UNIVERZITET U BEOGRADU

FAKULTET ORGANIZACIONIH NAUKA

Marija P. Janković

SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI U METODOLOŠKIM PRISTUPIMA RAZVOJU IS

Doktorska disertacija

Beograd, 2016.

UNIVERSITY OF BELGRADE

FACULTY OF ORGANIZATIONAL SCIENCES

Marija P. Janković

SPECIFICATION OF INTEROPERABILITY ASPECTS IN METHODOLOGICAL APPROACHES TO IS DEVELOPMENT

Doctoral dissertation

Belgrade, 2016.

Mentor:

prof. dr Zoran Marjanović,

redovni profesor Fakulteta organizacionih nauka u Beogradu

Članovi komisije:

prof. dr Dušan Starčević,
redovni profesor Fakulteta organizacionih nauka u Beogradu

prof. dr Ivan Luković,
redovni profesor Fakulteta tehničkih nauka u Novom Sadu

Datum odbrane:

_____

Datum promocije:

_____

Disertaciju posvećujem mojim dragim roditeljima

Snežani i Predragu

Zahvalnica

Najpre bih želela da izrazim izuzetnu zahvalnost mom mentoru, ***Prof. dr Zoranu Marjanoviću****, koji mi je pružio ovu jedinstvenu životnu priliku, obezbedio sve potrebne uslove i što je svojim izvanrednim primerom podstakao moj profesionalni razvoj. Neizmerno hvala na ukazanom poverenju, izuzetnoj stručnosti i pomoći prilikom izrade doktorske disertacije.*

Posebnu zahvalnost dugujem ***dr Nenadu Aničiću*** *na stručnim idejama koje su bitno unapredile rad, velikom istraživačkom entuzijazmu i nesebičnoj spremnosti da mi u svakom trenutku pomogne. Hvala na divnoj prijateljskoj podršci koja mi je pomogla da istrajem kad je bilo najteže.*

Veliku zahvalnost bih želela da iskažem ***dr Slađanu Babarogiću*** *na uloženom vremenu i veoma konstruktivnim komentarima i sugestijama.*

Hvala kolegi ***Miroslavu Ljubičiću*** *na izuzetnoj i efikasnoj saradnji, korisnim idejama i velikoj pomoći oko tehničke realizacije rada. Hvala kolegama* ***Srđi Bjeladinoviću*** *i* ***Eleni Milovanović****, koji su uvek bili tu mi sa voljom i od srca pruže bilo koju vrstu pomoći. Hvala koleginici* ***Biljani Nedić*** *na ogromnoj pomoći u lektorisanju i pri izradi nekih tehničkih detalja rada.*

Svojim dragim roditeljima ***Snežani i Predragu****, dugujem duboku zahvalnost za ogroman napor i odricanja koja su nesebično uložili da moja deca ne osete ukradeno vreme poklonjeno pisanju ove disertacije. Bez njihove ljubavi, vere, optimizma i podrške ovaj rad ne bi bio moguć.*

Hvala mojim divnim sestrama ***Anamariji, Marini i Jeleni*** *koje su uvek verovale u mene i pružale mi nesebičnu podršku u svakom smislu.*

Posebno hvala mojoj deci ***Dimitriju, Kleoniki i Penelope****, kao i suprugu* ***Andreasu****, koji su moja najveća ljubav i nadahnuće.*

Marija Janković

Beograd, jun 2016. godine

SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI U METODOLOŠKIM PRISTUPIMA RAZVOJU IS

Rezime

Ova disertacija se bavi problemom specifikacije aspekata interoperabilnosti u metodološkim pristupima za razvoj informacionih sistema. U uslovima opšte globalizacije i savremene saradnje se umesto čvrste integracije teži ka slabom povezivanju organizacionih sistema. Način na koji heterogeni, slabo-povezani sistemi mogu da ostvare efikasne inter-organizacione veze je jedna od aktuelnih tema istraživanja oblasti interoperabilnosti informacionih sistema.

Na osnovu analize relevantne literature, imajući u vidu preporuke Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application (ATHENA) referentnog modela za konceptulnu integraciju, odabrana su tri relevantna aspekta za specifikaciju interoperabilnosti: aspekt procesa, servisa i informacija. U disertaciji se definiše nov specifičan pristup za specifikaciju aspekata interoperabilnosti u metodološkim pristupima za razvoj informacionih sistema. Predloženi pristup je baziran na principima opšteg sistemsko teorijskog modela životnog ciklusa softvera koji ima tri osnovne faze: identifikaciju, realizaciju i implementaciju. Za svaku od faza su precizno definisani opšti koraci i date su preporuke za njihovu primenu.

