODELSWARD 2023: : 11th International Conference on Model-Based Software and Systems Engineering*, Lisbon, Portugal, 2023.

[13] Nicholas o, „All concepts,“ 18 10 2019. [Võrgumaterjal]. Available: https://contsys.org/index. [Kasutatud 14 05 2023].

[14] G. Piho, J.Tepandi ja M. Roost, “Domain Analysis with Archetype Patterns Based Zachman Framework for Enterprise Architecture,” in *International Symposium on Information Technology,* Kuala Lumpur, Malaysia, 2010.

[15] J. Arlow ja I. Neustadt, Enterprise patterns and MDA: building better software with archetype patterns and UML, Addison-Wesley, 2004.

[16] Riigi Teataja, „Vereseadus,“ Riigikogu, 01 04 2023. [Võrgumaterjal]. Available: https://www.riigiteataja.ee/akt/115032014088?leiaKehtiv. [Kasutatud 14 05 2023].

[17] Radzen, „Radzen Blazor Components,“ [Võrgumaterjal]. Available: https://blazor.radzen.com/. [Kasutatud 14 05 2023].

[18] Microsoft, „What's new in ASP.NET Core 7.0,“ 13 03 2023. [Võrgumaterjal]. Available: https://learn.microsoft.com/en-us/aspnet/core/release-notes/aspnetcore-7.0?view=aspnetcore-7.0. [Kasutatud 14 05 2023].

[19] Microsoft, „Entity Framework Core,“ 25 05 2021. [Võrgumaterjal]. Available: https://learn.microsoft.com/en-us/ef/core/. [Kasutatud 14 05 2023].

[20] R. C. Martin, Clean Code: A Handbook of Agile Software Craftsmanship, Pearson Education, Inc, 2009.

[21] K. Beck, Test-driven Development, Addison-Wesley, 2003.

[22] K. Beck, Extreme Programming Explained: Embrace Change, Addison-Wesley, 2000.

[23] E.A. Scott, Jr, SPA Design and Architecture: Understanding single-page web applications, Manning Publications Co, 2016.

[24] Microsoft, „ASP.NET Core Blazor hosting models,“ 04 04 2023. [Võrgumaterjal]. Available: https://learn.microsoft.com/en-us/aspnet/core/blazor/hosting-models?view=aspnetcore-7.0. [Kasutatud 14 05 2023].

[25] L.Kalling, „Toote arhetüüp mustri realiseerimine, testimine ja vastavuse hindamine ISO 13940:2015 (ContSys) põhjal mõistete deklaratiivseks spetsifitseerimiseks,“ 2023.

[26] T. Bates, „Automated Robotic Device for Faster Blood Testing,“ 01 09 2020. [Võrgumaterjal]. Available: https://www.techbriefs.com/component/content/article/tb/pub/briefs/machinery-and-automation/37626. [Kasutatud 14 05 2023].

[27] Ravimiregister, „Ravi omaduste kokkuvõte: Klabax,“ 01 05 2022. [Võrgumaterjal]. Available: https://www.ravimiregister.ee/Data/SPC/SPC_1042670.pdf. [Kasutatud 14 05 2023].

[28] Tervise ja Heaolu Infosüsteemide Keskus, „Tahteavaldused,“ Patsiendiportaal, [Võrgumaterjal]. Available: https://www.digilugu.ee/. [Kasutatud 14 05 2023].

[29] Riigi Teataja, „Perekonnaseadus,“ Riigikogu, 01 02 2023. [Võrgumaterjal]. Available: https://www.riigiteataja.ee/akt/13330603?leiaKehtiv. [Kasutatud 14 05 2023].

[30] Riigi Teataja, „Isikuandmete kaitse seadus,“ 15 01 2019. [Võrgumaterjal]. Available: https://www.riigiteataja.ee/akt/104012019011?leiaKehtiv. [Kasutatud 14 05 2023].

[31] Riigi Teataja, „Tervishoiuteenuste korraldamise seadus,“ Riigikogu, 01 04 2023. [Võrgumaterjal]. Available: https://www.riigiteataja.ee/akt/110032011009?leiaKehtiv. [Kasutatud 14 05 2023].

