KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

Žygimantas Legas

VERSLO TRANSAKCIJŲ MODELIAVIMAS KOMUNIKACINĖMIS KILPOMIS

Magistro darbas

Vadovas
doc. dr. B. Paradauskas

KAUNAS, 2007

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
INFORMACIJOS SISTEMŲ KATEDRA

VERSLO TRANSAKCIJŲ MODELIAVIMAS KOMUNIKACINĖMIS KILPOMIS

Magistro darbas

Vadovas
doc. dr. B. Paradauskas

Recenzentas
doc. dr. R. Misevičienė

Atliko
IFN 4/1 gr. studentas
Ž. Legas

KAUNAS, 2007

KVALIFIKACINĖ KOMISIJA

Pirmininkas	doc. dr. Kazys Kavaliauskas, UAB „Baltic Software Salutions“ generalinis direktorius
Sekretorius	Antanas Lenkevičius, docentas
Nariai	Jonas Kazimieras Matickas, docentas
	Bronius Paradauskas, docentas
	Vytautas Rėklaitis, docentas
	Dalius Rubliauskas, docentas
	Danguolė Rutkauskienė, docentė
	Irma Šileikienė, VGTU docentė
	Aleksandras Targamadzė, profesorius

Business transaction modeling using communicative action loops

SUMMARY

In changing world of e-commerce and internet based communication, business transactions and data flows play a key role in successful business management. It is vital to formalize business transactions. Communicative action loop was proposed for modeling these transactions. XML is often used for data exchange. BTP is a specific extended transaction model that allows coordination of resources which are exposed by multiple autonomous organizations. This model relaxes the traditional ACID properties and forms a protocol that can run for long periods of time over the inherently unreliable environment that is the Internet. This project aims to demonstrate if the .NET technology is a sufficiently flexible model to provide an implementation of BTP and whether the functionality provided by the framework is enough to support the complex interactions specified by the protocol.

TURINYS

Lentelių sąrašas

Paveikslėlių sąrašas

ĮVADAS

Kad verslo projektas būtų ėkmingas reikia idėjos, žmogiškųjų resursų, finansinių išteklių ir gero laiko planavimo. Šiuolaikinis verslas su kasdien tampa vis sudėtingesnis, nepastovesnis, reikalauja daugybės įvairių resursų. Informacinės verslo sistemos privalo greitai ir lanksčiai prisitaikyti prie tokių pokyčių. Elektroninių ir mobilių paslaugų veržimasis dar labiau sustiprino šią tendenciją, kadangi organizacijos vis dažniau bendrauja elektroniniu būdu. Projektuojant ir atnaujinant informacines sistemas, svarbu tinkamai valdyti duomenis ir jų srautus, užtikrinant jų išsamumą ir integralumą, saugumą ir atnaujinimą. Kadangi šiuolaikinėms verslo sistemoms bendrauti naudojamos verslo transakcijos, šiame darbe pabandysime panagrinėti jas bei jų modeliavimo metodus.

Elektroniniame versle ypač svarbus tikslus verslo procesų apibrėžimas. Būtina juos tiksliai specifikuoti ir formaliai apibrėžti. Kintant funkciniams reikalavimams, kinta ir duomenų struktūros, todėl lygiai taip pat svarbūs duomenų formato apibrėžimai. Verslo procesai neatsiejami nuo apsikeitimo duomenimis, todėl itin reikšmingas informacinėse sistemose yra verslo transakcijų palaikymas. Duomenys saugomi duomenų bazėse, todėl atsiranda poreikis susieti verslo transakciją su duomenų bazės transakcija. Tačiau verslo transakcijoms vienareikšmiškai taikyti duomenų bazių transakcijų savybes ne visada įmanoma. Tai susiję su pastarųjų daug trumpesnetrukme, bei verslo transakcijų atnaujinimu: jiems dažnai būdingi išoriniai efektai, dėl kurių atnaujinimas gali būti nepageidaujamas arba iš viso neįmanomas. Darbe apžvelgiami įvairūs verslo transakcijų specifikavimo modeliai, o plačiau nagrinėjamas komunikacinių kilpų metodas, kurio pagrindas – komunikacinės binarinės kilpos, susietos pragmatiniu ir semantiniu priklausomumu. Kiekviena tokia kilpa susideda iš dviejų dalyvių, kurie turi tikslus, iš dviejų dalyvių, vykdomų procesų bei duomenų srautų. Tokia kilpa specifikuoja binarinę verslo transakciją.

Bet koks apsikeitimas duomenimis reikalauja duomenų apibrėžties protokolo. Pastaruoju metu itin populiarus yra XML standartas, nes tai yra plačiai naudojama standartizuota kalba, nepriklausoma nuo informacinių sistemų architektūrinių sprendimų. XML yra universalus formatas: galima saugoti įvairių tipų ir formų duomenis. XML priemonėmis įmanoma atlikti loginius veiksmus ir t.t.

Šiame darbe nagrinėjamas verslo procesų modeliavimas, kuriuo siekiama automatizuoti veiklos valdymą, išskiriant darbų sekos modeliavimą kaip vieną iš pagrindinių IS kūrimo etapų. Didžiausias dėmesys yra kreipiamas į išorinių verslo procesų modeliavimą, t.y. į B2B (angl. Business – To- Business) transakcijas, kitaip tariant, verslo partnerių keitimąsi duomenimis internetu. Jų verslo procesai specifikuojami komunikacinėmis kilpomis, o architektūros realizacijai pasirinktas trijų lygių sprendimas: duomenų bazė, verslo logika, (pateiktos verslo taisyklės) bei atvaizdavimo lygiai. Kaip buvo minėta, XML yra universalus duomenų formatas, todėl juo remdasi OASIS (angl. Organization for the

Advancement of Structured Information Standards), t.y. struktūrizuotų informacinių standartų pažangos organizacija, aprašė ir sukūrė verslo transakcijų protokolą BTP (angl. Business transaction protocol), kurio viena iš praktinių realizacijų atlikta ir nagrinėjama šiame darbe naudojant komunikacines kilpas. Darbe verslo transakcijos protokolo realizacija atlikta. NET technologija C# programavimo kalba. Naudodami trijų lygių architektūrą (verslo transakcijų apdorojimą atlieka verslo sudedamoji dalis), viduriniame lygyje apdorodami duomenis, galime juos parengti saugoti duomenų bazėje ar siųsti, (priklauso nuo verslo transakcijos būklės).

Darbe atliekamas eksperimentinis tyrimas pasirinkus dalykinę sritį: el. parduotuvę ir joje esančių prekių pirkimą išsimokėtinai sudarant sutartį su išperkamosios nuomos kompanija. El. parduotuvė – tai paprastos parduotuvės analogas internetinėje erdvėje. Galima pasirinkti norimas prekes, jų kiekį ir pan. Už prekes atsiskaitoma įvairiais būdais: standartiniu tokioms parduotuvėms būdu – kreditinėmis kortelėmis, banko pavedimu ir panašiai. Eksperimente didžiausias dėmesys kreipiamas į komunikacinėmis kilpomis specifikuotų verslo transakcijų realizaciją trijų lygių architektūroje.

1. VERSLO TRANSAKCIJŲ MODELIAVIMO METODŲ ANALIZĖ

1.1. Verslo transakcijų apžvalga

Šiuolaikinis verslas vis labiau tampa priklausomas nuo informacinių technologijų. Vis svarbesni tampa informacijos mainai tarp verslo partnerių. Norint užtikrinti saugų ir patikimą duomenų perdavimą ir priėmimą, korektišką verslo informacinių sistemų veikimą, būtinas verslo transakcijų palaikymas. Tai ypač svarbu informacinėse sistemose, kurios yra susijusios su organizacijų darbų sekos valdymu bei suistemomis „Verslas - klientui B2C“ (angl. Business-To-Customer) ar „Verslas – verslui“ B2B (angl. Business-To-Business). Būtent į pastarąsias kreipiamas didžiausias dėmesys šiame darbe.

Nors nėra sutarta dėl formalaus verslo transakcijos apibrėžimo, tačiau paprastai suprantama, kad tai veiksmų tarp verslo partnerių aprašymas. Šiais veiksmais partneriai siekia realizuoti savus tikslus. Paskirstyto verslo proceso realizacija suprantama kaip tokių transakcijų visuma.

Kadangi elektroninio verslo procesas susideda iš veiksmų, kurie naudoja duomenis iš daugialypių duomenų bazių, jis turi būti siejamas su transakcijų semantika. Verslo procesams kaip ir transakcijoms gali būti taikomos ACID (angl. Atomicity, Consistency, Isolation, Durability), t.y. nedalumo, vientisumo, izoliavimo, išliekamumo savybės. Šios savybės gali būti taikomos ir proceso dalims. Tačiau visą verslo procesą traktuoti kaip ACID transakciją nėra tikslinga. Visų pirma, verslo procesams būdinga ilga trukmė, todėl procesą laikant transakciją reikėtų daug laiko atžvilgiu įšaldytų išteklių. Antra, verslo procesai dažniausiai būna susieti su daugialypėmis ar paskirstytomis duomenų bazėmis, todėl didelių sąnaudų pareikalautų šių bazių koordinavimas. Trečia, verslo procesams būdingi išoriniai efektai, dėl kurių būsenos atnaujinimo (angl. rollback) mechanizmai gali tapti nepageidaujami arba net visiškai semantiškai neįvykdomi. Todėl įvairūs šaltiniai pateikia įvairius išplėstinius transakcijų modelius. Pavyzdžiui, Saga modelyje [1] transakciją siūloma sudaryti iš elementarių transakcijų grandinės, kurioje kiekviena elementari grandinės transakcija turi ir savo kompensacinę transakciją. Jei elementari transakcija neįvykdoma teisingai, tai grąžinama ankstesnė būsena, o valdymas perduodamas atitinkamai kompensacinei transakcijai. Veiklos transakcijų modelyje ATM (angl. Activity transaction model) transakcijos gali būti skaidomos į subtransakcijas, o šios jungiamos į grandines (angl. chained) arba į rinkinius (angl. nested). Rinkinio transakcijos vykdomos lygiagrečiai, o klaidoms ir išimtims apdoroti yra taikoma hierarchinė kontrolė. Pirmasis rinkinių modelis [2] apibrėždavo tik uždarąsias subtransakcijas, tačiau vėliau buvo išplėstas [3] atvirtosioms subtransakcijoms palaikyti. Uždarosios subtransakcijos rezultatus patvirtina pirminė transakcija, o tokie daliniai rezultatai visiškai patvirtinami tik tada, kai yra patvirtinama viršutinė (šakninė) transakcija. Taip yra užtikrinami nedalumo ir izoliavimo principai visai

transakcijai. Tuo tarpu atvirosios subtransakcijos nepaiso izoliavimo principo ir jų rezultatai patvirtinami iš karto. Klaidų apdorojimas yra hierarchinis. Jei subtransakcija nepavyksta, siunčiamas pranešimas pirminei transakcijai, o ši sprendžia, ar vykdyti klaidos apdorojimą ir pabandyti kartoti antrinę subtransakciją , vykdyti kompensuojamą transakciją ar perduoti klaidos valdymą aukštesnei transakcijai. Klaidos valdymo perdavimas reikalauja kai kurių jau patvirtintų transakcijų kompensavimo. Kai kurios užduotys yra gyvybiškai svarbios, taigi klaida jose reiškia, jog visos žemesnės subtransakcijos yra klaidingos. Šaltinyje [4] pateikiami įvairūs ATM modeliai ir transakcijų semantikos formuluotės. „Pažymėtina, kad išplėstiniai transakcijų modeliai pernelyg riboja paslaugų uždavinius, išskirstytus laiko ir erdvės atžvilgiu“ [5].

Visi minėti modeliai yra tinkami ne tik internetiniams procesams specifikuoti. Kadangi B2B sistemoms gyvybiškai svarbu funkcionuoti elektroninėje erdvėje, yra siūloma keletas internetui skirtų transakcijų protokolų. Pavyzdžiui, interneto transakcijų protokolas TIP (angl. Transaction Internet Protocol) [6]. Tai yra paprastas dviejų patvirtinimo fazių protokolas, pagrįstas dviem TCP/IP susijungimais. Palyginus su kitais internetinėms aplikacijoms skirtais metodais, jis turi dvi naujoves. Pirma - įtraukimo (angl. pull) metodas. Tai nauja subtransakcija, kuri gali būti įtraukiama į bendrą transakciją jos vykdymo metu. Antra – perduodamas patvirtinimas (angl. delegated commit): transakcijos apribojimai yra kontroliuojami iš kliento taikomosios programos, o tarnybinė stotis tik patvirtina ir atšaukia. Toks siūlymas labai tinka internetinėms taikomosioms programoms, kur kliento pusėje nėra atnaujinamų resursų.

1.1.1. Komerciniai standartai

Yra vykdoma ir keletas komercinių projektų, susijusių su verslo transakcijų modeliavimu. Pvz., Exotica projektas [7, 8], kuriame buvo pasiūlyti metodai ir įrankiai transakcijoms modeliuoti. Pagrindinė jo užduotis – pasiūlyti vartotojams išplėstinį darbų sekos modelį, grįstą transakcijų samprata. Vartotojas specifikuoja verslo ir jas kompensuojančias užduotis. Procesorius pakeičia šiuos aprašus į FDL (angl. Flowmark Definition Language) kalbą, į reikiamas vietas (po vienos ar kelių užduočių) įterpdamas jas kompensuojančių užduočių aprašus. Pažymėtina, kad tai yra IBM kompanijos produktas ir skirtas būtent šios kompanijos platformoms ir sprendimams.

Toks Krishnamoorty V. ir Shan M.C pasiūlytas transakcijų modelis skirtas *HP Changengine* (dabar *Process Manager*). Šiame modelyje aprašomos virtualios transakcijos VT (angl. Virtual Transaction) apibendrinančios tam tikrą darbų seką. Jei įvyksta klaida vykdant kurią nors darbų sekos užduotį, tada visa apibendrinamoji seka grąžinama į prieš tai buvusią, kol pasiekiamas galutinis kompensacijos taškas (angl. compensation end point). Tada sistema sprendžia ar kartoti vykdymą nustatytą kiekį kartų, ar vykdyti alternatyvias transakcijas ar baigti visą procesą [9].

1.1.2. Pagrindinės transakcijų sąvokos

Šis skyrius supažindina su transakcijos koncepcija, apžvelgia tradicinį ACID modelį ir kaip jis gali būti siejamas su verslo transakcijomis.

Transakcija – sąveika, dažnai verslo vieneto ir objekto, kai vyksta apsikeitimas informacija. Apsikeitime dalyvaujantys objektai gali būti bet kas: pinigai, pasiūlymai, tai priklauso nuo konkrečios situacijos ir dalykinės srities. Nedalomosios transakcijos yra naudojamos operacijų kontrolei su išliekamąja informacija. Nedaliosios, kitaip atominės, transakcijos užtikrina paprastą „sėkmės“ arba „nesėkmės“ modelį, t.y. atominė transakcija gali prieš tai visus įvykdytus veiksmus arba patvirtinti, arba atmesti. Paprasčiausias atominės transakcijos pavyzdys - kliento noras paimti pinigų iš savo sąskaitos naudojantis bankomato teikiamomis paslaugomis. Visi veiksmai turi būti atlikti transakcijos ribose, tam, kad jei aparatas užstrigtų, ar bankomate nebūtų pinigų, pinigai iš kliento sąskaitos nebūtų nurašyti. Tokioms transakcijoms aprašyti yra žinomos klasikinės savybes ACID (angl. Atomicity, Consistency, Isolation, Durability).

- **Nedalumas** - tai reiškia, kad transakcija turi būti atominė (viskas arba nieko) arba įvykdoma arba ne. Jei nors viena dalis nepavyko, kitos likusios transakcijos programos dalys negali būti įvykdytos. Jei dėl kokių nors priežasčių transakciją negali būti užbaigta, visi pakeitimai tūri būti grąžinti į ankstesnę būseną iki tol, kol buvo pradėta minima transakcija panaudojant transakcijos operacija „grąžinti“ (angl. rollback).
- **Vientisumas** – bendri transakcijos resursai turi išlikti vientisi. Transakcija turi pervesti sistemą iš vienos vientisos būsenos į kitą. Vientisumo būsenos prasmė yra ta, kad tam tikros verslo taisyklės buvo išpildytos, pvz., pinigų likutis sąskaitoje yra didesnis, nei klientas pareikalavo.
- **Izoliacija** – kiekviena transakcija gali pasiekti resursus, jei nėra kitų konkurencinių transakcijų; transakcija įvykdoma be jokių įsikišimų iš kitų transakcijų ir turi išskirtinį priėjimą prie resursų. Transakcijos nuosekliai vykdomos viena paskui kitą ir jos nemato kitu transakcijų rezultatų tol, kol jos nepabaigiamos.
- **Išliekamumas** –transakciją baigus, visos būsenos ir atlikti pakeitimai turi išlikti duomenų saugykloje.