U fazi identifikacije se predlaže da se za specifikaciju inter-organizacionih poslovnih procesa pored funkcionalnog uključi i procesni pogled. U prvom koraku se identifikuju zahtevi za interoperabilnošću koji podrazumevaju specifikaciju: esencijalnih interoperabilnih poslovnih funkcija i poslovnih partnera koji učestvuju u kolaboraciji. Za reprezentaciju kolaborativnog poslovnog procesa je odabrana Business Process Model and Notation (BPMN) notacija. U drugom koraku se definišu opšti dijagrami konverzacije i kolaboracije, dok se u trećem koraku predlaže detaljna specifikacija njihove javne i privatne reprezentacije. U poslednjem koraku

faze identifikacije se vrši kreiranje nove ili izbor postojeće referentne ontologije, koja predstavlja osnovu za nedvosmislenu interpretaciju značenja poruka koje se razmenjuju u kolaboraciji.

Glavni doprinos disertacije je vezan za fazu realizacije, gde je omogućena detaljna specifikacija privatnog kolaborativnog poslovnog procesa koristeći koncepte referentne ontologije. U prvom koraku je izvršena ekstenzija BPMN meta-modela, pomoću koje poruke privatnog kolaborativnog procesa mogu da se definišu kao pogledi nad referentnom ontologijom. U narednom koraku je predložen način specifikacije servisa kolaborativnog procesa, korišćenjem Servis Adapter paterna sa ciljem da se postigne lakša transformacija specifikacije u programski kod. U poslednjem koraku je definisan UML View Profil koji je omogućio mapiranje lokalne konceptualne šeme na odgovarajuće koncepte globalne referentne ontologije, definisanjem odgovarajućih Object Constraint Language (OCL) pravila.

U samom radu je prikazana primena predloženog pristupa na izabranom poslovnom domenu lanaca snadbevanja i konkretnom realnom Iventory Visibility & Interoperability (IV&I) Electronic Kanban (eKanban) poslovnom procesu. Poseban značaj predloženog pristupa se ogleda u njegovoj zasnovanosti na opštem sistemsko-teorijskom pristupu životnog ciklusa softvera. U disertaciji je pokazano da su tri osnovne faze: identifikacija, realizacija i implementacija dovoljno opšte i da se predloženi koraci za svaku od faza, mogu primeniti za specifikaciju aspekata interoperabilnosti u različitim metodološkim pristupima za razvoj informacionih sistema. Bitna praktična prednost predloženog pristupa se ogleda u povezivanju privatnog kolaborativnog poslovnog procesa i referentne ontologije. Predloženi pristup specifikacije aspekata interoperabilnosti predstavlja teorijski okvir za automatizaciju razvoja interoperabilnih informacionih sistema, kao i odgovarajućih Computer-aided Software Engineering (CASE) alata kojim bi se povećala produktivnost modelovanja.

Ključne reči:

Razvoj informacionih sistema, Metodologija, Interoperabilnost, Proširenje BPMN-a, UML Profili

Naučna oblast:

Tehničke nauke

Uža naučna oblast:

Informacioni sistemi

UDK broj: 0.04

SPECIFICATION OF INTEROPERABILITY ASPECTS IN METHODOLOGICAL APPROACHES TO IS DEVELOPMENT

Abstract

This thesis addresses the problem of specification of interoperability aspects in methodological approaches to information system development. Under the conditions of general globalization and modern cooperation, instead of firm integration there is a tendency towards weak linking of organizational systems. The manner in which weakly linked systems may achieve efficient inter-organizational connections is one of the up-to-date research topics in the information systems' interoperability domain.

Based on the analysis of relevant literature and bearing in mind recommendations of the Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application (ATHENA) reference model for conceptual integration, three relevant aspects were selected for the specification of interoperability, namely: process, service and informations aspects. This thesis defines a new specific approach to specification of interoperability aspects in methodological approaches for informations systems' development. The proposed approach is based on the „System-Theoretic life cycle“ having three fundamental phases: identification, realization and implementation. For each of these phases general steps have been precisely defined and recommendations for their application given.

As for the identification phase, for specification of the inter-organizational business processes it is proposed to include the process view in addition to the functional one. The first step identifies interopebability requirements implying specification of the following: essential interoperable business functions and business partners participating in the collaboration. For representation of the collaborative business process Business Process Model and Notation (BPMN) has been selected. The second step defines general conversation and collaboration diagrams, while the

third step proposes a detailed specification for their private and public representation. The last step in the identification phase implies creation of a new or selection of the existing reference ontology representing a basis for an unambiguous interpretation of the meaning of messages being exchanged in scope of the collaboration.