[32] Riigi Teataja, „Sotsiaalhoolekande seadus,“ Riigikogu, 01 02 2023. [Võrgumaterjal]. Available: https://www.riigiteataja.ee/akt/130122015005?leiaKehtiv. [Kasutatud 14 05 2023].

[33] Microsoft, „ASP.NET Core Blazor,“ 24 02 2023. [Võrgumaterjal]. Available: https://learn.microsoft.com/en-us/aspnet/core/blazor/?view=aspnetcore-7.0. [Kasutatud 14 05 2023].

[34] M.Aponte, Building Single Page Applications in .NET Core 3: Jumpstart Coding Using Blazor and C#, Apress, 2020.

[35] P. Raj, A. Raman, H. Subramanian, Architectural Patterns, Packt Publishing, 2017.

[36] Riigi Teataja, „Nakkushaiguste ennetamise ja tõrje seadus,“ Riigikogu, 01 04 2023. [Võrgumaterjal]. Available: https://www.riigiteataja.ee/akt/12824817?leiaKehtiv. [Kasutatud 14 05 2023].

[37] E. Gamma, R. Johnson, J. Vlissides ja R. Helm, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1994.

[38] G.Piho, Interviewee, *Eravestlus raviperioodi teemal.* [Intervjuu]. 20 04 2023.

[39] R.Garcia-de-Leon-Chocano, C.Saez, V. Munoz-Soler jt, “Construction of quality-assured infant feeding process of care data repositories: definition and design (Part 1),” *Elsevier,* pp. 95-103, 2015.

[40] L.D. Serbanati, “Health digital state and Smart EHR systems,” *Elsevier,* 2020.

[41] T. Beale, “Archetypes: Constraint-based Domain Models for Future-proof Information Systems,” in *OOPSLA*, 2002.

[42] C.Lemmen, P.Oosterom,R.Bennett, “The Land Administration Domain Model,” *Elsevier,* pp. 535-545, 2015.

[43] P.Oosterom, C.Lemmen, T.Ingvarsson jt, “The core cadastral domain model,” *Elsevier,* pp. 627-660, 2006.

[44] Tervisekassa, „Digilugu.ee keskkond uueneb- tule anna oma panus!,“ 8 11 2022. [Võrgumaterjal]. Available: https://www.tervisekassa.ee/blogi/digiluguee-keskkond-uueneb-tule-anna-oma-panus. [Kasutatud 18 05 2023].