Jei transakcija vykdoma vietinėje ar paskirstytoje aplinkoje, privalo būti palaikomos ACID savybės. Tai sunkiau pasiekiama, jei transakcijos resursai yra skirtingose mašinose. Kai transakcija tūri būti patvirtinama, reikia įsitikinti, kad pakeitimai buvo atlikti abiejose sistemose. Pagrindinė problema, su kuria susiduriama vykdant transakcijas paskirstytoje aplinkoje, yra ta, kad bet kuris mazgas gali patirti

nesėkmę dėl įvykusios klaidos, kadangi duomenys yra fiziškai skirtingose mašinose, nesėkmę patyrusio mazgo aptikimas yra sudėtingas. Pavyzdžiui, jeigu transakcija patvirtina pakeitimus vienoje sistemoje, bet kitoje sistemoje įvyko klaida, prieš patvirtinant transakciją ji turėtų būti pervesta į nepilnumo būseną. Tokiu atveju dalis darbo buvo atlikta sėkmingai, kita pusė – ne. Tai prieštarauja nedalumo savybei, kuria apžvelgėme anksčiau. Šios problemos sprendimas būtų patvirtinimo (angl. Commit) protokolo panaudojimas. Dažniausiai pasitaikantis patvirtinimo protokolas transakcijų sistemose yra vadinamas dviejų fazių patvirtinimu (angl. Two phase commit), o jei reikia dar didesnio patikimumo naudojamas trijų fazių patvirtinimo protokolas.

Dviejų patvirtinimo fazių (2PF) protokolas yra paprastas protokolas, leidžiantis pasiekti vienodumą tarp visų šalių, dalyvaujančių transakcijoje, kol transakcija neužbaigta. Šiame protokole dalyvauja du procesai.

- **Koordinatorius** (angl. Coordinator). Jis sprendžia, kada vykdyti globalų patvirtinimą arba atnaujinimą, tai daro siųsdamas pranešimus dalyviams ir laukdamas iš jų atsakymo.
- **Dalyviai** (angl. Participants). Jie iš esmės yra resursų priešaky, transakcijos metu bando atlikti pakeitimus resursuose. Dalyviai nutaria, ar patvirtinti, ar grąžinti pakeitimus, atliktus resursuose pagal pranešimus, gautus iš koordinatoriaus, kai transakciją nusprendžiama patvirtinti.

Patvirtinimo sprendimas atliekamas laikantis globalios patvirtinimo taisyklės (angl. Global commit rule):

- jeigu nors vienas dalyvis liepia nutraukti transakciją, koordinatorius turi priimti globalaus nutraukimo sprendimą (angl. Global abort decision);
- jeigu visi dalyviai liepia patvirtinti transakciją, koordinatorius turi priimti globalaus patvirtinimo sprendimą (angl. Global commit decision).

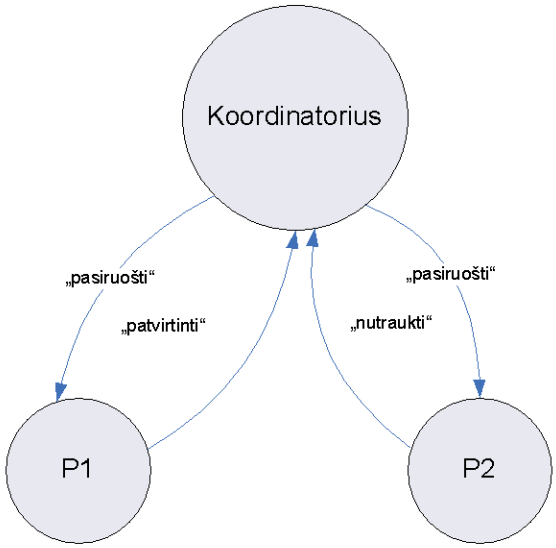
1.1.3. Dviejų patvirtinimo fazių protokolas

Dviejų patvirtinimo fazių (kaip teigia pats pavadinimas) protokolas susideda iš dviejų fazių. Žemiau pateikiamas tipiškas jų aprašymas.

- **Pirmoji fazė.** Pradedamas koordinatoriaus procesas, (dažniausiai ten, kur transakcija yra sukuriama pirmą kartą), įrašomas „*pradėti*“ (angl. Begin) įrašas į peržiūros žurnalą, toliau siunčiamas „*pasiruošti*“ (angl. Prepare) pranešimas dalyviams ir pereinama į laukimo būseną. Siunčiamas pranešimas taip pat gali turėti unikalų transakcijos identifikatorių (TID), kurį toliau naudos visi pranešimai. Kai dalyvis gauna „*pasiruošti*“ (angl. Prepare) pranešimą, jis

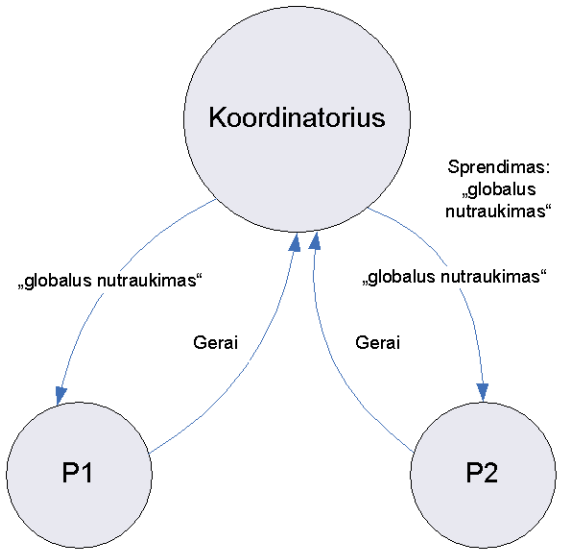
patikrina, ar gali patvirtinti transakciją. Jeigu dalyvis gali, patvirtinti transakciją, įrašo „*pasiruošęs*“ (angl. Ready) įrašą į savo peržiūros žurnalą, siunčia „*patvirtinti*“ (angl. Vote commit) pranešimą koordinatoriui ir pereina į būseną „*pasiruošęs*“ (angl. Ready). Kitu atveju, vienareikšmiškai nusprendęs nutraukti transakciją, dalyvis įrašo „*nutraukti*“ (angl. Abort) įrašą į savo peržiūrų žurnalą ir siunčia „*nutraukti*“ (angl. Vote abort) pranešimą koordinatoriui. Jis pereina į „*nutraukti*“ (angl. Abort) būseną ir toliau transakcijos neseka.

- **Antroji fazė** – Koordinatorius gavęs pranešimus iš visų dalyvių, sprendžia ar transakciją patvirtinti, ar nutraukti pagal globalią patvirtinimo taisyklę (angl. Global commit rule), ir įrašo savo sprendimą į savo peržiūrų žurnalą. Jeigu sprendimas yra pravirtinti, tuomet siunčiamas „*globalaus patvirtinimo*“ (angl. Global commit) pranešimas visiems dalyviams, kitu atveju siunčiamas „*globalaus nutraukimo*“ (angl. Global abort) pranešimas tiems dalyviams, kurie pranešė, kad jie pasirengę patvirtinti transakciją. Galiausiai koordinatorius įrašo transakcijos pabaigą savo peržiūrų žurnale, o dalyviai pabaigia transakciją remdamiesi koordinatoriaus sprendimu ir įrašo rezultatus savo peržiūrų žurnaluose.



1.1 pav. Koordinatoriaus ir dalyvių schema

Schemoje pateiktas dviejų patvirtinimo fazių su dviem dalyviais veikimo pavyzdys. Koordinatorius išsiunčia abiem dalyviams „*pasiruošti*“ pranešimus, P1 siunčia „*patvirtinti*“, o P2 siunčia „*nutraukti*“ pranešimus. Pagal globalaus patvirtinimo taisyklę visi dalyviai turi atsiųsti „*patvirtinti*“ pranešimus. Tokiu atveju P2 siunčia „*nutraukti*“, todėl koordinatorius nusprendžia tokią transakciją nutraukti ir išsiunčia „*globalaus nutraukimo*“ pranešimą.



1.2 pav. Sprendimo pateikimas dalyviams

Dviejų patvirtinimo fazių protokolas užtikrina, kad būtų panaudotos ACID savybės paskirstytoje aplinkoje, net jei mazgas patyrė nesėkmę toli nuo koordinatoriaus.

1.1.4. ACID savybių pakankamumas

Paskirstyti objektai ir ACID transakcijos užtikrina resursų ir jų būsenos pilnumo koordinavimą. Tai dažniausiai pasitaikantis transakcijų panaudojimo atvejis, tačiau ACID transakcijos yra trumpalaikės. ACID transakcijos nėra pritaikytos ilgai funkcionuoti, tam reikalui yra naudojamos išplėstinės transakcijos (angl. extended transactions), kurios gali trukti minutes, valandas kartais ir net dienas.

Susiduriama su keliomis problemomis, naudojant ilgalaikes ACID transakcijas.

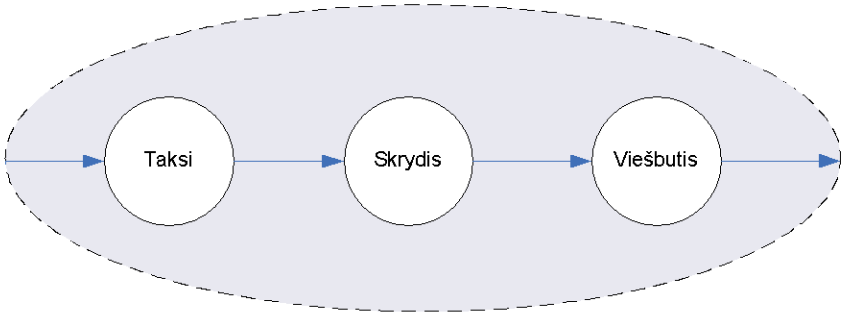
- Ilgalaikės transakcijos sumažina konkurencingumo problemą sistemoje iki nepriimtino lygio, užrakindamos resursus iki tol, kol transakcija tęsis, tai yra tol, kol ji nebus patvirtinta arba atmesta. Tai reiškia, kad jokia kita transakcija negaus priėjimo prie šių resursų.
- ACID transakcijos yra atominiai vienetai, tai yra viskas arba nieko, kai kuriais atvejais tas nebūtina, kai reikalingas koks nors loginis pasirinkimas.

Jei transakcija yra ilgalaikė, jos griežtos ACID savybės gali būti truputi palengvintos. Tai yra ne visais atvejais, jų laikytis reikia griežtai ir tai panaudojama išplėstinėse transakcijose. Verta paminėti, kad šios transakcijos gali turėti subtransakcijų, kurios palengvina operaciją. Tokios transakcijos susideda iš pagrindinės ir daugybės subtransakcijų.

1.1.5. Išplėstinės transakcijos

Išplėstinės transakcijos gali būti naudojamos, kai reikia naudoti ilgalaikes transakcijas: tuomet ACID savybėms taikomi žemesni reikalavimai. Išplėstinė transakcija gali būti struktūriškai apibrėžta kaip daugelis trumpalaikių aukščiausio lygio transakcijų. Taigi resursai, naudojami trumpalaikių aukščiausio lygio transakcijų metu, yra užrakinami tik tol, kol ta transakcija vykdoma, o ne visos išplėstinės transakcijos metu. Išplėstinės transakcijos samprata panaikina anksčiau aptartas problemas, t.y. resursai nėra laikomi užrakinti visos ilgalaikės transakcijos metu, tokiu būdu kitos transakcijos gali naudotis bendrais resursais, kol jie nėra užrakinti. Tai pagerina sistemos konkurencingumą ir efektyvumą.

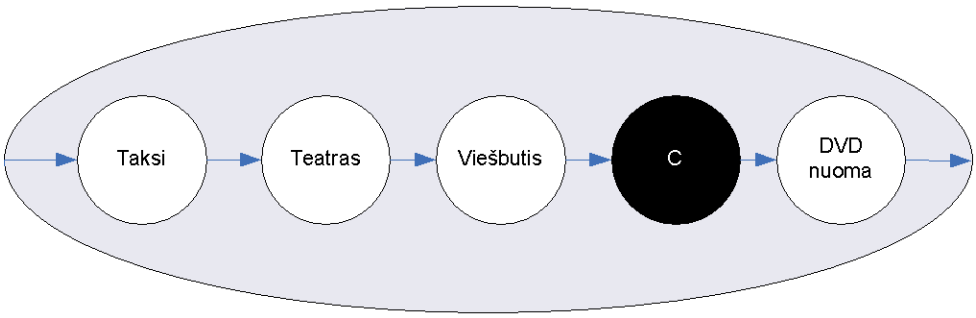
Ilgalaikės transakcijos gali būti sudarytos iš nuoseklių trumpalaikių aukščiausio lygio transakcijų (žr. žemiau esančią diagramą).



1.3 pav. Ilgalaikė loginė transakcija

Ši diagrama vaizduoja užsakymą atostogoms. Klientas nori užsakyti taksi į oro uostą, skrydį į vietovę ir viešbutį, kuriame bus apsistota, kaip vieną atominę transakciją. Tai reiškia, kad visi užsakymai turi tenkinti kliento poreikius ir būti įvykdyti, arba jei nors vienas netiks, transakcija turėtų būti nutraukta. Tokiu atveju individualių resursų bus naudojama tiek, kiek reikia vienai individualiai trumpalaikei transakcijai įvykti.

Išplėstinės transakcijos taip pat gali toliau vykti netgi tuomet, kai viena iš trumpalaikių transakcijų nepavyksta. Galima veiksmų seka pavaizduota žemiau esančioje diagramoje.



1.4 pav. Tolesnė transakcija

Nagrinėjant pateiktą situaciją, vykdoma tokia veiksmų seka: užsakyti kambario neįmanoma nes visi kambariai jau užsakyti, taigi tokia seka negali būti užbaigta. Bet galima pakeisti sekos situaciją, jei klientas nusprendžia pasilikti namie ir išsinuomoti DVD diską. Tai gali būti atlikta, jei visas atliktas darbas t.y. (taksi ir teatro užsakymai) buvo atstatytas. Tokį efektą galima gauti transakcijos kompensavimu, kai visas prieš tai darbas grąžinamas, ir įvykdomas naujas užsakymas. Yra daug išplėstinių transakcijų modelių, kurie naudojami priklausomai nuo aplikacijos specifikos.

1.1.6. BTP protokolas

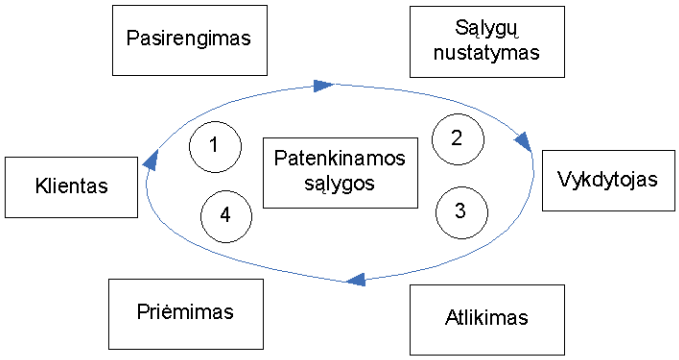
Siekdamas apibrėžti išorinių organizacijų verslo procesų transakcijų protokolą BTP (angl. Business Transaction Protocol), Standartų ir pramonės konsorciumas OASIS suformavo Verslo transakcijų techninį komitetą. BTP [10] – tai XML pagrindu sudarytas protokolas, skirtas vykdyti transakcijas internetu. Kaip ir TIP, jis naudoja dviejų fazių patvirtinimo protokolą darbų sekai kontroliuoti. BTP sukurtas taikomosioms programoms, kurios priklauso dviem ar daugiau verslo partnerių, koordinuoti. Verslo transakcija apibrėžiama kaip nuosekli šių partnerių verslo būsenų kaita. Šiame protokole pristatomos tokios konceptualios sąvokos, kaip nedalomas vienetas (angl. atom) ir tamprumas (angl. cohesion), kurios „sušvelnina“ ACID reikalavimus. Taigi BTP nedaloma (atominė) transakcija išlaiko ACID reikalavimus, tačiau glaudi transakcija leidžia į sekų formavimo uždavinį įsikišti iš išorės, patvirtinant arba atšaukiant nedalomus darbų vienetus. Šis protokolas taip pat apibrėžia ir dalyvius, bei ryšius. Yra du pagrindiniai ryšiai: valdymo ir aukštesniojo bei žemesniojo (angl. inferior - superior). Valdymo ryšys – tai ryšys tarp baigties elemento (angl. Terminator), nustatančio transakcijos pabaigą, ir dalyvio (angl. Decider) (BTP veikėjo, kuris yra transakcijos medžio viršuje, ir kuris sprendžia ar dalyvis patvirtina, ar atšaukia transakciją). Aukštesniojo ir žemesniojo ryšys - tai ryšys tarp transakcijų medžio elementų, kur aukštesnis elementas (koordinatorius) informuoja žemesnį elementą (dalyvį) apie sprendimą dėl transakcijos patvirtinimo ar atšaukimo. BTP yra laikomas vienu perspektyviausių internetinių transakcijų standartų srities projektų. Jis atkreipia dėmesį į dažniausias internetinių transakcijų problemas: ilga trukmė, resursų blokavimas, nepatikimas ryšys ir pan. Tačiau BTP specifikuoja tik pranešimus tarp transakcijų partijų ir todėl prireikia nemažai išteklių tolesniems įdiegimo veiksmams. Kiekvienas transakcijos dalyvis turi specifikuoti savąjį sudėtingą darbų sekos modelį bei komunikacinį protokolą.