Major contribution of this thesis is related to the realization phase where it enables an in-depth specification of the private collaborative business process using reference ontology concepts. In the first step extension of the BPMN meta-model has been performed and used to define the private collaborative process messages as the reference ontology views. In the next step the manner of the collaborative process service specification has been proposed, using a Service Adapter pattern in order to facilitate transformation of the specification into the programming code. The UML View Profile was defined in the last step, thus enabling mapping of the local conceptual scheme to corresponding global reference ontology concepts by defining adequate Object Constraint Language (OCL) rules.

The paper focuses on the application of the proposed approach on the selected supply chain business domain and concrete and actual Iventory Visibility & Interoperability (IV&I) Electronic Kanban (eKanban) business process. Special relevance of the proposed approach is reflected in its footing in the „System-Theoretic life cycle“ approach. The thesis demonstrates that three basic phases, namely: identification, realization and implementation are generic enough so as for the proposed steps for each of the phases to be applied for specification of interoperability aspects within different methodological approaches to information system development. An important practical advantage of the proposed approach lies in linking the private collaborative business process with the reference ontology. The proposed approach to specification of interoperability aspects represents a theoretical framework for automation of interoperable information systems' development, same as of appropriate Computer-aided Software Engineering (CASE) tools to increase the modeling productivity.

Keywords:

Information systems development, Methodology, Interoperability, BPMN Extension, UML Profiles

Scientific Area:

Technical Sciences

Specific Scientific Area:

Information Systems

UDK Number: 0.04

SADRŽAJ

1. UVOD

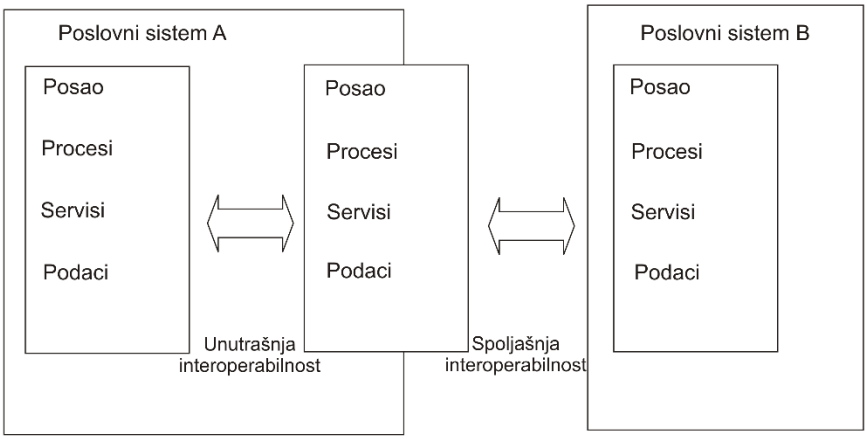
U ovom delu rada opisani su problem, predmet i ciljevi istraživanja doktorske disertacije. Shodno tome, definisane su opšta i posebne hipoteze koje će istraživanjem biti potvrđene ili odobrene. Sažeto je opisano rešenje koje se predlaže disertacijom. Na kraju uvodnog dela je dat kratak pregled ostalih poglavlja disertacije.

1.1. Problem i predmet istraživanja

Problem koji će biti analiziran u okviru ovog rada je specifikacija aspekata interoperabilnosti u metodološkim pristupima razvoju informacionih sistema. Analiza i specifikacija interoperabilnih softverskih sistema je komplikovana, budući da ne postoji opšte prihvaćen i proveren metodološki pristup koji omogućava da se kompleksni problemi interoperabilnosti sagledaju sa različitih aspekata.

Uspeh interneta je doveo do pojave Busines-to-Business (B2B) poslovnih sistema, koje možemo da definišemo kao informacione sisteme poslovnih subjekata koji obavljaju poslovne transakcije sa drugim subjektima elektronskom razmenom podataka. U uslovima savremene saradnje se umesto čvrste integracije, teži ka slabom povezivanju organizacionih sistema. Iako stupaju u kolaboraciju, poslovne organizacije ostaju ekonomski autonomne, i nije isključena mogućnost konkurencije između njih. Inter-organizacione veze se smatraju bitnim trendom informacione ere. Koncept inter-organizacionih poslovnih procesa je implementiran početkom novog milenijuma. Na samom početku poslovni partneri su u potpunosti skrivali detalje internog poslovanja, odnosno poslovni procesi su tretirani kao *crne kutije*. Sa ciljem poboljšanja kolaboracije, *crne kutije* su postepeno transformisane u *sive*, za koje je karakteristično izlaganje onih poslovnih procesa koji su od značaja za međusobnu saradnju. Model kolaboracije po principu *bele* kutije nije zastupljen u praksi, budući da zahteva čvrstu integraciju između poslovnih partnera.