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

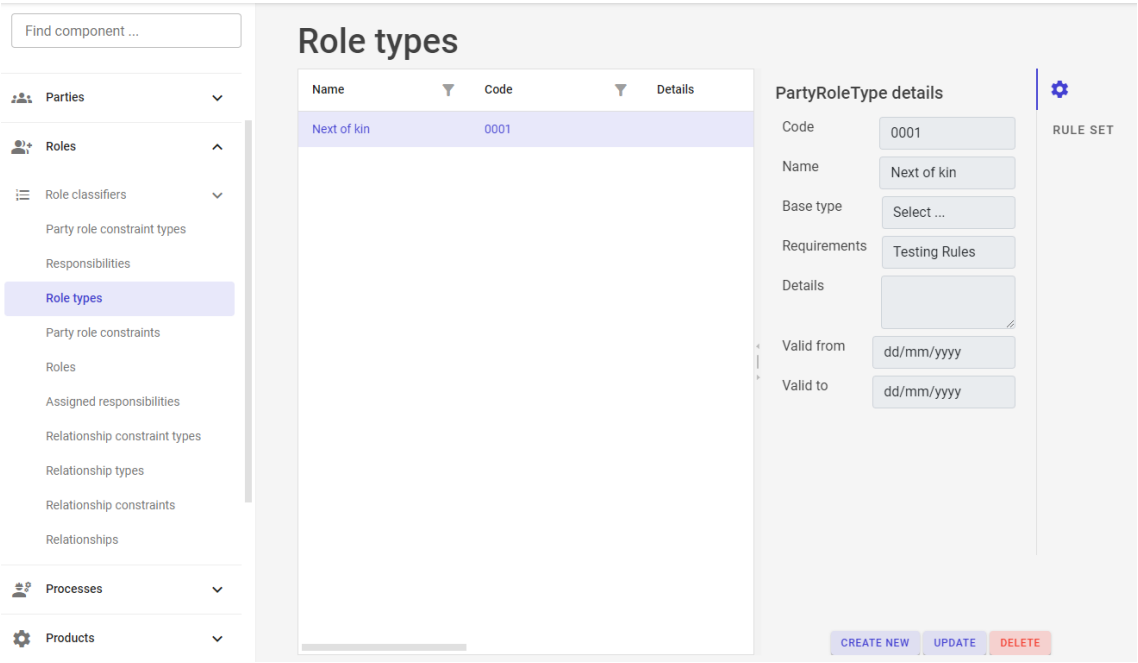
Mina, Jane-Ly Buhvestova

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Osapoole ja Osapoole Seose arhetüüpmustrite realiseerimine, testimine ja sobivuse hindamine ISO 13940:2015 (ContSys) mõistete deklaratiivseks spetsifitseerimiseks“, mille juhendaja on Toomas Klementi ja kaasjuhendaja Gunnar Piho.
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

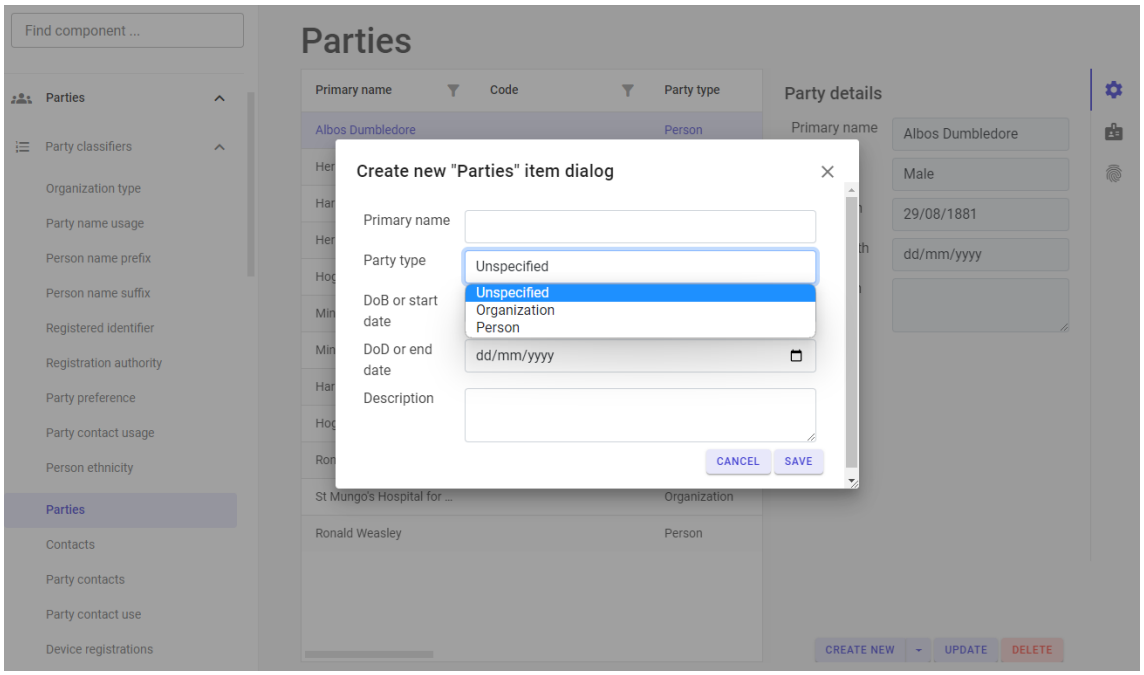
17.05.2023

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

LISA 1 – *RoleType* vaade kasutajaliideses



LISA 2 - Uue kindlaks määramata osapoole (*Unspecified*) lisamine kasutajaliideses



LISA 3 – *GetTest* implementeerimine

```
[TestMethod]
public async Task GetTest() {
        var l = await controller.Get(skip, take, filter, fixedFilter,
fixedValue, orderBy);
        areEqual(filter, controller.repo.DynamicLinqFilter);
        areEqual(fixedFilter, controller.repo.FixedFilter);
        areEqual(fixedValue, controller.repo.FixedValue);
        areEqual(skip / controller.repo.PageSize + 1,
controller.repo.PageIndex);
        areEqual(take, controller.repo.PageSize);
        areEqual(orderBy, controller.repo.SortOrder);
        areEqual(0, l.Count());
}
```