Jau yra nemažai komercinių produktų, sukurtų BTP pagrindu. Pvz., *Hewlett Packard* firmos *Web Services Transaction* (*HP-WST*) [11]. Šiam produktui nepavyko užimti didelės rinkos dalies, kadangi jame naudojama daug ir sudėtingų pranešimų struktūrų, nėra pakankamai gerai parengtų priemonių lankstiems būsenos atnaujinimo mechanizmams. Pavyzdžiui, jei atominėje transakcijoje vienas iš elementų yra

neįvykdomas, atšaukiami visi kiti dalyviai. Tai yra labai svarbu, ypač turint omenyje, kad interneto aplikacijose dėl ryšio sutrikimų ar panašių veiksnių gana dažnai pasitaiko neįvykdomų užduočių.

1.1.7. Wynograd ir Flores uždaros kilpos verslo modelis

Dr. Terry Wynograd ir dr. Fernando Flores paminėjo savo knygoje valdymo sistemą, pagristą uždaros kilpos verslo sąveikos modeliu.



1.5 pav. Wynograd ir Flores uždaros kilpos verslo modelis

Tarptautinės produktų kaitos valdymo sistemos (angl. Product Change Management System), grįstos tokiu proceso modeliavimu, kuris orientuotas į sąlygų patikros valdymą bei koordinavimą, taip pat šios sąlygos skirtos proceso paraiškoms, sutartims, įsipareigojimams ir patvirtinimams. Sklandus procesas apibrėžiamas „kilpų“ kalba, kurios yra sistemos modeliavimo ir realizacijos pagrindas. PCMS koordinuoja sąveikas tarp individo ar grupės, darančios užklausą (kliento), ir tos užklausos gavėjo (vykdytojo) keturiomis fazėmis.

- **Pasirengimas**. Klientas pateikia darbą, kuris turi būti atliktas vykdytojo per užklausą.
- **Sąlygų nustatymas.** Klientas ir vykdytojas derina sąlygas, kol pasiekiamas susitarimas dėl darbo, kuris turi būti atliktas.
- **Atlikimas (vykdymas).** Vykdytojas įvykdo užklausą ir praneša apie pabaigą.
- **Priėmimas.** Klientas priima darbą ir įvertina, ar jis tinkamai atliktas, jei ne – išsiaiškina, ko trūksta, ir pateikia naują užklausą.

1.1.8. Elektroninių ryšių valdymas ebXML

EbXML [12] naudojama metodologija verslo procesą apibrėžia kaip verslo bendradarbiavimą, t.y. dviejų ar daugiau partnerių veiklą, siekiant gauti tam tikrą rezultatą. Tai palyginti nauja technologija. EbXML projektas buvo inicijuotas OASIS *(Organization for the Advancement of Structured Information Standards,* t.y. Struktūrizuotų informacinių standartų pažangos prganizacijos) bei UN/CEFACT *(United*

Nations Center for Trade Facilitation and Electronic Business) ir buvo pristatytas 1999 m. 2004 m. kovo mėnesį *International Standartization Organization* (ISO) patvirtino keturių ebXML OASIS standartų rinkinį. ISO/TS 15000 techninės specifikacijos susiddeda iš keturių dalių, kurių kiekviena atitinka vieną ebXML modulio standartą:

- • ISO 15000-1: ebXML partnerio profilio susitarimo specifikacija (ebCPP - *Collaborative Partner Profile Agreement Specification*);
- • ISO 15000-2: ebXML pranešimų apdorojimo specifikacija (ebMS - *Messaging Service Specification*);
- • ISO 15000-3: ebXML registrų informacijos modelis (ebRIM - *Registry Information Model*);
- • ISO 15000-4: ebXML registrų serviso specifikacija (ebRS - Registry Services Specification).

Svarbu suprasti, kad ebXML nėra viena technologija: tai greičiau specifikacijų rinkinys, skirtas kurti el.-verslo struktūrą.

Projekto tikslas buvo suformuluotas ebXML projekto kūrimo pradžioje: „Suteikti atvirą XML grįstą infrastruktūrą, kuri leistų plačiai naudoti elektroninio verslo informaciją veikiančiu, saugiu bei pastoviu būdu visoms šalims“ [13]. Mažos ir vidutinio dydžio įmonės taip pat gali naudoti elektroninio B2B (*Business to Business*) transakcijas per ebXML.

EbXML sudedamosios dalys:

- ebXML partnerio profilio susitarimo specifikacija (ebCPP).

Šios specifikacijos tikslas yra užtikrinti veikimą tarp dviejų informacinių sistemų, net jei jos yra skirtingų kūrėjų.

- ebXML pranešimų apdorojimo specifikacija (ebMS).

Ši specifikacija apibrėžia pranešimų formatą ir antraštinį dokumentą, naudojamus ebXML pranešimams persiųsti per tokius protokolus kaip SMTP ar HTTP, apibrėžia programinės įrangos, kuri siunčia ir priima ebXML pranešimus, veiksmus. Ši programinė įranga gali būti realizuota kaip savarankiškas pranešimų valdiklis arba gali būti el. - verslo funkcionalumo integracijos produktas arba programinis servisas.

- ebXML registrų serviso specifikacija (ebRS).

ebXML registrai stabiliai saugo verslo subjekto įvedamus duomenis. Tokia informacija yra naudojama, kad palengvintų ebXML grįstą B2B bendradarbiavimą bei transakcijas. Įvestas turinys gali būti XML schema ir dokumentai, procesų aprašai, ebXML branduolio sudedamosios dalys, konteksto aprašymai, UML modeliai, informacija apie šalis bei programinės įrangos dalis. Tokia paviešinta informacija yra saugoma saugykloje ir ją valdo ebXML registrų servisais.

- ebXML registrų informacijos modelis (ebRIM).

Registro informacijos modelis yra aukšto lygio ebXML registrų schema. Ji apibrėžia registre saugomų objektų tipus ir saugomų objektų organizavimą registre.

- Verslo procesų specifikavimo schema (ebBPSS).

EbBPSS tikslas yra sujungti el. verslo procesų modeliavimą ir el. verslo programinių dalių specifikavimą. Tiksliau tarant, ebBPSS yra specifikavimo rinkinys, būtinas apibrėžiant dviejų verslo partnerių bendradarbiavimą. Negana to, ji nustato partnerių realaus laiko sistemų parametrus, kad el. verslo programinės dalys galėtų komunikuoti [14].

1.1.9. Elektroninio verslo modelių sudarymo kalba XLANG

XLANG [15] yra vietinės reikšmės kalba. Ji vartojama Microsoft Biz Talk serveryje. Ji skirta formaliam verslo procesų specifikavimui.

XLANG kiekvieno dalyvio serviso sąsajoms specifikuoti vartoja WSDL kalbą (angl. Web Services Description Language). Operacijų duomenų srautai nespecifikuojami. Taikoma ir ilgos kompensuojamos transakcijos samprata. Transakcijas galima rinkti į rinkinius. Kompensacijos taikomos klaidoms apdoroti. XLANG siūlo lanksčius klaidų apdorojimo įrankius ir būsenas atnaujinimo uždaviniams.

Verslo procesus ji apibūdina kaip duomenų apsikeitimą tarp dalyvių (tinklo paslaugų sistemų). Darbų seka apibrėžiama kaip blokinė struktūra, kur procesai gali būti ir dekomponuojami. Ryšiai tarp darbų sekos elementų nusakomi įvairiais nuosekliais lygiagrečiais ar sąlyginiais veiksmais.

Nors XLANG neapibrėžia, kokia veiksmų seka gali būti prižįstama transakcija, tačiau transakcijos, kaip verslo proceso sudedamosios dalies, samprata išlieka. Transakcijų struktūros aprašomos transakcijų blokais, kurie susideda iš bet kokio skaičiaus transakcijų. Su kiekvienu tokiu bloku galima susieti ir kompensuojamąją transakciją. Jei įvyksta klaida, vykdomos kompensuojamiosios transakcijos, kurioms galima nustatyti vykdymo tvarką, tačiau, jeigu nenurodyta kitaip, jos vykdomos atvirkštine tvarka.

1.2. Veiklos procesais grindžiamas IS modeliavimas

Darbe [16] buvo atlikta verslo transakcijų modelių lyginamoji analizė. Buvo prieita prie išvados, kad verslo transakcija skirtingai suprantama įvairiuose modeliuose ir nėra tiksliai apibrėžtas. Yra tokių modelių ir protokolų (pvz., XLANG, BTP), kurie visiškai nepaaiškina, kokia veiksmų seka gali būti laikoma transakcija. Dažniausiai verslo transakciją siūloma laikyti UML (angl. Unified Modelling Language) kalbos vartojimo atveju, t.y. apibrėžtų veiksmų seka, kuri duoda kažkokį reikšmingą rezultatą, tačiau [5] pažymima, kad „toks traktavimas sukelia problemas dėl panaudojimo atvejų detalizavimo laipsnio neapibrėžtumo“.

Projektuojant šiuolaikines informacines sistemas, paprastai pradedama nuo veiklos procesų modeliavimo. Šiame darbe dėmesys sutelktas į elektroninio verslo sistemas. Dažnai jos apima kelias bendradarbiaujančias organizacijas, todėl apibrėžiant darbų seką, modeliavimas yra orientuotas tiek į vidinį, tiek į išorinį organizacijų B2B procesų modeliavimą. Šiame darbe plėtojamas IS modeliavimas grindžiamas paslaugų požiūriu, t.y. elektroninio veiklos procesų modeliavimo rezultatas – vykdomo proceso schema - susieta su tinklo paslaugų (angl. web services) sistema.

Šiame darbe verslo transakcija apibrėžiama naudojant [5] suformuluotus principus. Ji suprantama kaip „nedalomas veiklos vienetas, vaizduojamas pragmatiškai motyvuota išplėstine komunikacine kilpa. Verslo transakcijos yra EM modelio elementai. EM modelį sudarys tarpusavyje susijusių veiklos procesų aibė, kurioje kiekvienas procesas sudarytas iš pragmatiškai motyvuotų komunikacinių kilpų. Įvedus tokį veiklos vienetą, galima užduoti kriterijų panaudojimo atvejams detalizuoti“[5].

1.3. Verslo transakcijos

Šio darbo tikslas - išsiaiškinti verslo transakcijų, modeliavimo principus, bei pagrindines jų sudedamąsias dalis. Tam šiame skyriuje pasirinktas bendras verslo transakcijų pritaikymo atvejis versle ir jis išnagrinėtas. Apibrėšime verslo transakcijos protokolo pagrindinius dalyvius ar sudedamąsias dalis, suformuosime abstrakčią to protokolo realizavimo schemą.

Pati verslo transakciją yra apibrėžiama kaip pastovus verslo ryšio būsenos keitimasis tarp dviejų šalių. Verslo ryšys yra bet kokia paskirstyta būsena, išlaikoma verslo šalių, ir kuri yra kontraktinių taisyklių abipusio sutarimo objektas tarp jų. Pavyzdžiui, pagrindinė pirkimo sutartis, kuri leidžia daryti žinomų kompanijų produktų užsakymus, numato reikiamos informacijos apsikeitimą užsakymo sudarymui ir skaičiavimui tarp pirkėjo ir padavėjo. Tokios sutartys gali įtraukti į specifikacijas pranešimus, kurie perneša bendruosius ar standartinius duomenų formatus ir jų leistiną seką, taip pat viską, ko gali reikėti automatizuotai realizuoti sutartį. Verslo ryšys sąmoningai neskelbiamas, nes verslo šalys reikalauja konfidencialumo, kai atliekamas duomenų apdorojimas.

Kiekvienai verslo pusei, turinčiai verslo ryšio būseną, reikalinga taikomoji sistema, kuri gali tvarkyti tą verslo ryšį ir prireikus bendrauti su kitomis verslo pusėmis. Verslo transakcijų protokolas padeda įvairių pusių taikomosioms sistemoms sukelti pastovius ir koordinuojamus ryšių pasikeitimus.

1.3.1. Išoriniai veiksniai

Verslo transakcijų protokolas koordinuoja būsenos pasikeitimus, kurių atsiranda keičiantis taikomiesiems pranešimams. Tokie pokyčiai yra kontrakto dalis tarp verslo pusių, naudojančių verslo transakcijų protokolą. Apskritai, verslo transakcijų protokolu paremtas servisas turi tam tikru būdu

palaikyti laikinus arba parengtinius būsenos pasikeitimus (laikinas efektas, angl. Privisional Effect) ir užbaigimą arba patvirtinimą (galutinis Efektas, angl. Final Effect) arba nutraukimą (nutraukimo efektas, angl. Counter Effect). Laikino, galutinio ir nutraukimo efektų prasmė priklauso nuo taikomosios programos specifikos ir realizavimo.

Kai kurie bendrieji realizavimo metodai pateikti 1.1 lentelėje.

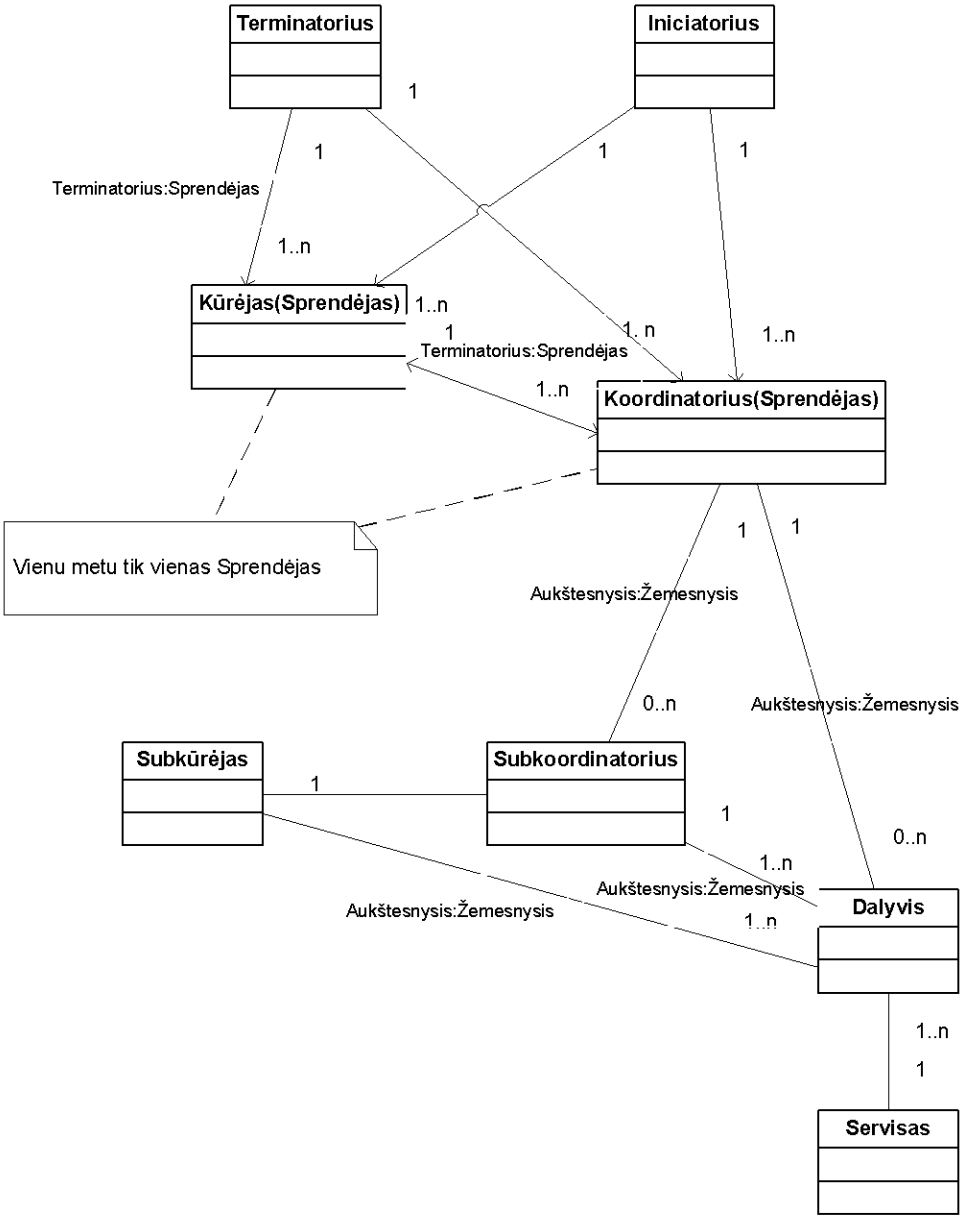
1.1 lentelė, Kai kurios alternatyvos laikinam, galutiniam ir nutraukimo efektams

Laikinas efektas	**Galutinis efektas**	**Nutraukimo efektas**	**Komentaras**
Numatytų pakeitimų išsaugojimas be jų įvykdymo	Atlikti pakeitimus	Ištrinti neįvykdytus išsaugotus pakeitimus	Laikinumo efektas gali apimti patikrą tinkamumui
Atlikti pakeitimus, padaryti juos matomus, anuliuoti pakeitimus	Anuliuojama informacija ištrinama	Įvykdomas grąžininamasis veiksmas	Vienos formos kompensavimo metodas
Išsaugoti originalią būseną, užkirsti priėjimą iš išorės, įvykdyti pakeitimus	Leisti priėjimą	Grąžinti originalią būseną, leisti priėjimą	Tipiškas duomenų bazės metodas
Atlikti pakeitimus, pažymėtus kaip laikinus, padaryti juos matomus	Pažymėti arba transformuoti kaip galutinį	Arba ištrinti, arba pažymėti, arba transformuoti kaip nutrauktą	Pvz.,: užsakymo kiekis

Šios alternatyvos nėra vienintelės, jos gali būti kombinuojamos ar kistis. Matoma būsena Taikomosios programos matoma būsena t.y. ankstesnė informacija patvirtinimui arba nutraukimui, gali skirtis nuo originalios ir galutinės būsenos. Ypač kompensavimo metodo atveju, kait pakeitimai yra nutraukiami, nutraukimo efektas aiškia tvarka panaikina laikinus pakeitimus, arba gali būti atliktas operacijų, kurios kažkokiu būdu atlieka kompensacinį efektą, apdorojimas.