Način na koji heterogeni, slabo povezani sistemi mogu da ostvare efikasnu kolaboraciju na poslovnom i tehničkom nivou, je jedna od bitnih tema istraživanja interoperabilnosti. Interoperabilnost inter-organizacionih poslovnih sistema se odnosi na mogućnost dva ili više sistema ili komponenti da razmenjuju podatke i da upotrebljavaju razmenjene podatke (IEEE, 1991). U zavisnosti od zahtevanog nivoa kolaboracije između B2B sistema, mogu da se definišu četiri različita nivoa interoperabilnosti koja su ilustrovana na slici 1.1: (1) interoperabilnost podataka, (2) interoperabilnost servisa, (3) interoperabilnost procesa i (4) poslovna interoperabilnost (ATHENA D.A4.2, 2007).



Slika 1.1 Nivoi interoperabilnosti. Preuzeto i modifikovano prema (ATHENA D.A4.2, 2007)

Tradicionalni način za podršku **interoperabilnosti podataka** je usvajanje standarda kao što su Electronic Data Interchange (EDI), United Nations Directories for Electornic Data Interchange for Administration, Commerce and Transport (UN/EDIFACT) ili Open Application Group Integration Specification (OAGIS). Od tehnologija koje se mogu primeniti za efikasnu realizaciju **interoperabilnosti servisa** istaknućemo Servisno-Orijentisanu Arhitekturu (SOA) i Web Servise (WS). Budući da koncepti SOA podržavaju opis i slabo povezivanje distribuiranih servisa, predstavljaju dobru osnovu za implementaciju intra-organizacionih scenarija. **Interoperabilnost procesa** se tiče pronalaženja rešenja za povezivanje internih procesa dva ili više poslovna sistema sa ciljem realizacije zajedničkog kolaborativnog procesa. Realizacija interoperabilnosti poslovnih procesa je podržana brojnim standardima za modelovanje i implementaciju distribuiranih procesa kao što su na primer Web Services Description Language (WSDL) i Business

Process Execution Language (BPEL). Nivo **interoperabilnosti poslovnog sistema** se bavi problemima interoperabilnosti koji su posledica različite poslovne vizije, kulture, organizacione strukture i pravila poslovanja.

Da bi se obezbedila efikasna interoperabilnost između B2B sistema, potrebno je realizovati interoperabilnost na svim nivoima. Navedeni pristupi za rešavanje problema interoperabilnosti na različitim nivoima, imaju svoje prednosti ali se kao bitan nedostatak ističe činjenica da ne postoji integrisani, sveobuhvatan metodološki pristup za sistematičan razvoj interoperabilnih poslovnih sistema. **Širi predmet istraživanja** ove disertacije je upravo podrška metodoloških pristupa za realizaciju interoperabilnosti poslovnih sistema. Jedan od bitnih preduslova uspešne realizacije interoperabilnosti je njena jasna i precizna specifikacija. **Uži predmet** istraživanja kojim se disertacija detaljno bavi je specifikacija različitih aspekata interoperabilnosti u metodološkim pristupima razvoju informacionih sistema.

1.2. Ciljevi i hipoteze istraživanja

Polazeći od definisanog predmeta istraživanja, postavlja se za cilj razvoj novog originalnog pristupa za specifikaciju aspekata interoperabilnosti u metodološkim pristupima za razvoj informacionih sistema. Faze predloženog pristupa treba da budu dovoljno opšte, kako bi mogle da se primene za specifikaciju aspekata interoperabilnosti u različitim metodološkim pristupima za razvoj IS. Za svaku od faza je potrebno precizno definisati neophodne korake.

Na osnovu analize dostupne literature i postavljenog predmeta i cilja istraživanja može se postaviti opšta hipoteza:

- Moguće je definisati proširenje postojećih metodoloških pristupa za razvoj informacionih sistema (IS) kako bi se obezbedila podrška za realizaciju interoperabilnosti.

Na osnovu opšte hipoteze su izvedene sledeće pojedinačne:

- Moguće je identifikovati i sistematizovati interoperabilne zahteve.

- Moguće je formalizovati proširenje postojećih modela i tehnika.
- Moguće je formalizovati uvođenje novih modela i tehnika.
- Moguće je verifikovati ostvarene rezultate proširenja metodoloških pristupa na primeru dovoljno kompleksnog realnog sistema.