LISA 4 - *RepoIs(Not)NullGetTest* implementeerimine

```
[TestMethod]
public async Task RepoIsNullGetTest() {
        var l = await controller.get((IRepo<TObject>)null, skip, take,
filter, fixedFilter, fixedValue, orderBy);
        areEqual(null, controller.repo.DynamicLinqFilter);
        areEqual(null, controller.repo.FixedFilter);
        areEqual(null, controller.repo.FixedValue);
        areEqual(0, controller.repo.PageIndex);
        areEqual(0, controller.repo.PageSize);
        areEqual(null, controller.repo.SortOrder);
        areEqual(0, l.Count());
}
[TestMethod]
public async Task RepoIsNotNullGetTest() {
        var l = await controller.get(controller.repo, skip, take, filter,
fixedFilter, fixedValue, orderBy);
        areEqual(filter, controller.repo.DynamicLinqFilter);
        areEqual(fixedFilter, controller.repo.FixedFilter);
        areEqual(fixedValue, controller.repo.FixedValue);
        areEqual(skip / controller.repo.PageSize + 1,
controller.repo.PageIndex);
        areEqual(take, controller.repo.PageSize);
        areEqual(orderBy, controller.repo.SortOrder);
        areEqual(0, l.Count());
}
```

LISA 5 - *GetCountRepoIs(Not)NullTest* implementeerimine

```
[TestMethod]
public async Task GetCountRepoIsNullTest() {
        var count = await controller.getCount(null, filter, fixedFilter,
fixedValue);
        areEqual(0, count);
}
[TestMethod]
public async Task GetCountRepoIsNotNullTest() {
        var count = await controller.getCount(controller.repo, filter,
fixedFilter, fixedValue);
        areEqual(filter, controller.repo.DynamicLinqFilter);
        areEqual(fixedFilter, controller.repo.FixedFilter);
        areEqual(fixedValue, controller.repo.FixedValue);
        areEqual(0, count);
}
```

LISA 6 - *BodyMetrics* arhetüübile vastavad ContSys terminid

BodyMetrics arhetüüp	
ContSys mõiste	**ContSys mõiste eestikeelne vaste**
Certificate related to a healthcare matter	Terviseküsimusega seotud tõend
Considered (health) condition	Arvesse võetav seisund
Discounted condition	Välistatud seisund
Entry	
Excluded condition	Välistatud seisund
Health condition	Terviseseisund
Health issue	Tervisemure
Health need	Tervisevajadus
Health problem	Terviseprobleem
Health state	Terviseolek
Input health state	Algne terviseolek
Intended outcome	Sihtseisund
Non-verified condition	Välistatud seisund
Observed (health) condition	Täheldatud seisund
Outcome	Saavutatud terviseolek
Output health state	Saavutatud terviseolek
Potential health condition	Võimalik seisund
Professional health record	Erialane terviselugu
Professionally assessed (health) condition	Tervishoiutöötaja hinnatud seisund
Prognostic (health) condition	Ennustatav seisund
Resultant condition	Saavutatud seisund
Risk (health) condition	Riskiseisund
Ruled out (considered) condition	Välistatud seisund
Target (health)condition	Sihtseisund
Working diagnosis/hypothesis	Töödiagnoos

LISA 7 – Varasemate arhetüübimääratluste kriitiline hindamine

ContSys mõiste	**ContSys mõiste eestikeelne vaste**	**Varasem arhetüübimääratlus**	**Käesoleva töö autori määratlus**
Healthcare actor	Tervishoiu osapool	*PartyRole*	*Party*
Healthcare commitment	Tervishoiuteenuse osutamise kohustus	*PartySignature*	*Responsibility*
Healthcare funds	Tervishoiu rahalised vahendid	*PartyRoleType*	*ProductType*
Healthcare personnel	Meditsiinipersonal	*Person*	*Party*
Healthcare resource	Tervishoiuressurss	*Party*	Ei sobi *Party* ega *PartyRelationship* arhetüüpidest ükski
Medical device	Meditsiiniseade	*Party*	*Product*