1.3.2. Verslo transakcijų protokolo sudėtis

Žemiau pateiktoje diagramoje pavaizduotos pagrindinės BTP protokolo funkcijos. Jos parodo, kaip apskritai protokolas taikomas.



1.6 pav. Pagrindinės BTP protokolo funkcijos

BTP protokolas yra sukurtas veikti per internetą, kuris yra laisvai susiejama, nepatikima aplinka. Jo tikslas - koordinuoti transakcijos darbą kaip nuoseklų verslo būsenos pasikeitimą.

1.3.2.1. Aukštesnysis ryšys (angl. Superior)

Aukštesnysis ryšys (angl. Superior) yra koordinatoriaus abstrakcija. Jis patvirtina registraciją iš žemesnio (angl. Inferior) ir sukuriamas aukštesnysis ryšys ir žemesnysis ryšys. Vienas aukštesnysis gali turėti ryšių su daugeliu žemesniųjų. Aukštesnysis apsprendžia rezultatą tinkamą žemesniesiems, kurie priregistruoti prie jo. Kai patvirtinimo protokolas inicijuotas, aukštesnysis perduoda parengties signalą visiems registruotiems žemesniesiems, šie bet kuriuo atveju atsako, kaip pasirengta darbui. Sprendimas

padaromas ir tuomet perduodamas to aukštesniojo registruotiems žemesniesiems. Aukštesnysis gali būti arba atominis, arba darnusis (angl. Cohesive).

1.3.2.2. BTP atomai

BTP atomas - tai toks atominis darbo vienetas, kai visas darbas padaromas arba nepadaromas. Atomas gali būti laikomas tradicine atomine transakcija, tvirtai susietose sistemose, tam tikra prasme kai globalus sprendimas yra taikytinas visiems dalyviams, registruotiems prie to atomo. Tai pasiekiama taikant dviejų fazių patvirtinimo protokolą registruotiems žemesniems dalyviams, norint gauti sudėtinį sprendimą. Kiekvienas atomas gali valdyti vieną ar daugiau žemesnių dalyvių, o šie savo ruožtu veikia serviso vardu, kad patvirtintų arba atmestų darbą, pasiūlytą serviso. Pavyzdinis atomo veikimas aprašytas tolesniame skyrelyje.

1.3.2.3. BTP darnumas (angl. Cohesion)

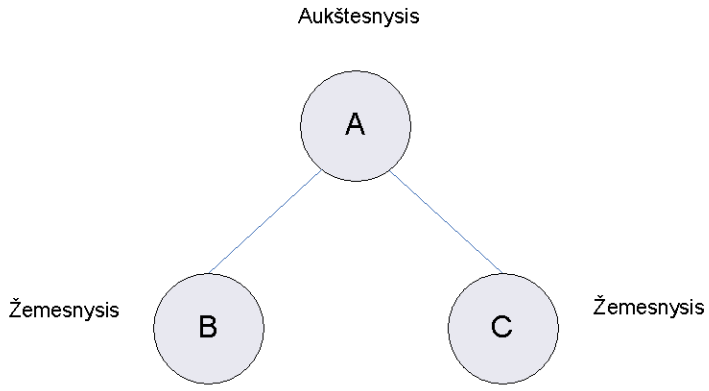
BTP darnumas - tai toks darbo vienetas, kai skirtingi rezultatai gali būti taikomi skirtingiems žemesniems dalyviams, registruotiems tame ryšyje. Rezultatų taikymas žemesniems dalyviams, registruotiems tame ryšyje, priklauso nuo ryšio vartotojo sprendimo, kuris nurodo, kurį žemesnių dalyvių poaibį patvirtinti, o kurį atšaukti. Darnumas leidžia tiksliau koordnuoti, žemesniuosius dalyvius registruotus transakcijoje, taip pat įgyvendina pasirinkimo galimybę, norint pasiekti reikiamą rezultatą. Darnos (angl. Cohesive) koordinatorius atlieka daug sudėtingesnių vaidmenų nei atominis koordinatorius.

1.3.2.4. Žemesnysis ryšys (angl. Inferior)

Žemesnysis ryšys yra atsakingas už rezultato, gauto iš aukštesniojo, taikymą operacijų aibei. Žemesnysis gali būti registruotas su vienu aukštesniuoju. Kai žemesnysis dalyvis gauna „*pasiruošti*“ pranešimą, jis atsako aukštesniam prie, kurio jis yra registruotas, nepriklausomai ar jis paruošė darbą ar ne. Jei žemesnysis užtikrina, kad „*patvirtinti*“ sprendimas gali būti įvykdytas, susietoms operacijoms, ir jis gali išlaikyti informaciją, tuomet išsiunčiamas pranešimas „*pasiruošęs*“. Jei žemesnysis negali užtikrinti, kad darbas bus įvykdytas, tada „*atšaukti*“ pranešimas išsiunčiamas aukštesniajam dalyviui. Aukštesnysis išsiunčia rezultatą žemesniajam. Išsiųstas parnešimas nurodo aukštesnio dalyvio sprendimą, šis sprendimas gali turėti skirtingą semantiką priklausomai nuo žemesniojo registracijos su aukštesniu t.y. jo tipo, atominis ar darnusis.

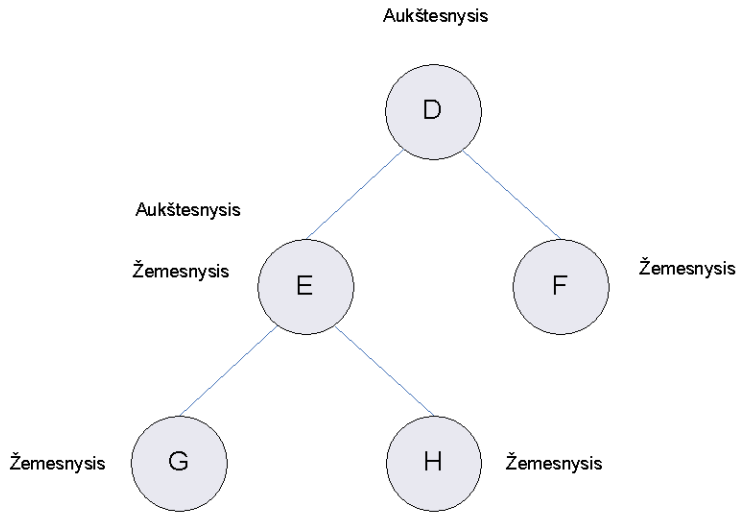
1.3.2.5. Žemesnysis ir aukštesnysis ryšiai

Verslo transakcijos gali būti palyginti komplikuotos ir sudaryti kompleksines aukštesniojo ir žemesniojo ryšių kombinacijas. Pavyzdyje pateiktas paveikslas, kuriame pavaizduoti nepriklausomi žemesni dalyviai (A ir C), kurie registruoti prie aukštesnio dalyvio (A). Žemesni dalyviai (B ir C) vienas nuo kito priklauso tik iš dalies, jie sujungti tiktai per (A), kuris pritaiko rezultatą žemesniems dalyviams, schema išlieka tokia pati esant atominiam koordinatoriui ir pakinta esant darnos koordinatoriui.



1.7 pav. Aukštesnysis ryšis su dviem registruotais žemesniais dalyviais

Žemesnysisir aukštesnis ryšiai gali būti įterpti vienas į kitą, E yra žemesnis už D, bet kartu ir aukštesnis už G ir H. Jei mazgas E yra atominis aukštesnis už G ir H, tuomet jis yra žinomas dar kaip subatominis koordinatorius, jei yra darnos aukštesnysis, tuomet subkūrėjas (angl. sub composer). Šiuo atveju E surenka informaciją iš registruotų žemesnių dalyvių G ir H prieš pranešimą savo aukštesniam dalyviui D.

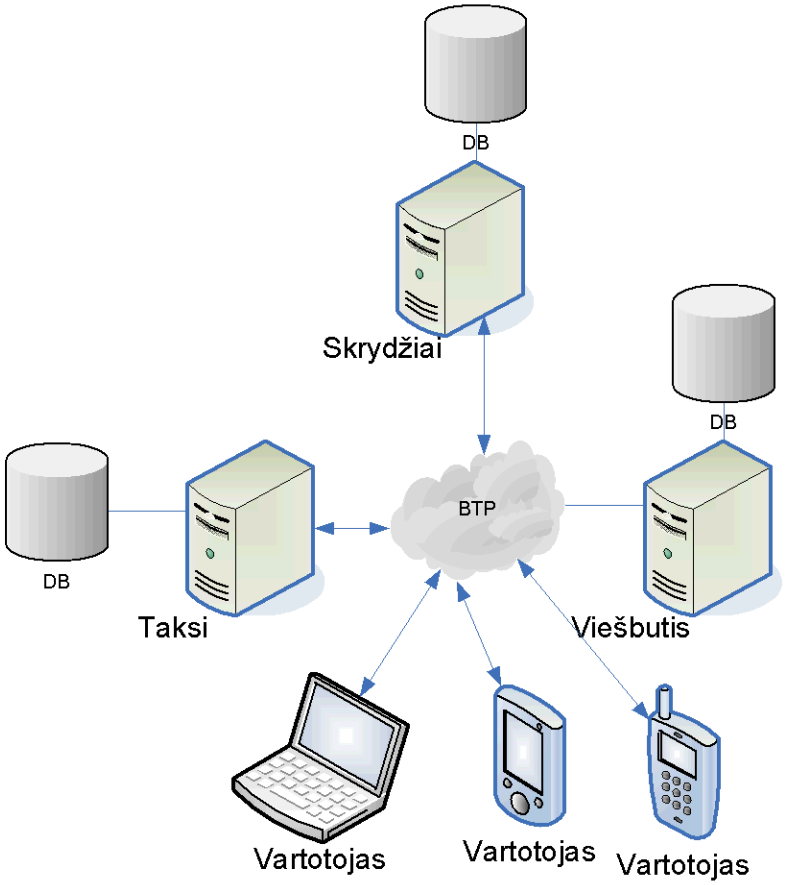


1.8 pav. Galimi įterptiniai aukštesnysis ir žemesnysis ryšiai

Nėra numatyta ribų, kokio gylio gali būti transakcijos medis. Jis gali susidėti iš tam tikro skaičiaus aukštesniojo ir žemesniojo ryšių ir gali baigtis tam tikra labai sudėtinga sąveika. Iniciatorius užklausia konstruktorių (angl. Factory), grąžinantį sprendėją, kuris gali būti naujas aukščiausio lygio BTP sprendėjas, arba subkoordinatorius (angl. sub-cordinator) arba subkūrėjas (angl. sub-composer), jei operacija buvo atlikta šios verslo transakcijos konteksto ribose.

1.3.2.6. Servisas

Servisas apima tam tikrą verslo logikos dalį, reikalingą atlikti tam tikrą užduotį kurios pareikalavo vartotojas. Pavyzdžiui, jei norima užsisakyti taksi, pirmiausia kreipiamasi į taksi servisą. Kai gaunamas taksi serviso objektas, taikomas nuotolinio kvietimo metodas (pvz., *bookTaxi()*)su kai kuriais vartotojo užduotais apribojimais servisas tuomet užklausia savo duomenų bazę pagal reikalavimus, kuriuos uždavė vartotojas. Jei reikalavimai tenkinami, dalyvis registruojamas prie koordinatoriaus sukurtoje transakcijoje. Kai prasideda tvirtinimas, transakcijos koordinatorius bendrauja su registruotais dalyviais.



1.9 pav. BTP sąveika (interakcija)

1.3.2.7. Dalyvis (angl. Participant)

Dalyvis yra žemesnis dalyvis, priklausomas nuo aplikacijos specifikos. Jei sąvoka „dalyvis“ vartojama BTP ryšyje, kalbama apie aplikacijos specifinį žemesnį dalyvį, kuris atsakingas už resursų modifikavimą tam tikroje serviso dalyje. Taigi *Oracle* DB turės skirtingą dalyvį į specifiniame servise, nei kitos DB. Dalyvis realizuos tokį patį interfeisą kaip ir žemesnis dalyvis (angl. Inferior), kuris apibūdina elgseną, kai gaunamos tinkamos BTP žinutės, taip pat jis atspindi jau konkretaus serviso tiekėją, t.y. jame realizuotą specifinę verslo logiką. Dalyvis atsakingas už sprendimą, ar paruošta sąlyga įmanoma, kai dviejų patvirtinimo fazių protokolas inicijuotas, taip pat globalaus sprendimo įvykdymą resurse.

1.3.2.8. Sprendėjas (angl. Decider)

Sprendėjas yra aukštesnis dalyvis (angl. Superior), kuris nėra žemesnis aukštesniojo ir žemesniojo ryšio dalyvis. Jis yra aukščiausias mazgas BTP transakcijos medyje ir gauna užklausas iš terminatoriaus kaip norimą verslo transakcijos rezultatą. Sprendėjas gauna komandą „*parengti*“ arba „*nutraukti*“ (t.y nutraukti kai kurių veiksmus atliekančius ar visų žemesnių dalyvių veiklą ar ir patvirtinti ar nutraukti transakciją) taip pat pateikia terminatoriui verslo transakcijos rezultatą.

Sprendėjai būna dviejų tipų:

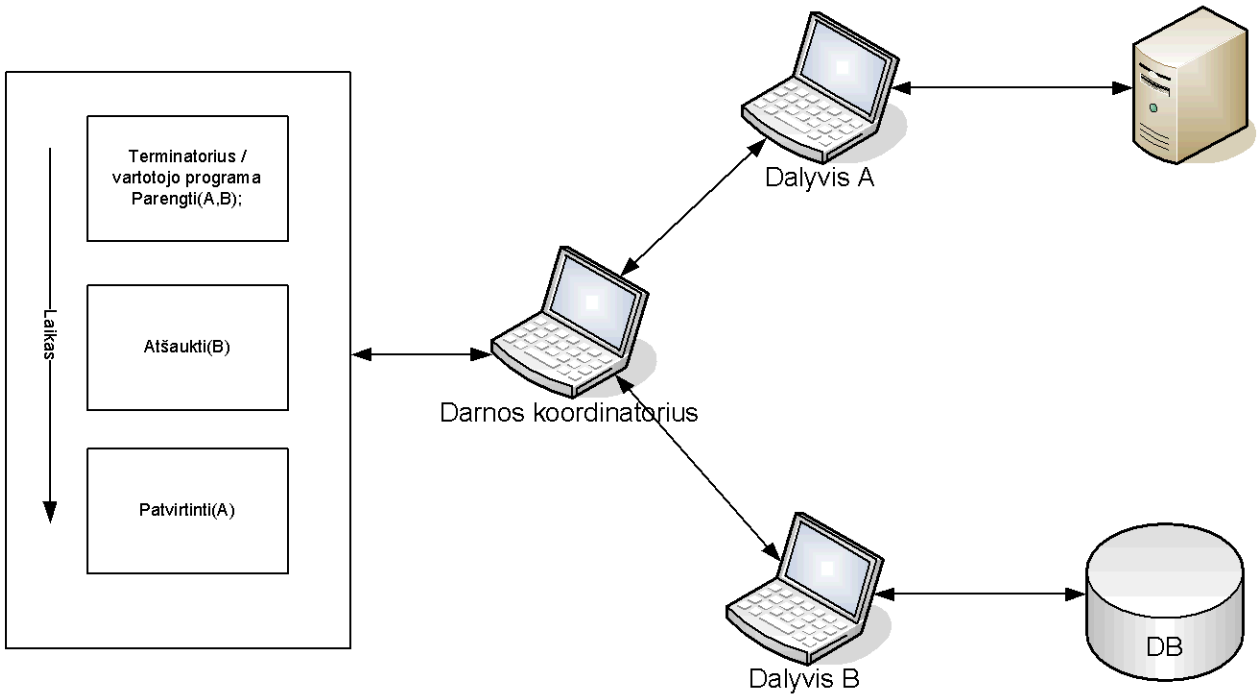
- Koordinatorius, (tai atominis aukštesnis dalyvis, turintis atomo savybių);
- Kūrėjas (angl. Composer), (tai darnos aukštesnis dalyvis (angl. Cohesive Superior), turintis ryšio savybių.

1.3.2.9. Terminatorius (angl. Terminator)

Terminatorius bendrauja su aukščiausiu mazgu BTP transakcijos medyje – sprendėju. Terminatorius dažnai yra aplikacijos elementas, inicijuojantis komunikaciją tarp terminatoriaus ir sprendėjo. Terminatorius bendrauja su atominiu koordinatoriumi kitu atveju - su kūrėju (darnos koordinatoriumi), pvz., terminatorius gali paprašyti kūrėjo parengti kai kuriuos žemesnius dalyvius, kai tuo tarpu tai neleidžiama koordinatoriui, kadangi jis yra atominis vienetas. Terminatorius gali paprašyti sprendėjo nutraukti visą ar tam tikrą verslo transakcijos dalį.