1.3. Kratak pregled predloženog rešenja

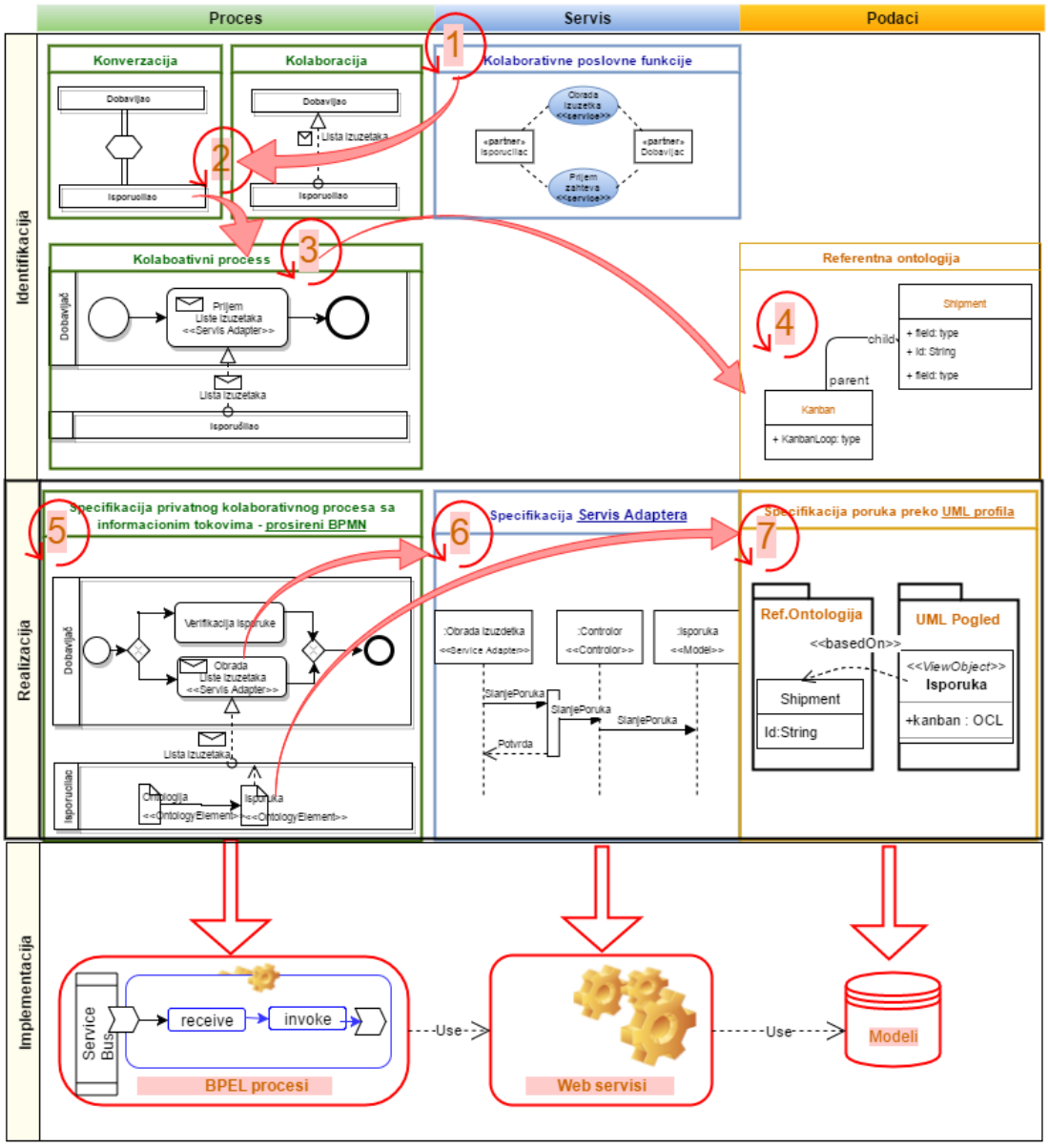
Na osnovu preporuka Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application (ATHENA) referentnog modela za konceptualnu integraciju identifikovana su četiri osnovna aspekta interoperabilnosti: informacioni aspekt, aspekt servisa, aspekt procesa i ne-funkcionalni aspekti (ATHENA D.A4.2, 2007). U doktorskoj disertaciji je predložen pristup za specifikaciju aspekata interoperabilnosti koji se bazira na teorijskim osnovama sistemsko-teorijskog pristupa razvoju softvera. Za svaku od tri osnovne faze: identifikaciju, realizaciju i implementaciju su identifikovani opšti koraci, za koje je pokazano da mogu da se primene za specifikaciju aspekata interoperabilnosti u različitim metodološkim pristupima. Kao značajna karakteristika predloženog pristupa specifikaciji aspekata interoperabilnosti se ističe njegova uniformnost i mogućnost primene u različitim metodološkim pristupima. U zavisnosti od konkretnih tehnika koje se primenjuju u opštim fazama, analizirane su prednosti i nedostaci pojedinačnih metodologija.