Terminatorius yra aplikacijos elementas, kai sistemos vartotojas gali sukurti ryšį, galintį sujungti kelias verslo sistemas, norint kad jis pasiektų reikiamą rezultatą. Kai vartotojas iškviečia kitų sistemų verslo logiką, registruojami žemesni dalyviai ir priklausomai nuo vartotojo poreikių jis gali parengti arba atšaukti kai kuriuos iš žemesnių dalyvių, todėl patvirtinama arba atšaukiama verslo transakcija.

Verslo transakcijos protokolas leidžia abi patvirtinimo protokolo fazes matyti ir valdyti galutiniam vartotojui. Protokolas leidžia vartotojui nepriklausomai parengti, patvirtinti ar atšaukti dalyvius, registruotus transakcijoje. Toks sušvelnintas kontrolės lygis reikalingas, kadangi BTP transakcijos vyksta ilgai, dalyviai gali būti parengti atskirai dideliu laiko skirtumu, kol galutinis sprendimas (patvirtinimas ar atšaukimas) bus įvykdytas. Žemiau pateiktame paveiksle yra pavaizduota, kaip terminatorius visiškai kontroliuoja transakcijos seką.



1.10 pav. Abi patvirtinimo protokolo fazės

Kaip matome iš paveikslėlio, abi protokolo fazės leidžia lanksčiau valdyti transakciją, nei tą galėtų atlikti įprastinė transakcija.

Normaliai aukštesnis dalyvis siunčia žinutes visiems prie jo registruotiems žemesniems dalyviams, taip inicijuodamas bendravimą tarp jų, tačiau BTP leidžia žemesniems dalyviams priimti sprendimą autonomiškai, nelaukiant žinučių iš aukštesnio lygio dalyvių. Papildoma informacija gali būti įterpta žinutėje, kuri siunčiama koordinatoriui, ši informacija pristatoma kaip kvalifikacinis atributas (angl. Qualifier). Kvalifikacinio atributo pagalba yra įterpiama papildoma informacija į žinutę, kuri siunčiama tarp aukštesniojo ir žemesniojo ryšio. BTP transakcija turi tris pagrindinius kvalifikacinius atributus.

- **Transakcijos laiko riba** (angl. Transaction Time Limit) Toks kvalifikacinis atributas naudojamas BTP taikomosios programos interfeiso vartotojo, kuris pasiūlo maksimalią verslo transakcijos trukmę. Dalyvis gali panaudoti tą trukmę tuo atveju, jei negauna žinučių iš koordinatoriaus (tikėtina, kai jis nebeveiksmingas).

- **Žemesnio dalyvio galiojimas** (angl. Inferior Timeout) Tokį atributą dalyvis grąžina koordinatoriui su parengta žinute. Jis nusako, kiek ilgai dalyvis išliks pasirengęs ir kokių veiksmų imsis, kai laikas pasibaigs.
- **Minimalus žemesnio dalyvio galiojimas** (angl. Minimum Inferior Timeout) Toks atributas paduodamas iš koordinatoriaus dalyviui ir jis nurodo, kiek dalyvis mažiausiai turėtų išlikti pasirengęs, dalyvaudamas transakcijoje. Jei dalyvis negali įgyvendinti šio reikalavimo, jis grąžina rezultatą „Atšaukta“.

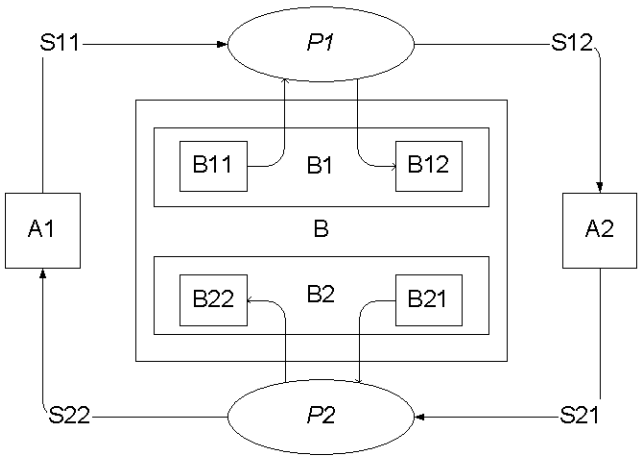
1.3.2.10. Išvados

Atlikus verslo sistemų transakcijų valdymo analizę, buvo nustatytą, kad naudoti tradicinį atominį transakcijų valdymą yra neefektyvu dėl resursų blokavimo, ypač jei transakcijas reikia vykdyti paskirstytose sistemose. Tam buvo sukurtas verslo transakcijų protokolas, kuris neturi minėtų trukumų ir kuriam būdingas didelis lankstumas bei konfigūracijų galimybės. Naudojant šį protokolą paskirstytose sistemose, galima kiekvieną verslo logikos veiksmą, kurio reikalauja vartotojas, pavaizduoti kaip žemesniojo lygio dalyvius, kurie registruojami tos verslo transakcijos koordinatoriuje, kuris buvo sukurtas vartotojo sistemoje. *Microsoft* kompanija yra sukūrusi paskirstytiems resursams valdyti vadinamosius *COM+* servisus, kurie dirba jau minėtu dviejų fazių patvirtinimo protokolo pagrindu, tačiau pats koordinatorius remiasi atominėmis savybėmis, t.y., jei yra nors viena nepatenkinta sąlyga, transakcija atšaukiama. Verslo transakcijų protokolas turi pranašumą, nes gali turėti dvejopus sprendėjus: minėtą koordinatorių ir kūrėją (darnos koordinatorių), kai dalyvis negali atlikti darbo transakcija vis tiek yra patvirtinama. Be to, tyrimo metu buvo nustatyta, kad verslo transakcijas paprasta aprašyti XML formatu, todėl tampa nesudėtinga sujungti kelių sistemų verslo logikos elementus. Šiame darbe daugiausia orientuojamasi į B2B bendravimą, o šiam bendravimui geriausiai visus poreikius atitinka verslo transakcijų protokolas, kuris ir bus realizuojamas projektinėje dalyje.

2. DUOMENŲ SRAUTŲ VALDYMO MODELIS NAUDOJANT KOMUNIKACINES KILPAS

2.1. Verslo transakcijų modeliavimas komunikacinėmis kilpomis

Komunikacinė kilpa [17] – būdas formaliai aprašyti verslo transakcijas tarp dviejų verslo vienetų. Kiekviena komunikacinė kilpa susideda iš dviejų dalyvių – užsakovo A1 ir vykdytojo A2, duomenų srautų S11 – S22 ir dviejų procesų – užsakovo veiksmų P1 ir P2. Duomenų srautų dinamika apibūdinama būsenų perėjimais B.



2.1 pav. Binarinė komunikacinė kilpa

Šis modelis buvo pasiūlytas nagrinėjant naujų funkcinių reikalavimų įvedimui į jau sukurtas ir veikiančias informacines sistemas.

Toks formalizavimo modelis leidžia užtikrinti kai kuriuos semantinio integralumo kriterijus.

Darnos (angl. coherence) – galinė būsena suderinama su dalyvių tikslais.

Pilnumo (angl. completeness) – nėra probleminės būsenos, kuri rodytų būtinybę tobulinti kompiuterizuotą sistemą.

Įgyvendinamumas (angl. viability) – komunikacinė kilpa įgyvendinama, jei

$A1 \xrightarrow[p]{} B11, A2 \xrightarrow[o]{} B22, A2 \xrightarrow[p]{} B21, A1 \xrightarrow[g]{} B12$, tuomet

$B11 \xrightarrow[-]{} B12, B22 \xrightarrow[+]{} B12, B22 \xrightarrow[-]{} B11, B12 \xrightarrow[-]{} B21;$

čia simbolis $\longrightarrow$ žymi pragmatinį priklausomumą: $\xrightarrow[g]{}$ reiškia tikslą, $\xrightarrow[p]{}$ problemą, $\xrightarrow[o]{}$ – progą, $\xrightarrow[+]{}$ – teigiamos įtakos priklausomumą ir $\xrightarrow[-]{}$ – neigiamos įtakos priklausomumą. Kitaip tariant, komunikacinė kilpa yra įgyvendinama, jei gavėjo problema neigiamai veikia iniciatoriaus

problemą ir gavėjas gali įvykdyti reakcijos veiksmą, kurio rezultatas teigiamai veikia tiek gavėjo, tiek iniciatoriaus tikslus[18].

Įvedant naujus funkcinius reikalavimus, kinta ir duomenų srautai bei jų būsena. Jei apsiribojame vieninteliu duomenų formatu – XML, komunikacinėmis kilpomis aprašytų transakcijų duomenų srautus patogu saugoti duomenų bazėje. Duomenų aprašams naudojami standartiniai XML dokumentų struktūros aprašai (DTD ar XSD).

2.2. Duomenų valdymas tarnybinių stočių dalyse

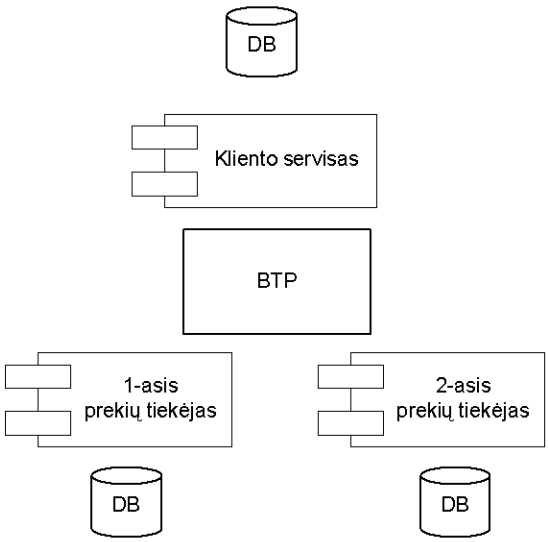
Plėtodami šio mokslinio tyrimo idėjas, stengiamės didžiausią dėmesį kreipti į šių dienų verslo poreikių keliamus uždavinius bei problemas. Ypatingą dėmesį skiriame savo patirties dalykinių sričių analizei. Tai prekyba internetu, pvz., internetinė elektronikos prekių parduotuvė bei išperkamosios nuomos bendrovė. Internetinės parduotuvės keliamas tikslas yra didinti pardavimą.

Šiam tikslui pasiekti keliami įvairūs uždaviniai, pradedant kuo tikslesnių prekių katalogo atnaujinimu, patogiai pateikiama informacija ir baigiant visapusišku klientų poreikių tenkinimu. Šių dienų pirkėjas internetu pageidauja ne tik patogaus pristatymo į namus, tačiau vis dažniau tikisi brangesnius pirkinius įsigyti išsimokėtinai. Pastarajam poreikiui patenkinti elektroninė parduotuvė turi bendradarbiauti su išperkamosios nuomos paslaugas teikiančiomis bendrovėmis bei logistikos paslaugas teikiančiomis įmonėmis todėl ir kyla poreikis kompiuterizuoti tokių verslo partnerių bendradarbiavimą. Klientas pageidauja iškart, dar besirinkdamas prekes, matyti pirkimo išsimokėtinai sąlygas, išsirinkti jam labiausiai priimtiną variantą, galbūt net patvirtinti sutartį nuotoliniu būdu internetu. Deja, šiuo metu tokioms sutartims sudaryti nėra reikiamo teisinio pagrindo, trūksta elektroninio parašo įgyvendinimo galimybių. Nepaisant šių trikdžių, jau dabar galima viską parengti kompiuteriniu variantu ir klientui belieka ateiti pas bendrovės partnerį (juo gali būti internetinės parduotuvės atstovai) ir savo parašu patvirtinti sutartį dėl prekių įsigijimo išsimokėtinai.

Taigi įspraudžiame savo nagrinėjamą uždavinį šitokius rėmus: du verslo partneriai, tarp kurių vyksta duomenų apsikeitimo transakcijos internetinėje erdvėje. Labai patogu rinktis nemokamą, gerai sukurtą ir plačiai naudojamą programavimo aplinką, pažangias technologijas. Internetinėje erdvėje tokios vienos populiariausių yra .NET, bei JAVA technologijos. Verslo transakcijų protokolo realizacijai vartosime C# .NET technologijos programavimo kalbą. Svarbu suderinti duomenų apsikeitimo protokolą bei duomenis perduoti bendrai suprantamu formatu: tokiu atveju puikiai tinka XML duomenų standartas, kadangi jis yra labai plačiai naudojamas internete ir yra labai patogus informaciniams mainams.

2.2.1. Modulinė realizuojamos sistemos struktūra

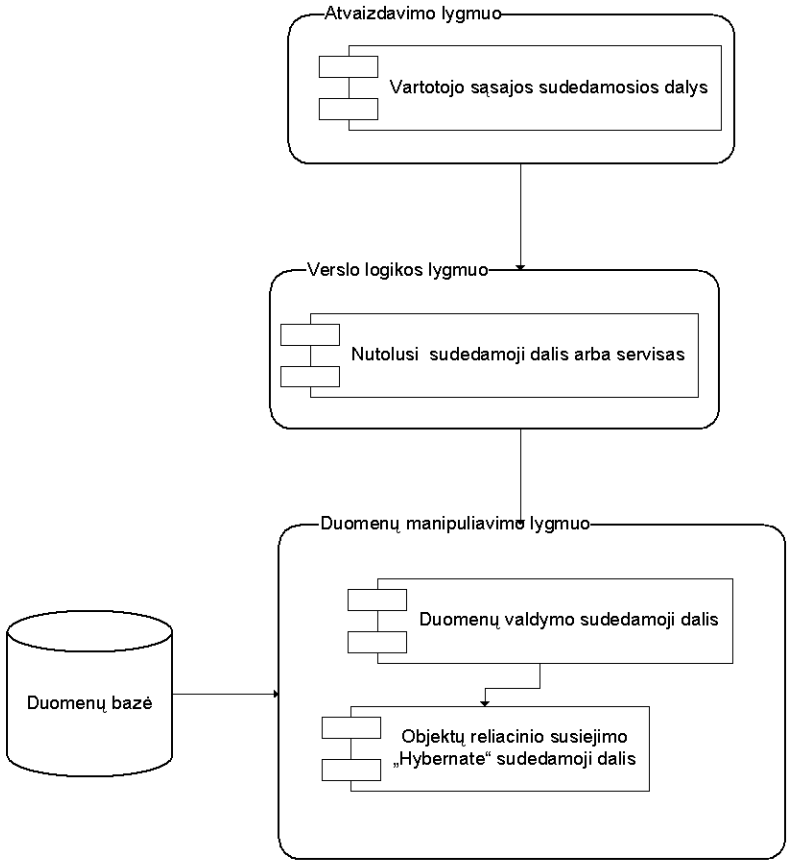
Verslo transakcijų protokolo realizacijai planuojama sukurti eksperimentinę sistemą, kuri apima kelias išorines sistemas, per savo sistemos servisą. Modulinė sistemos struktūra pavaizduota žemiau esančiame paveiksle.



2.2 pav. El. parduotuvės ir išorinių sistemų modulinė schema

2.2.2. Eksperimentinės sistemos lygiai

Pirkėjo pusėje planuojama realizuoti sistemą, kurios lygių diagrama pateikta žemiau esančiame paveiksle.



2.3 pav. El. parduotuvės lygiai

El. parduotuvės eksperimentinis modelis sudarytas iš trijų lygių. Pirmasis tai grafinė vartotojo sąsaja. Šiame lygmenyje atliekamas duomenų atvaizdavimas, jų teisingumo patikrinimas ir pateikimas galutiniam vartotojui. Vidurinysis sluoksnis skirtas verslo logikai, tai nutolęs komponentas, kuris gali egzistuoti kaip nepriklausomas servisas ir kuris gali kreiptis į kitus nurodytus servisus. Taip pat šis lygmuo atsakingas už verslo transakcijų valdymą bei duomenų apdorojimą ir jų pateikimą saugojimui. Žemiausias sluoksnis atsakingas už duomenų valdymą ir visas operacijas, kurios susijusios su duomenų baze. Duomenų prieigos lygis naudoja objektų reliacinį susiejimą, kurį realizuoja „*Hybernate*“ komponentas. Šis komponetas turi visas reikiamas operacijas, kurios reikalingos darbui su duomenų baze. Jis patogus tuo, kad nėra prisirišama prie konkrečios duomenų bazės.