Osnovni cilj faze identifikacije standardnih metodoloških pristupa za razvoj IS je identifikacija esencijalnih poslovnih funkcija. Rezultat ove faze je model funkcija poslovnog sistema za čiju identifikaciju može da se koristi bilo koja konvencionalna ili objektna metoda za funkcionalnu specifikaciju sistema. Sa stanovišta specifikacije aspekata interoperabilnosti, ključna je identifikacija kolaborativnih poslovnih procesa između partnera. Iz tog razloga se u tezi predlaže da se pored funkcionalnog, uključi i procesni pogled. Naime, u fazi identifikacije je bitno identifikovati: (1) one poslovne esencijalne funkcije koje treba da podrže interoperabilnost i (2) poslovne partnere koji učestvuju u kolaboraciji. Na slici 1.2, u prvom koraku faze identifikacije je prikazan način predloga formalizacije identifikovanih poslovnih funkcija i

relevantnih partnera. Na proširenom dijagramu slučajeva korišćenja se za prikaz identifikovanih partnera koji učestvuju u kolaboraciji uvodi sterotip <<partner>>, dok su relevantni poslovni servisi označeni pomoću stereotipa <<servis>>. U drugom koraku se na osnovu prethodno identifikovanih elemenata kolaborativnog procesa, kreiraju odgovarajući Business Process Model and Notation (BPMN) modeli. Za početnu identifikaciju kolaborativnih poslovnih procesa se predlaže dijagram konverzacije, gde se definišu logički povezane grupe poruka. Nakon identifikacije pojedinačnih poruka u konverzaciji, kreira se odgovarajući dijagram kolaboracije. U trećem koraku faze identifikacije se vrši specifikacija privatnog kolaborativnog poslovnog procesa. Javni kolaborativni poslovni proces se kreira na osnovu njegove privatne reprezentacije, apstrakcijom internih poslovnih aktivnosti. U četvrtom koraku faze identifikacije se vrši kreiranje ili izbor referentne ontologije, kako bi se obezbedila nedvosmislena interpretacija značenja poruka koje se razmenjuju u kolaboraciji.

Nakon identifikacije kolaborativnih procesa, u fazi realizacije se vrši detaljna specifikacija informacionih tokova privatnog kolaborativnog poslovnog procesa. Jedan od osnovnih doprinosa doktorske disertacije je uspostavljanje veze kolaborativnog poslovnog procesa sa referentnom ontologijom. U petom koraku predloženog pristupa se vrši specifikacija informacionih tokova privatnog kolaborativnog procesa. Poruke kolaborativnog procesa su definisane kao pogledi nad globalnom referentnom ontologijom. Kako bi se na dijagramu kolaborativnog procesa omogućila adekvatna reperezentacija poruka i njihove veze sa referentnom ontologijom, u radu je predložena ekstenzija BPMN meta-modela. U šestom koraku je definisano proširenje Unified Modeling Languge (UML) dijagrama sekvenci za adekvatnu specifikaciju servisa. Naime, u ovom koraku se predlaže definisanje logičke arhitekture u skladu sa Servis Adapter Paternom i daju se preporuke za njegovu implementaciju. U poslednjem koraku faze realizacije se vrši specifikacija strukture poruka u skladu sa elementima referentne ontologije. Definisan je UML View Profil koji omogućava mapiranje lokalne konceptualne šeme na relevantne koncepte referentne ontologije, definisanjem odgovarajućih Object Constraint Language (OCL) pravila.

Definisani modeli se čuvaju u bazi, kako bi mogli da se iskoriste u fazi implementacije za potrebe automatske transformacije. Na slici 1.2 je prikazan predlog jednog mogućeg rešenja, gde se za implementaciju kolaborativnog poslovnog procesa preporučuje transformacija u BPEL, dok se za implementaciju servisa predlažu Web Servisi.



Slika 1.2 Specifikacija aspekata interoperabilnosti

Za validaciju predloženog pristupa specifikacije aspekata interoperabilnosti, definisani su kriterijumi validacije koji su na najvišem nivou klasifikovani na četiri osnovne klase zahteva: (1) podrška za apstrakciju kolaborativnog procesa, (2)

podrška za dizajn kolaborativnog procesa, (3) podrška za efikasnu kompoziciju kolaborativnog procesa i (4) podrška globalne poslovne informacione šeme. Za svaku od četiri osnovne klase zahteva, identifikovani su relevantni podzahtevi. U prvom koraku je sprovedena analiza na osnovu koje su identifikovani zahtevi klasifikovani u skladu sa tri osnovne faze sistemsko-teorijskog pristupa: identifikacijom, realizacijom i implementacijom. Za svaku od faza je prodiskutovano da li predloženi pristup specifikacije aspekata interoperabilnosti omogućava adekvatno ispunjenje zahteva ili ne.