3. EKSPERIMENTINIS ELEKTRONINĖS PARDUOTUVĖS MODELIS

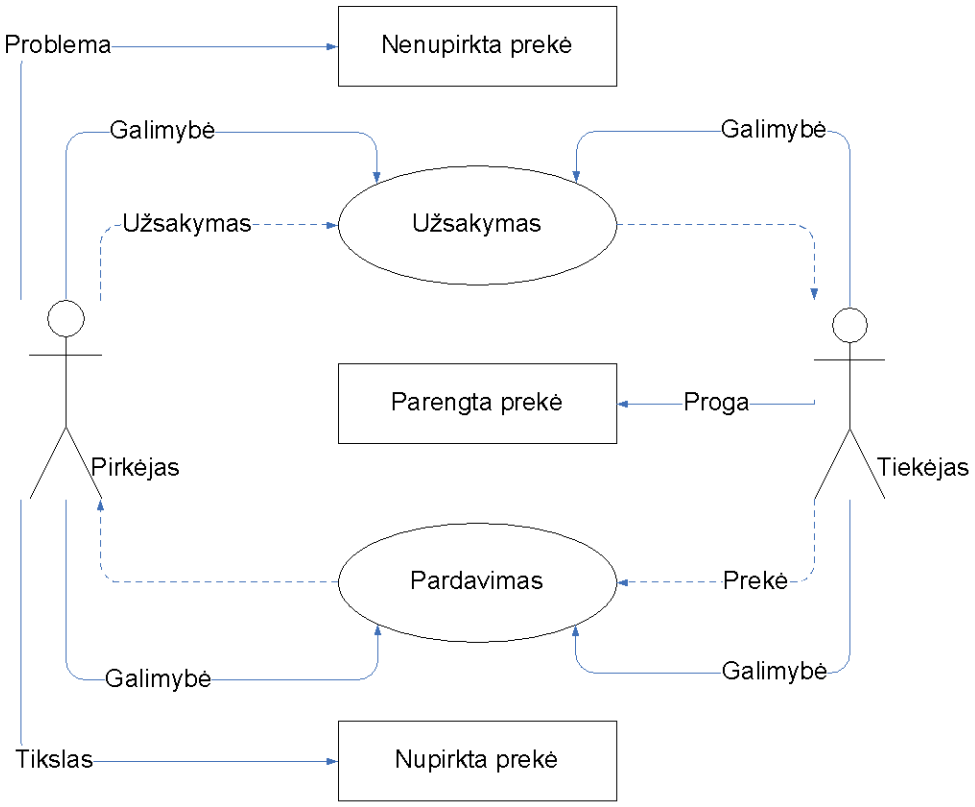
3.1. Dalykinė sritis

Sparčiai besivystant internetinėms technologijoms, didėja elektroninių paslaugų spektras. Darbe nagrinėjamam modeliui realizuoti buvo atliktas eksperimentas. Pasirinkta dalykinė sritis – el. parduotuvė ir joje esančių prekių pirkimas, prekės imamos iš kelių tiekėjų sistemų, kad vartojas galėtų pasirinkti jam labiausiai tinkamą variantą. El. parduotuvė – tai paprastos prekių parduotuvės analogas internetinėje erdvėje. Galima pasirinkti norimas prekes, jų kiekį, tiekėją, palyginti tokių pačių prekių kainas. Už prekes yra atsiskaitoma įvairiais būdais. Įprastiniu tokioms parduotuvėms būdu: kreditinėmis kortelėmis, banko pavedimu ir pan. Eksperimento metu didžiausias dėmesys kreipiamas į išplėstinėmis komunikacinėmis kilpomis specifikuotų verslo transakcijų realizaciją trijų lygių architektūroje, duomenims apibrėžti naudojant XML standartą.

Kaip jau minėta, eksperimentas atliekamas tik techniniame lygyje, nes trūksta juridinių pagrindų tokiam eksperimentui realiai gyvuoti. Kadangi išperkamosios nuomos kompanija, kaip kreditorius, turi patikrinti kliento mokumą, ji gali kreiptis į atitinkamas institucijas: socialinio draudimo duomenų bazę, skolininkų administravimo informacinę sistemą ir pan. Bet tam reikalingas kliento sutikimas tokiems duomenims tvarkyti. Kol kas nėra realiai veikiančių elektroninio parašo nuostatų, todėl tokio sutikimo dokumentas elektroninėje erdvėje kol kas yra neįmanomas.

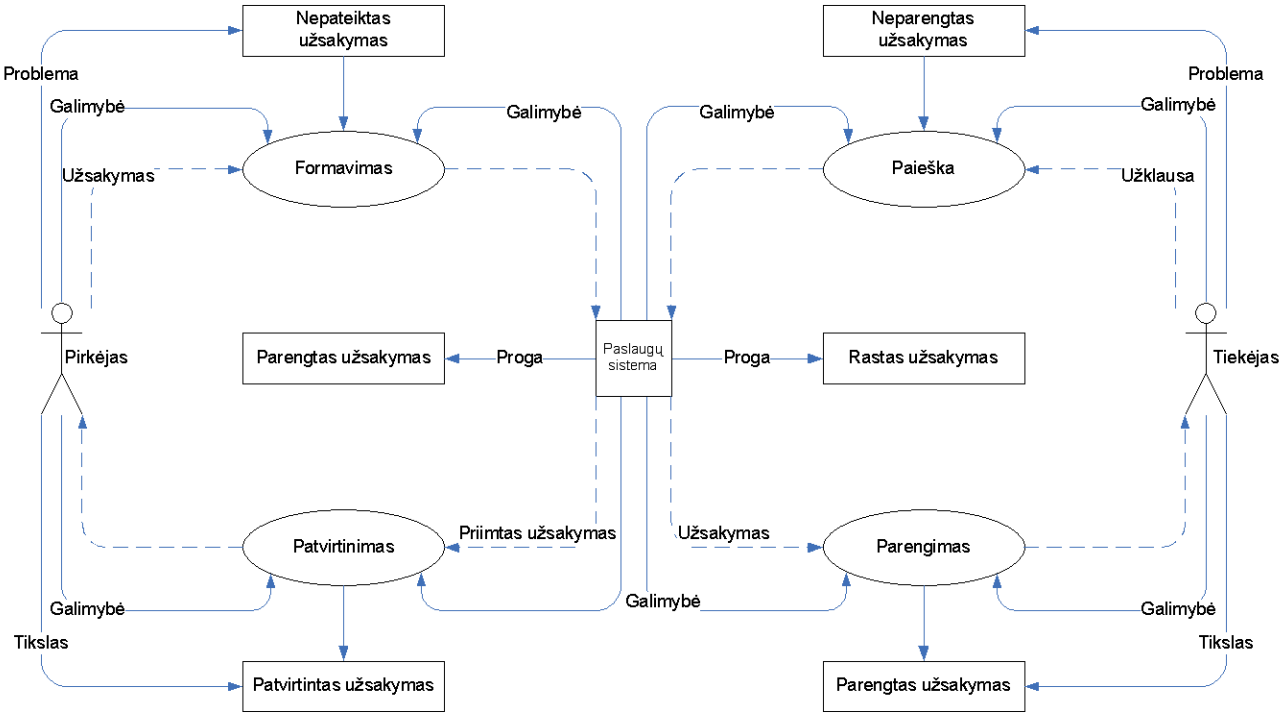
3.2. Kilpų sudarymas

3.2.1. Užsakymo sudarymas el. parduotuvėje



3.1 pav. Pirkėjo ir tiekėjo komunikacinė kilpa

Pirkėjas pateikia užsakymą, kurį sudaro tam tikras pasirnktų prekių kiekis tiekėjui. Jei tiekėjas tokių prekių turi, jas paruošia pirkėjui. Jei pirkėją tenkiną paruoštos prekės, įvykdomas pirkimas.



3.2. pav. Pirkėjo ir tiekėjo išplėstinė komunikacinė kilpa

Prekės kodas	Gamintojas	Pavadinimas	Tipas	Kaina	Kiekis	Garantija	Aprašas
DHD1600DLV1C	AMD	AMD Duron Socket A	DURON	129.64	3	12	AMD CPU Duron 1600MHz (266MHz) 64KB A, tray
	Papildyti	Šalinti	Koreguoti				
DHD1800DLV1C	AMD	AMD Duron Socket A	DURON	145.68	2	12	AMD CPU Duron 1800MHz (266MHz) 64KB A, tray
	Papildyti	Šalinti	Koreguoti				
AX2000DMT3C	AMD	AMD Athlon XP Socket A	PALOMINO	196.47	2	24	AMD Athlon XP 2000+ (1,67GHz/266MHz) Palomino Socket A, tray
	Papildyti	Šalinti	Koreguoti				
AXDA2000DUT3C	AMD	AMD Athlon XP Socket A tray	THOUROUGHBRED	197.47	4	12	AMD Athlon XP 2000+ .13µ (1, 67GHz / 26 266KB Thouroughbred Socket A
	Papildyti	Šalinti	Koreguoti				

Įvesti prekę
Pirkti

3.3 pav. Prekių sąrašo lango grafinis vaizdas

Aukščiau pateiktame paveiksle matomas bendras tiekėjo arba kelių tiekėjų siūlomų prekių vaizdas. Pirkėjas tūri galimybę įsigyti prekes, įdėdamas jas į savo pirkinių krepšelį. Kai pirkėjas įdeda prekes į savo krepšelį sukuriama verslo transakcija kurios aprašas XML formatu pateiktas žemiau. Kai prekių pasirinkimas baigtas, vartotojas patvirtina pirkimą, tuomet verslo transakcija patikrina ar visų tiekėjų, kurių prekes pasirinko pirkėjas, yra pardavime, jei taip transakcija užbaigiama patvirtinimu, kitu atveju pirkėjui pateikiamas sąrašas tik tų prekių kuriuos šiuo metu yra pardavime. Neegzistuojančios prekės išmetamos iš pirkėjo krepšelio (transakcijos dalyviai netenkinantys sąlygos buvo atšaukti), verslo transakcija toliau laukia patvirtinimo arba atšaukimo iš vartotojo.

Parduodamų prekių asortimentas

Prekės kodas	Gamintojas	Kaina	Kiekis sandelyje	Aprašymas	Kiekis	Krepšelis
DHD1600DLV1C	AMD	129.64	3	AMD CPU Duron 1600MHz (266MHz) 64KB/128KB Socket A, tray	1	Pridėti
DHD1800DLV1C	AMD	145.68	2	AMD CPU Duron 1800MHz (266MHz) 64KB/128KB Socket A, tray	1	Pridėti
AX2000DMT3C	AMD	196.47	2	AMD Athlon XP 2000+ (1,67GHz/266MHz) 128KB/256KB Palomino Socket A, tray	1	Pridėti
AXDA2000DUT3C	AMD	197.47	4	AMD Athlon XP 2000+ .13µ (1, 67GHz / 266MHz) 128KB / 256KB Thouroughbred Socket A tray	1	Pridėti

3.4 pav. Parduodamų prekių lango grafinis vaizdas

Pirkėjo krepšelis

Kodas	Aprašymas	Kiekis	Vnt. kaina	Suma	
DHD1600DLV1C	AMD CPU Duron 1600MHz (266MHz) 64KB/128KB Socket A, tray	1	129.64	129.64	Išimti
AXDA2000DUT3C	AMD Athlon XP 2000+ .13µ (1, 67GHz / 266MHz) 128KB / 256KB Thouroughbred Socket A tray	3	197.47	592.41003	Išimti

3.5 pav. Pirkėjo krepšelio lango grafinis vaizdas

Užsakymo metu buvo suformuotas toks verslo transakcijos protokolo XML dokumentas, kai pirkėjas suformuoja krepšelį.

```
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    soap-env:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

  <soap:Header>

    <btp:messages xmlns:btp="urn:oasis:names:tc:BTP:xml">
      <btp:context superior-type="atom">
        <btp:superior-address>
          <btp:binding>soap-http-1</btp:binding>
          <btp:binding-address>http://localhost/client/soaphandler</btp:binding-
address>
          <btp:additional-information>btpengine</btp:additional-information>
        </btp:superior-address>
        <btp:superior-identifier>1001</btp:superior-identifier>
        <btp:qualifiers>
          <btpq:transaction-timelimit
xmlns:btpq="urn:oasis:names:tc:BTP:qualifiers">1800</btpq:transaction-timelimit>
         </btp:qualifiers>
      </btp:context>
    </btp:messages>

  </soap:Header>

  <soap:Body>

    <ns1:orderGoods xmlns:ns1="http://localhost/Services/goods">
      <custID>125</custID>
      <itemID>25</itemID>
      <suplierID>1</suplierID>
      <itemCode>DHD1600DLV1C>
```

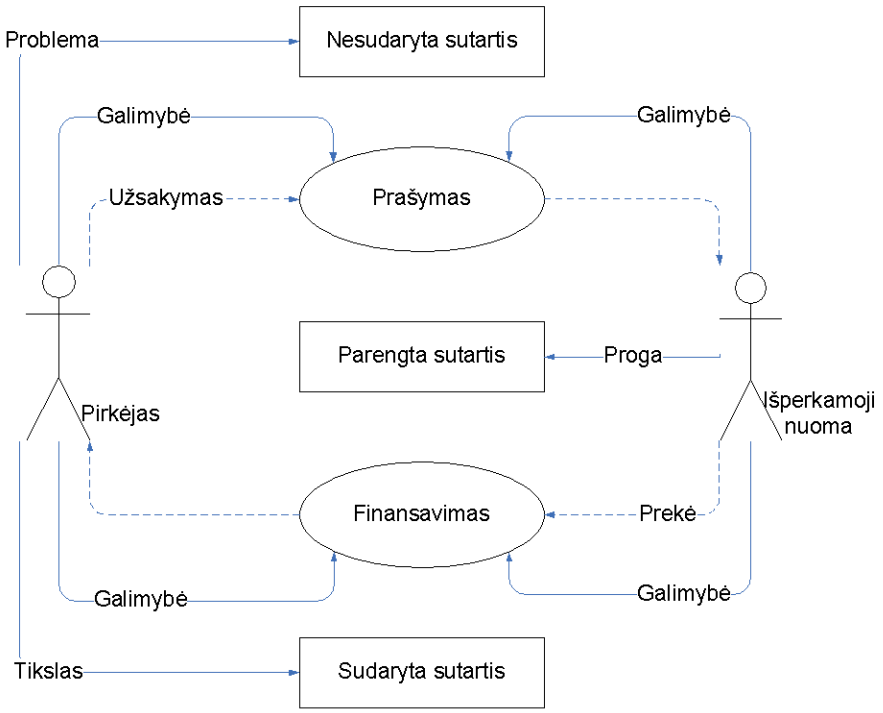
```
      <quantity>1</quantity>
    </ns1:orderGoods>
    <ns1:orderGoods xmlns:ns1="http://localhost/Services/goods">
      <custID>125</custID>
      <itemID>32</itemID>
      <suplierID>2</suplierID>
      <itemCode>AXDA2000DUT3C>
      <quantity>3</quantity>
    </ns1:orderGoods>
  </soap:Body>
</soap:Envelope>
```

Suformuotas XML BTP protokolo dokumentas patvirtinus krepšelį.

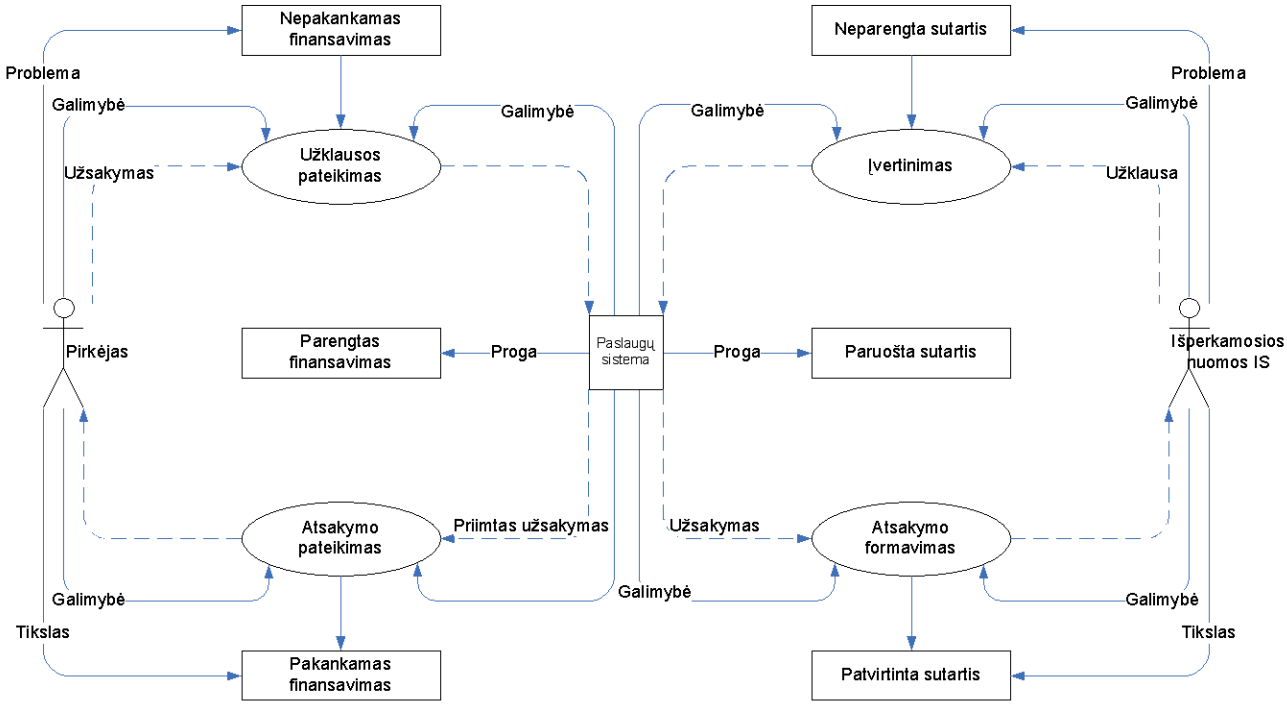
```
<soap:Envelope
    xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
    soap-env:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

  <soap:Header>
  </soap:Header>

  <soap:Body>

    <btp:messages xmlns:btp="urn:oasis:names:tc:BTP:xml">
        <btp:context-reply>
          <btp:superior-address>
           <btp:binding>soap-http-1</btp:binding>
           <btp:binding-address>
               http://localhost/client/soaphandler
           </btp:binding-address>
           <btp:additional-information>
               btpengine
            </btp:additional-information>
        </btp:superior-address>
        <btp:superior-identifier>1001</btp:superior-identifier>
        <completion-status>related</completion-status>

         <btp:enrol reply-requested="false">
          <btp:target-additional-information>
               btpengine
          </btp:target-additional-information>
           <btp:superior-identifier>
               1001
           </btp:superior-identifier>
          <btp:inferior-address>
            <btp:binding>soap-http-1</btp:binding>
            <btp:binding-address>
               http://localhost/services/soaphandler
            </btp:binding-address>
          </btp:inferior-address>
           <btp:inferior-identifier>
               125896
           </btp:inferior-identifier>
          </btp:enrol>

       </btp:context-reply>

    </btp:messages>
```

```
  </soap:Body>
</soap:Envelope>
```

3.2.2. Išperkamosios nuomos sutarties sudarymas



3.6 pav. Pirkėjo ir išperkamosios nuomos komunikacinė kilpa



3.7 pav., Pirkėjo ir išperkamosios nuomos išplėstinė komunikacinė kilpa

3.8 pav. Pirkėjo ir išperkamuosios nuomos eksperimentinis grafinis langas

Užpildžius vartotojo forma duomenimis, buvo suformuotas XML dokumentas apibrėžtu formatu ir perduotas servisui tolimesniam apdorojimui.

```
<?xml version="1.0" encoding="utf-8"?>
<doc xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<uzklausa id="10">
<vardas>Vardenis</vardas>
<pavarde>Pavardenis</pavarde>
<asmens_kodas>38206040000</asmens_kodas>
```

```
<paso_numeris>LR00000</paso_numeris>
<galiojimo_data>2015-01-01</galiojimo_data>
<gatve>Studentų</gatve>
<namas>03</namas>
<butas>124</butas>
<miestas>Kaunas</miestas>
<pasto_kodas>LT-40004</pasto_kodas>
<trukme>18</trukme>
<suma>3000</suma>
<pradinis_inasas>1200</pradinis_inasas>
</uzklausa>
</doc>
```

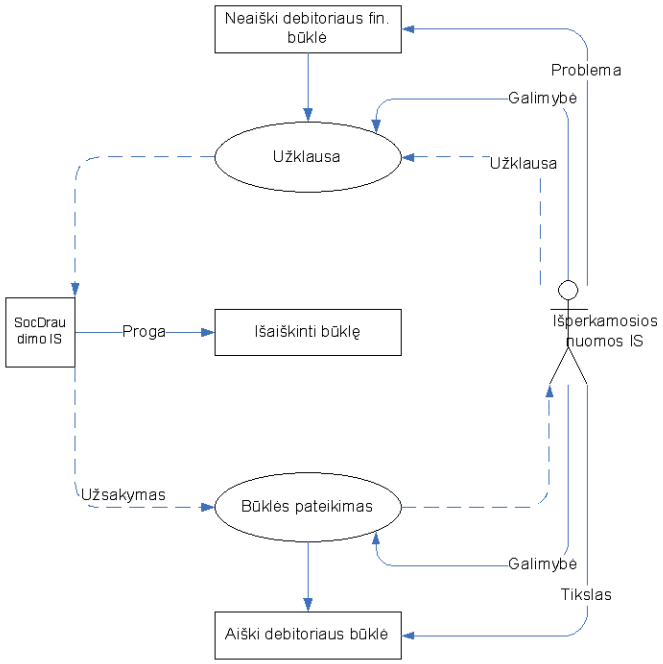
XML identifikavimo parašas:

```
ibgtXZ3HkLfUPC56Lmj4m9f/rpWyvQ5GAU/E3Lwsu4iTpTuC9/Kw5M7bdxcD9fbYqzYaqg2ZW
0Jrp5Z4qe/bFzl7pEFT5xlNLrLsHHEOgTQHoR7AzfyHYuLLcvzJscPptrbhW01duCFnlXCbsrguRDUrJ0lW329oUSkZuv0xmi0=
```

Gautas atsakymas:

```
<?xml version="1.0" encoding="utf-8"?>
<doc xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" >
<uzklausa id="10">
<atsakymas>1</atsakymas>
</uzklausa>
</doc>
```

3.2.3. Debitorių finansinės būklės nustatymas



3.9 pav. Socialinio draudimo ir išperkamosios nuomos kompanijos komunikacinė kilpa

3.3. Saugumo užtikrinimas OpenSSL priemonėmis

3.3.1. SSL raktai

Jei apsikeitimai duomenimis vyksta nesaugioje internetinėje erdvėje, kyla didžiulis pavojus, jog atsiras piktavalių, kurie mėgins įsiskverbti į informacines sistemas. O galimybių tam tikrai yra. Be abejonės, yra sprendimas keliems verslo partneriams susijungti į bendrą tinklą, kuris būtų nepasiekiamas iš viešos erdvės, tačiau neretai tai reikalauja ypatingai didelių investicijų. Tenka ieškoti pigesnės išeities.

OpenSSL ssl bibliotekoje įgyvendinami SSL v2/v3 (angl. Secure Sockets Layer) ir TLS v1 (angl. Transport Layer Security) protokolai. *OpenSSL* komandinės eilutės įrankis naudojamas iškviesti *OpenSSL* bibliotekos metodus. Jie gali būti naudojami [19]:

- RSA, DH ir DSA raktų parametrams kūrti;
- X.509 sertifikatams, CSR ir CRLs kurti;
- Užkoduoti bei atkoduoti su kodavimo algoritmais;
- SSL/TLS kliento bei serverio bendravimui patikrinti;
- S/MIME pasirašytiems ar užkoduotiems elektroniniams laiškams apdoroti.

3.3.2. Raktų generavimas

Apskritai, raktų kūrimas atrodo taip:

1. Klientas sukuria savo viešojo ir slaptojo rakto porą, be to, suformuojama sertifikato užklausa (*certificate request*), kurioje sudėtį yra viešasis raktas (komerciniai *web* serveriai dažnai turi priemonių raktams generuoti).
2. Klientas atsiunčia sertifikato užklausą mums.
3. Mes patvirtiname jų užklausą savo raktu ir suformuojame sertifikatą

Viena pusė, siųsdama pranešimą kitai, patvirtina jį slaptuoju raktu, o kita pusė, turėdama atitinkamą viešąjį raktą (pusę poros), gali patikrinti, ar pranešimas nėra pakeistas. Autorizacijos centrai nereikalingi, mes patys esame sertifikavimo organizacija (angl. certificate authority).

Sertifikatui gauti reikalinga kliento sertifikato užklausa, su šiais duomenimis:
C=LT (šalis)

L=<Vietove> (pvz. Kaunas)

O=<Organizacija> (Organizacija)

OU=BankLink

CN=<serverio vardas> (pvz., banklink-server.organizacija.lt>

E=<kontaktinis el. paštas>

Raktams kurti mes siūlome naudoti *OpenSSL*(http://www.openssl.org), kuris yra gana dažnai atnaujinamas nemokamas kriptografijos įrankių rinkinys.

Kompiliuotą *OpenSSL* versiją galima gauti iš http://www.modssl.org/contrib/, tačiau naudoti ją siūlome tik tada, jei pasitikite minėtu interneto puslapiu.

Rakto ir sertifikato užklausos kūrimas (*OpenSSL*):

openssl req -new -config <Kelias iki openssl.cnf> -out cert_request.pem -keyout private_key.pem

- Paklaus slaptažodžio slaptajam raktui.
- Paprašys duomenų sertifikato užklausai.

Peržiūrėti sertifikato užklausą galite:

openssl req -in cert_request.pem -text -noout

Išsaugoti privatų raktą nešifruota forma galite:

openssl rsa -in private_key.pem -out new_private_key.pem

Pakeisti privataus rakto šifravimo slaptažodi galite:

openssl rsa -in private_key.pem -3des -out new_private_key.pem

Peržiūrėti SSL sertifikatą galite:

openssl x509 -in cert.pem -text -noout

Pavyzdys, kaip suformuoti sertifikato užklausą, pateiktas *http://www.modssl.org/docs/2.8/ssl_faq.html#ToC28*

3.3.3. Raktų ir *Certificate Request* pavyzdžiai

```
-----BEGIN RSA PRIVATE KEY-----
MIICXgIBAAKBgQCvqlkOMrYxGw85j7rkF+KNChajMJL+o6pwGK2bt1IRP/VQxWyB
4AqrljzcUDn1wWTmEzISEJqxF9t/WN8jWcdFCO1Zws5inYt0+WZqkZW+MXVCf+8g
sn8Tbdt7C69++ZH6MUqjJj+Q5EjauchXDmPWPg2v07HBZBIvAhTFDVE1ZwIDAQAB
AoGAMj3OarkmUrUijZwGH+aU7THNd68U26+Dt7vXK4oq9rQMPaW5ewvRAXJexcRM
T3WYnhUvZOtK0wu3w5xvdXTgAOz+Y7ZIIziQCut0+Uy4QERLnKGXZ4830DDh+M7m
oXeF3e+Eur4vx6Whg38jbZs+WyaRJqB2DgNnHhtakyZb/IECQQDkr2KKE6GSJT8h
FXih5Ux2boiN85ehZmN9Maprxviz3V90leyINL6dXapmBDBXClGV2QCDDOMqMEcw
nfqI31HJAkEAxKXBLrvveJ3G5YnOL0IhlGdpBURJtr7h3Pd6E9exdLJ0NBesaour
oj7eJ0+5CSHtCz9ST7S7WgtnguVGaERlrwJBANe8ZRehgXEf1jhdyxnf+YxYplZU
ER3gO8cljYJziLgWBTE1JpkwOQq4DEKFfRz/zQuyVftLsxpfxSkmP+3Tz5ECQQCs
f75rkX5qrvS3i9/rQraUKPZOIW4MOXuFyy0yVMYc2SHoRFABko23oDBeCagGI9V9
RUmeE6s5PMHLZ/YQJtePAkEAn/0DCozbYbz+xlGwhg7kHvH0LqCB6sITxfzMaFQb
T8EGmloj4wIkgvEImgBTyJ3xIZulcDwFpT9qApIKmcCm3Q==
-----END RSA PRIVATE KEY-----
```

Toks pat slaptasis RSA raktas, užšifruotas slaptažodžiu *kala*:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,E6C541BC0138209A

UM49jAy/j7b1JTcuaWJzseIQBk+P7ZOWI6Wp9myGiDk+0+AVj4klNaNOWiH1L3kT
Fma3rOuKGF80ntoPNWcjEcLgK4Zjte026XXcr2w/yM1E8yVlmjVcrLAjR1l9bfe1
rKigDJVEsyUjGDI61rDcfEAbCSO/E5nl4P0M139PeV4paoG03VnvALNQcmI30gHW
BhDK9T2kpwG2rz1ATTf30e7GzpMWvpd7ONMV+39flHAMg7IgRy3GBbERjrwsalxr
dCMdhYyGvrnrBDQDmKuV95jZJu8siTeFDKlFINRS7DsLmhYXCIj50wOUizPLKvxV
dFvAZLiEVXWJ8UoZMn1+QYPNrEty1ox/0rFVM4o1rxK6zSyDoMAoNH6ynoWlLGDx
+nokFgobIWXvKZ2zHhm4+TUbF1cavOXX45yZ837/7udgDnQQ4P02vqpgI0w3oVGG
+5S4gU4ZCjUaax3L3UDAgYRagPY9kJGYcSiJ3399j6gUgcin7ebFAAhfKsi6ioRQ
GadiqhlACMKs30H2oxXJ+4rjNIP6zKlHvkYb3HFrpKOGdArN9KEX7qrOkWDpQaLS
pNlTjR1rKbzsLoqk9cbnRINf+8/Y0BHtnmhISmR0AM4uKYiAfcoCtwwfXMWG5oFT
OMF9oVCGLXFlNpPA60PpLvfYEwZuSEuLr2kJAMOfkSeJyDbInbztwKERRaB2LikC
LxGvpBBphCf48fR8EIQSN9VH7fSLLd3DohkLi5XupUZxxp58TG2WyqBz0isQPfp0
wGvz5ATJ5xtZQ6thrIZlayKV40zUiDSXu9N6ugRG/YJLCncaw8xrWg==
-----END RSA PRIVATE KEY-----
```

Sertifikato užklausa PEM formatu:

```
-----BEGIN CERTIFICATE REQUEST-----
MIIB+TCCAWICAQAwgY8xCzAJBgNVBAYTAkxUMRAwDgYDVQQHEwdWaWxuaXVzMRww
GgYDVQQKExNBQiBiYW5rYXMgSGFuc2EtTFRCMREwDwYDVQQLEwhCYW5rTGluazEa
MBgGA1UEAxMRYmFua2xpbmsuaGFuc2EubHQxITAfBgkqhkiG9w0BCQEWEndlYm1h
c3RlckBoYW5zYS5sdDCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEAr6pZDjK2
MRsPOY+65BfijQoWozCS/qOqcBitm7dSET/1UMVsgeAKq5Y83FA59cFk5hMyEhCa
sRfbf1jfI1nHRQjtWcLOYp2LdPlmapGVvjF1Qn/vILJ/E23bewuvfvmR+jFKoyY/
kORI2rnIVw5j1j4Nr9OxwWQSLwIUxQ1RNWcCAwEAAaApMCcGCSqGSIb3DQEJDjEa
MBgwCQYDVR0TBAIwADALBgNVHQ8EBAMCBPAwDQYJKoZIhvcNAQEEBQADgYEALdf7
7J/GLfGZE27vXRLJrqAvjZeeV3/QlJMQTjhFihVb8N9h6Z+s7e+VbgEpKq5feVot
gbN8oPPbSYC0TVPOdRpKIi7+IzfVWWFgSZxUGKqdkypaGcRM4qa+wgPFsUi6syhZ
c5gJDvcjp+lEUvOett2/KT7KLFiQs9o/glY8PEI=
-----END CERTIFICATE REQUEST-----
```

Tokia pati sertifikato užklausa tekstu:

```
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=LT, L=Vilnius, O=AB bankas Hansa-LTB, OU=BankLink,
CN=banklink.hansa.lt/Email=webmaster@hansa.lt
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:af:aa:59:0e:32:b6:31:1b:0f:39:8f:ba:e4:17:
                    e2:8d:0a:16:a3:30:92:fe:a3:aa:70:18:ad:9b:b7:
                    52:11:3f:f5:50:c5:6c:81:e0:0a:ab:96:3c:dc:50:
                    39:f5:c1:64:e6:13:32:12:10:9a:b1:17:db:7f:58:
                    df:23:59:c7:45:08:ed:59:c2:ce:62:9d:8b:74:f9:
                    66:6a:91:95:be:31:75:42:7f:ef:20:b2:7f:13:6d:
                    db:7b:0b:af:7e:f9:91:fa:31:4a:a3:26:3f:90:e4:
                    48:da:b9:c8:57:0e:63:d6:3e:0d:af:d3:b1:c1:64:
                    12:2f:02:14:c5:0d:51:35:67
                Exponent: 65537 (0x10001)
        Attributes:
```