Predloženi pristup specifikacije aspekata interoperabilnosti je ilustrovan na realnom primeru Inventory Visibilty and Interoperability (IV&I) eKanban poslovnog sistema primenom FON Labis metodologije za projektovanje informacionih sistema i Larmanove metode. Izvršeno je poređenje i ukazano je na prednosti, odnosno nedostatke jednog pristupa u odnosu na drugi.

U radu je u skladu sa identifikovanim zahtevima za interoperabilnošću, formulisan originalni pristup za specifikaciju aspekata interoperabilnosti u metodološkim pristupima za razvoj IS. Predloženo rešenje omogućava: (1) proširenje postojećih metodoloških pristupa sa ciljem da se na adekvatan način reprezentuju potrebni aspekti interoperabilnosti i (2) formalizaciju uvođenja novih modela i tehnika po potrebi.

1.4. Struktura rada

Doktorska disertacija je organizovana na sledeći način:

U **drugom poglavlju** je dat pregled najćešće citiranih definicija interoperabilnosti, i objašnjeni su osnovni koncepti interoperabilnosti koji se tiču različitih nivoa, prepreka i pristupa interoperabilnosti. Posebno su analizirana četiri osnovna aspekta interoperabilnosti koja su bitna za konceptualnu integraciju sistema: informacioni aspekt, aspekt servisa, aspekt procesa i nefunkcionalni aspekti.

U okviru **trećeg poglavlja** su izloženi osnovni koncepti metodoloških pristupa za razvoj informacionih sistema. Nakon određenja pojma informacionog sistema, objašnjeni su osnovni principi sistemsko-teorijskog modela životnog ciklusa, koji

predstavlja teorijsku osnovu predloženog pristupa za specifikaciju aspekata interoperabilnosti. Dat je pregled različitih metodoloških pristupa za razvoj informacionih sistema, i ukazano je na značaj relevantne podrške za specifikaciju interoperabilnosti. Zatim su prikazane osnovne faze i osobine FON Labis metodologije za razvoj informacionih sistema i Larmanove metode razvoja softvera koje su odabrane za validaciju predloženog pristupa specifikacije aspekata interoperabilnosti.

U **četvrtom poglavlju** je dat pregled postojećih pristupa koji se bave rešavanjem problema interoperabilnosti u razvoju informacionih sistema. U prvom delu su prikazane karakteristike reprezentativnih poslovnih arhitektura (eng. business architectures), koje su bile od koristi za sagledavanje relevantnih aspekata kolaborativnog poslovnog procesa. U drugom delu poglavlja fokus je usmeren na detaljnu analizu onih okvira interoperabilnosti, koji za cilj imaju reprezentaciju kolaborativnih poslovnih procesa na konceptualnom i tehničkom nivou. Na kraju je dat pregled osnovnih okvira Servisno-orijentisane arhitekture (SOA). Dok su prva dva pristupa bila od koristi za identifikaciju poslovnih aspekata, SOA obezbeđuje bitne koncepte za implementaciju poslovnih procesa.

U **petom poglavlju** je prikazan pristup za specifikaciju aspekata interoperabilnosti u metodološkim pristupima za razvoj IS koji se predlaže u disertaciji. Za fazu identifikacije, realizacije i implementacije sistemsko-teorijskog pristupa su definisani potrebni koraci za specifikaciju aspekata interoperabilnosti, koji su prikazani na slici 1.2. Na kraju poglavlja, sumirani su zahtevi za interoperabilnošću kolaborativnih poslovnih sistema, na osnovu kojih su definisani kriterijumi za validaciju predloženog pristupa.

U **šestom poglavlju** je prikazano korišćenje predloženog pristupa za specifikaciju aspekata interoperabilnosti na realnom primeru IV&I eKanban poslovnog sistema primenom FON Labis metodologije za projektovanje informacionih sistema i Larmanove metode. Izvršeno je poređenje i ukazano je na prednosti, odnosno nedostatke jednog pristupa u odnosu na drugi.

Na kraju rada su zaključak i literatura. U okviru zaključka ukratko su izloženi i dalji pravci istraživanja.

2. POJAM INTEROPERABILNOSTI INFORMACIONIH SISTEMA

U ovom poglavlju je dat pregled najčešće citiranih definicija interoperabilnosti, i objašnjeni su osnovni koncepti interoperabilnosti. Posebno su analizirana četiri osnovna aspekta interoperabilnosti koja su bitna za konceptualnu integraciju sistema: informacioni aspekt, aspekt servisa, aspekt procesa i nefunkcionalni aspekti. Na kraju poglavlja je objašnjen pojam zahteva za interoperabilnošću.