```
            Requested Extensions:
                X509v3 Basic Constraints:
                    CA:FALSE
                X509v3 Key Usage:
                    Digital Signature, Non Repudiation, Key Encipherment, Data
Encipherment
        Signature Algorithm: md5WithRSAEncryption
            2d:d7:fb:ec:9f:c6:2d:f1:99:13:6e:ef:5d:12:c9:ae:a0:2f:
            8d:97:9e:57:7f:d0:94:93:10:4e:38:45:8a:15:5b:f0:df:61:
            e9:9f:ac:ed:ef:95:6e:01:29:2a:ae:5f:79:5a:2d:81:b3:7c:
            a0:f3:db:49:80:b4:4d:53:ce:75:1a:4a:22:2e:fe:23:37:d5:
            59:61:60:49:9c:54:18:aa:9d:93:2a:5a:19:c4:4c:e2:a6:be:
            c2:03:c5:b1:48:ba:b3:28:59:73:98:09:0e:f7:23:a7:e9:44:
            52:f3:9e:b6:dd:bf:29:3e:ca:2c:58:90:b3:da:3f:82:56:3c:
            3c:42
```

IŠVADOS

Norint garantuoti patikimą verslo informacinių sistemų funkcionavimą, jas projektuojant būtina formaliai aprašyti verslo transakcijas. Tokiam aprašymui siūloma nemažai metodų, komercinių produktų, standartų ir pan., kurie buvo apžvelgti. Šiame darbe buvo gilinamasi į išplėstines komunikacines kilpas verslo transakcijoms modeliuoti. Taikant šį metodą, verslo transakcija suprantama kaip nedalomas veiklos vienetas, vaizduojamas praktiškai motyvuotai komunikacinėse kilpose.

Buvo pastebėta, kad transakciją galima susieti su *UML* kalbos panaudojimo atvejo modeliu. Tada ji suprantama kaip dalyvio veiksmų seka, duodanti tikslingą rezultatą. Tačiau toks traktavimas sukelia problemų dėl panaudojimo atvejo detalizavimo. Nėra būdų formaliai apibrėžti, koks būtent veiksmas jau gali būti laikomas nedalomu. Tai visuomet priklauso nuo konkrečios situacijos.

Išplėstinių komunikacinių kilpų modelis verslo transakcijoms specifikuoti nepriklauso nuo konkrečios platforminės IS realizacijos. Tai standartinis vienetas, kuris yra labai aiškiai suprantamas ir atitinka pačią IS projektavimo koncepciją – formalų procesų aprašymą. Šiame darbe pateiktas siūlymas pritaikyti šį modelį konkrečiai realizacijai, komunikacinėmis kilpomis aprašytus dalyvio procesus realizuoti trijų lygių architektūra. Toks platforminis sprendimas ypač tinka elektroniniams verslas – verslui (B2B) procesams realizuoti.

Siekiant užtikrinti patikimą duomenų apsikeitimą tarp dviejų ar kelių verslo partnerių elektroninėje erdvėje, būtinas konkretus ir aiškus duomenų apibrėžimas. Kadangi IS projektuoti siūlomas trijų lygių platforminis sprendimas, kurio konkrečių realizacijos būdų yra nemažai, todėl reikia parinkti tinkamą duomenų formatą. Tam labai tinka šiuo metu dažnai naudojamas XML standartas. XML duomenų valdymas (gavimas, perdavimas, transformavimas, kūrimas ir pan.) trijų lygių architektūroje gali būti realizuotas duomenų bazės arba tarnybinės stoties dalių lygyje. Duomenų bazės lygyje XML srautų valdymas kol kas nėra labai išplėtotas. Tik didžiosios DBVS(*Oracle, MS SQL Server*) siūlo įrankių XML srautams apdoroti. Kadangi tai palyginti brangūs komerciniai produktai, XML apdorojimas dažnai vykdomas tarnybinės stoties dalyse.

Darbe buvo realizuotas verslo transakcijų protokolas ir pritaikytas transakcijoms valdyti paprasčiausioje el. parduotuvėje.

Šiame projekte jungimuisi ir valdymui su duomenų bazėmis nepriklausomai nuo platformos buvo naudota „*Hybernate*“ pagalbinė sistema, kuri leidžia prisijungti prie įvairių duomenų bazių. Jos funkcijos yra kurti SQL užklausas patogiu ir priimtinu programuotojams metodu, pateikti jas duomenų bazėms,

gautą atsakymą gauti C# klasių objektų ar jų kolekcijų pavidalu. Programuotojai gaudami atsakymus tokiu pavidalu gali nesunkiai ir tvarkingai atlikti tolesnius dalykinės srities reikalavimus atitinkančią realizaciją.

XML apdoroti buvo naudotos tik įprastinės priemonės ir bibliotekos.

Šiame darbe nagrinėjamam modeliui realizuoti buvo atliktas eksperimentinis tyrimas. Eksperimento dalykinė sritis: el. parduotuvė ir pirkimas joje išsimokėtinai sudarant sutartį su išperkamuosios nuomos kompanija. Eksperimento metu nustačius verslo procesus, buvo sudarytos juos aprašančios išplėstinės komunikacinės kilpos, kurių procesai buvo realizuoti trijų lygių architektūra. Duomenų apibrėžimui panaudotas XML formatas, kuriuo ir buvo realizuotas verslo transakcijų protokolas. Praktiškai patikrintas „*Hybernate*“ posistemės veikimas, patikrintas XML formato universalumas bei trijų lygių architektūros naudingumas realizuojant B2B procesus.

LITERATŪRA

[1] Dogac A., Yuruten B., Sapacapietra S. A Generalized Expert System for Database Design. IEEE Transactions on Software Engineering, 1989, 18(4), pp. 479-491.

[2] Moss E. Nested Transactions. MIT Press, 1985 pp. 189.

[3] Weikum G., Schek H. Concepts and Applications of Multi-level Transactions and Open Nested Transactions. Morgan Kaufmann, 1992. pp. 146-183.

[4] Grefen P., Vonk J., Apers P. Global transaction support for workflow management systems: from formal specification to practical implementation. The VLDB Journal N 10, 2001, pp. 316-333.

[5] Nemuraitė L., Paradauskas B. Duomenų bazės ir semantiniai modeliai. Monografija. Kaunas: Technologija (2002) p. 227-243.

[6] Klein J. Transaction Internet Protocol Version 3.0. Proceedings Of The Forty-First Internet Engineering TaskForce. 1998. Los Angeles, USA

[7] Alonso G., Kamath M., Agrawal D., Abbadi A.El, Gunthor R., Mohan C. Failure handling in large scale workflow management systems. Technical Report RJ9913, IBM Almaden Research Center, 1994.

[8] Alonso G, Agrawal D, Abbadi A. El, Kamath M, Gunthor R., Mohan C. Advanced transaction model in workflow context. Proceedings of the 12th International Conerence. on Data Engineering, New Orleans, USA 1996

[9] Krishnamoorty V., Shan M.C. Virtual Transaction Model for Workflow Applications. Procs of SAC'00. Como, Italy. 2000.

[10] Business Transaction protocol. Draft Specification, version 1.0. –Organization for the Advancement of Structured information Systems (OASIS), 2002

[11] HP, HP Web Service Transaction 1.0 Tech Preview. 2002 HP. Prieiga per internetą: www.hpmiddleware.com/HPISAPI.dll/hpmiddleware/products/webservices_transactions/default.jsp. Žiūrėta 2006-12-02

[12] Walsh A. E. EbXML: the Technical reports. –Prentice Hall PTR, 2002

[13] EbXML.org, 'ebXML – Enabling a global electronic market', Organization for the Advancement of Structured Information Standards (OASIS), 2004. www.ebxml.org (žiūrėta 2006-12-15)

[14] Hakvoort R. EbXML and its impact on conventional Business Information Systems, 1st Twente Student Conference on IT, Enschede 14 June 2004,Faculty of EEMCS – University of Twente, Enschede, The Netherlands

[15] Thatte S. XLANG. Web Services for Business Process Design. Microsoft Corporation, 2001

[16] Nemuraitė L. Elektroninio verslo procesų modeliavimo metodų tendencijos. Informacijos mokslai, Vilniaus universitetas, 2002.

[17] Nemuraitė L., Paradauskas B., Salelionis L. Extended Communicative Action Loop for Integration of New Functional Requirements // Information Technology and Control. Kaunas: Technologija, 2002, no. 2(23), p. 18–26.

[18] Ričardas Kanaitis, Bronius Paradauskas. Naujų funkcinių reikalavimų įvedimas informacinėse sistemose, Informacinės technologijos: X tarpuniversitetinė magistrantų ir doktorantų konferencija, Kaunas, 2005, p.155-158

[19] OpenSSL informacijos saugykla. Prieiga per internetą: http://www.openssl.org/docs/HOWTO/. Žiūrėta 2006-12-10

TERMINŲ IR SANTRUMPŲ ŽODYNAS

XML dokumentas – XML programavimo kalba pateiktas dokumentas.

XML schema – XML kalba parašytas dokumentas, skirtas XML dokumentui apibrėžti.

Duomenų bazė – kartu saugomų ir susijusių duomenų visuma, kuriai būdinga integruotumas, nepertekliškumas, nepriklausomumas.

Duomenų bazių valdymo sistema – programų rinkinys, atliekantis duomenų apdorojimo operacijas.

DB- duomenų bazė.

BTP – verslo transakcijų protokolas.

2PF – dviejų patvirtinimo fazių protokolas.

1 PRIEDAS. Verslo transakcijų protokolo XML schema

Pateikiama XML schema pagal kuria eksperimente buvo tikrinamas verslo transakcijos XML dokumentas.

```
<?xml version="1.0"?>
<schema targetNamespace="http://docs.oasis-open.org/business-transaction/business_transaction-btp-1.1-core-schema-cd-
01.xsd"
        elementFormDefault="qualified" xmlns="http://www.w3.org/2001/XMLSchema"
        xmlns:btp="http://docs.oasis-open.org/business-transaction/business_transaction-btp-1.1-core-schema-cd-01.xsd">
    <!-- Qualifiers -->
    <complexType name="qualifier-type" mixed="true">
        <complexContent mixed="true">
            <restriction base="anyType">
                <sequence>
                    <any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
                </sequence>
                <attribute name="must-be-understood" type="boolean" use="optional" default="true"/>
                <attribute name="to-be-propagated" type="boolean" use="optional" default="false"/>
            </restriction>
        </complexContent>
    </complexType>
    <element name="qualifier" type="btp:qualifier-type" abstract="true"/>
    <element name="qualifiers">
        <complexType>
            <choice>
                <element ref="btp:qualifier" minOccurs="0" maxOccurs="unbounded"/>
                <any processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
            </choice>
        </complexType>
    </element>
    <!-- example qualifier definition:
        <element name="some-qualifer" type="btp:qualifier-type" substitutionGroup="btp:qualifier"/>
    -->
    <!-- Message set data types -->
    <simpleType name="identifier">
        <restriction base="anyURI"/>
    </simpleType>
    <simpleType name="additional-information">
        <restriction base="string"/>
    </simpleType>
    <complexType name="address">
        <sequence>
            <element name="binding-name" type="string"/>
            <element name="binding-address" type="string"/>
            <element name="additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="priority" type="positiveInteger" use="optional" default="1"/>
    </complexType>
    <simpleType name="superior-type">
        <restriction base="string">
            <enumeration value="cohesion"/>
            <enumeration value="atom"/>
        </restriction>
    </simpleType>
    <simpleType name="transaction-type">
        <restriction base="string">
            <enumeration value="cohesion"/>
            <enumeration value="atom"/>
        </restriction>
    </simpleType>
    <!-- Compounding -->
    <element name="messages">
        <complexType>
            <sequence>
                <element ref="btp:message" minOccurs="0" maxOccurs="unbounded"/>
            </sequence>
        </complexType>
    </element>
    <element name="related-group" substitutionGroup="btp:message">
        <complexType>
            <sequence>
                <element ref="btp:message" minOccurs="0" maxOccurs="unbounded"/>
            </sequence>
        </complexType>
    </element>
    <!-- Message set -->
    <element name="message" abstract="true"/>
    <element name="context" substitutionGroup="btp:message">
        <complexType>
```

```
        <sequence>
            <element name="superior-address" type="btp:address" maxOccurs="unbounded"/>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="superior-type" type="btp:superior-type" default="atom" minOccurs="0"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="context-reply" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier" minOccurs="0"/>
            <element name="completion-status">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="completed"/>
                        <enumeration value="incomplete"/>
                        <enumeration value="related"/>
                        <enumeration value="repudiated"/>
                    </restriction>
                </simpleType>
            </element>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="request-status" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="target-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="status" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="responders-identifier" type="btp:identifier"/>
            <element name="status-value">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="created"/>
                        <enumeration value="enrolling"/>
                        <enumeration value="active"/>
                        <enumeration value="resigning"/>
                        <enumeration value="resigned"/>
                        <enumeration value="preparing"/>
                        <enumeration value="prepared"/>
                        <enumeration value="confirming"/>
                        <enumeration value="confirmed"/>
                        <enumeration value="cancelling"/>
                        <enumeration value="cancelled"/>
                        <enumeration value="cancel-contradiction"/>
                        <enumeration value="confirm-contradiction"/>
                        <enumeration value="hazard"/>
                        <enumeration value="contradicted"/>
                        <enumeration value="unknown"/>
                        <enumeration value="inaccessible"/>
                    </restriction>
                </simpleType>
            </element>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="fault" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier" minOccurs="0"/>
            <element name="inferior-identifier" type="btp:identifier" minOccurs="0"/>
            <element name="fault-type">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="communication-failure"/>
                        <enumeration value="duplicate-inferior"/>
```

```
                        <enumeration value="general"/>
                        <enumeration value="invalid-decider"/>
                        <enumeration value="invalid-inferior"/>
                        <enumeration value="invalid-superior"/>
                        <enumeration value="status-refused"/>
                        <enumeration value="invalid-terminator"/>
                        <enumeration value="unknown-parameter"/>
                        <enumeration value="unknown-transaction"/>
                        <enumeration value="unsupported-qualifier"/>
                        <enumeration value="wrong-state"/>
                        <enumeration value="redirect"/>
                    </restriction>
                </simpleType>
            </element>
            <element name="fault-text" type="string" minOccurs="0"/>
            <element name="fault-data" type="anyType" minOccurs="0"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="enrol" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="response-requested" type="boolean" default="false" minOccurs="0"/>
            <element name="inferior-address" type="btp:address" maxOccurs="unbounded"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="enrolled" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="resign" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element name="response-requested" type="boolean" default="false" minOccurs="0"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="resigned" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="prepare" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
```

```
<element name="prepared" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element name="default-is-cancel" type="boolean"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="confirm" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="confirmed" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element name="confirm-received" type="boolean"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="cancel" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="cancelled" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier" minOccurs="0"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="confirm-one-phase" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="hazard" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element name="level">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="mixed"/>
                        <enumeration value="possible"/>
                    </restriction>
```

```
                </simpleType>
            </element>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="contradiction" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="superior-state" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element name="status">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="active"/>
                        <enumeration value="prepared-received"/>
                        <enumeration value="confirmed-received"/>
                        <enumeration value="cancelled-received"/>
                        <enumeration value="contradiction-known"/>
                        <enumeration value="inaccessible"/>
                        <enumeration value="unknown"/>
                    </restriction>
                </simpleType>
            </element>
            <element name="response-requested" type="boolean" default="false" minOccurs="0"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="inferior-state" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element name="status">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="active"/>
                        <enumeration value="prepare-received"/>
                        <enumeration value="confirm-received"/>
                        <enumeration value="cancel-received"/>
                        <enumeration value="inaccessible"/>
                        <enumeration value="unknown"/>
                    </restriction>
                </simpleType>
            </element>
            <element name="response-requested" type="boolean" default="false" minOccurs="0"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="sender-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="redirect" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="superior-identifier" type="btp:identifier" minOccurs="0"/>
            <element name="inferior-identifier" type="btp:identifier"/>
            <element name="old-address" type="btp:address" maxOccurs="unbounded"/>
            <element name="new-address" type="btp:address" maxOccurs="unbounded"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
```

```
<element name="begin" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="transaction-type" type="btp:superior-type"/>
            <element ref="btp:context" minOccurs="0"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="begun" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="decider-address" type="btp:address" minOccurs="0" maxOccurs="unbounded"/>
            <element name="inferior-address" type="btp:address" minOccurs="0" maxOccurs="unbounded"/>
            <element name="transaction-identifier" type="btp:identifier"/>
            <element ref="btp:context"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="prepare-inferiors" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="transaction-identifier" type="btp:identifier"/>
            <element name="inferiors-list" minOccurs="0">
                <complexType>
                    <sequence>
                        <element name="inferior-identifier" type="btp:identifier" maxOccurs="unbounded"/>
                    </sequence>
                </complexType>
            </element>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="confirm-transaction" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="transaction-identifier" type="btp:identifier"/>
            <element name="inferiors-list" minOccurs="0">
                <complexType>
                    <sequence>
                        <element name="inferior-identifier" type="btp:identifier" maxOccurs="unbounded"/>
                    </sequence>
                </complexType>
            </element>
            <element name="report-hazard" type="boolean"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="transaction-confirmed" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="transaction-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="cancel-transaction" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="transaction-identifier" type="btp:identifier"/>
            <element name="report-hazard" type="boolean"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="cancel-inferiors" substitutionGroup="btp:message">
```

```
    <complexType>
        <sequence>
            <element name="transaction-identifier" type="btp:identifier"/>
            <element name="inferiors-list">
                <complexType>
                    <sequence>
                        <element name="inferior-identifier" type="btp:identifier" maxOccurs="unbounded"/>
                    </sequence>
                </complexType>
            </element>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="transaction-cancelled" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="transaction-identifier" type="btp:identifier"/>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="request-inferior-statuses" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="target-identifier" type="btp:identifier"/>
            <element name="inferiors-list" minOccurs="0">
                <complexType>
                    <sequence>
                        <element name="inferior-identifier" type="btp:identifier" maxOccurs="unbounded"/>
                    </sequence>
                </complexType>
            </element>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
            <element name="reply-address" type="btp:address" minOccurs="0"/>
        </sequence>
        <attribute name="id" type="ID" use="optional"/>
    </complexType>
</element>
<element name="inferior-statuses" substitutionGroup="btp:message">
    <complexType>
        <sequence>
            <element name="responders-identifier" type="btp:identifier"/>
            <element name="status-list">
                <complexType>
                    <sequence>
                        <element name="status-item" maxOccurs="unbounded">
                            <complexType>
                                <sequence>
                                    <element name="inferior-identifier" type="btp:identifier"/>
                                    <element name="status">
                                        <simpleType>
                                            <restriction base="string">
                                                <enumeration value="active"/>
                                                <enumeration value="resigned"/>
                                                <enumeration value="preparing"/>
                                                <enumeration value="prepared"/>
                                                <enumeration value="autonomously-confirmed"/>
                                                <enumeration value="autonomously-cancelled"/>
                                                <enumeration value="confirming"/>
                                                <enumeration value="confirmed"/>
                                                <enumeration value="cancelling"/>
                                                <enumeration value="cancelled"/>
                                                <enumeration value="cancel-contradiction"/>
                                                <enumeration value="confirm-contradiction"/>
                                                <enumeration value="hazard"/>
                                                <enumeration value="invalid"/>
                                            </restriction>
                                        </simpleType>
                                    </element>
                                    <element ref="btp:qualifiers" minOccurs="0"/>
                                </sequence>
                            </complexType>
                        </element>
                    </sequence>
                </complexType>
            </element>
            <element ref="btp:qualifiers" minOccurs="0"/>
            <element name="target-additional-information" type="btp:additional-information" minOccurs="0"/>
```

```
            </sequence>
            <attribute name="id" type="ID" use="optional"/>
        </complexType>
    </element>
</schema>
```