2.1. Definicija interoperoperabilnosti

Reč *interoperativan* ukazuje na činjenicu da jedan sistem obavlja neku operaciju umesto drugog sistema (Chen & Doumeingts, 2003; Chen & Vernadat, 2003). Prema Vernadatu, interoperabilnost je preduslov uspešne komunikacije između aplikacija. On definiše interoperabilnost kao mogućnost jedne aplikacije da razmenjuje informacije sa drugom aplikacijom na bazi zajedničkog razumevanja i da uspešno koristi njene funkcionalnosti (ili servise) (Francois Vernadat, 1996). Prema Chen-u koncept *interoperabilnosti* je usko vezan za oblast projektovanja softverskih sistema i može da se bliže odredi kao mogućnost lake i jednostavne saradnje između dva softverska sistema (Chen & Doumeingts, 2003). Drugim rečima, on podrazumeva da dva interoperabilna softverska sistema mogu da međusobno sarađuju bez posebnih prepreka.

Ipak, pojam interoperabilnosti se ne odnosi samo na mogućnost saradnje između različitih softverskih sistema. The Open Group Application Fremework (TOGAF) definiše interoperabilnost kao: (1) mogućnost dva ili više sistema ili komponenti da razmenjuju i da koriste razmenjene informacije i (2) sposobnost razmene i korišćenja servisa sa ciljem realizacije efikasne međusobne saradnje (Open Group, 2000).

Pojam interoperabilnosti poslovnih sistema (eng. enterprise interoperability) se prema (Ziemann, 2010) odnosi na sposobnost interakcije između dva ili više

poslovnih sistema kao celina, ili njihovih pojedinačnih delova. Cilj je da se naglasi da se pojam *interoperabilnosti poslovnih sistema* generalno može analizirati sa dva aspekta: (1) *unutrašnja interoperabilnost* (eng. intra interoperability) se odnosi na inteoperabilnost između unutrašnjih jedinica (podsistema) jednog poslovnog sistema i (2) *spoljašnja interoperabilnost* (eng. inter interoperability*)* koja uključuje interoperabilnost na nivou proširenih poslovnih sistema (eng. extended enterprises), virtuelnih poslovnih sistema (eng. virtual enterprises), mrežnih i distribuiranih sistema. U uvodu (slika 1.1) su prikazani nivoi interoperabilnosti između dva poslovna sistema: poslovni, nivo procesa, servisa i podataka. Za uspešnu kolaboraciju između dva poslovna sistema je poželjno obezbediti interoperabilnost na svim nivoima (ATHENA D.A4.2, 2007).

Bitno je ukazati na razliku između pojma integracije i interoperabilnosti. Prema (Ziemann, 2010), pojam integracije je usko povezan sa definicijom interoperabilnosti kao sposobnosti sistema da rade zajedno. Grupe autora ukazuju na razliku između čvrsto i slabo povezanih sistema (Chen, Doumeingts, & Vernadat, 2008; Ziemann, 2010). U slučaju potpuno integrisanog sistema komponente su čvrsto povezane (eng. tightly coupled), dok je za interoperabilnost karakteristična slaba povezanost komponenti (eng. loosely coupled).

2.2. Koncepti interoperabilnosti

Postoje brojni koncepti koje je potrebno identifikovati kako bi se definisao kompleksan pojam interoperabilosti poslovnih sistema. U ovom poglavlju je opisan skup osnovnih koncepata, koji su selektovani kao bitni na osnovu detaljne analize postojeće literature.

2.2.1. Nivoi interoperabilnosti

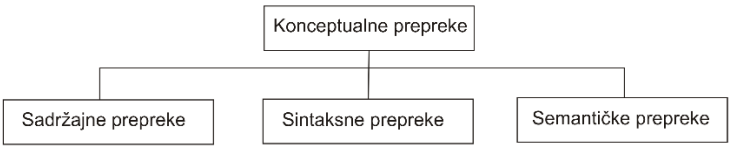
Analiza različitih nivoa interoperabilnosti koji su definisani u aktuelnim okvirima, je poslužila kao polazna osnova za selekciju relevantnih aspekata interoperabilnosti. Osnovu *Athena Interoperability Framework-a (AIF)* čine četiri nivoa koja su prikazana na slici 1.1: nivo poslovanja, procesa, servisa i podataka (ATHENA D.A4.2, 2007). *Ideas Development for Enterprise Application and Software (IDEAS)* okvir

smatra da je bitno ostvariti komunikciju na tri osnovna nivoa: poslovni nivo, nivo znanja, informaciono-komunikacioni nivo. Dodatna semantička dimenzija se uvodi kako bi se postiglo razumevanje na prethodna tri nivoa (Blanc, 2005). U skladu sa metodološkim osnovama *Evropskog centra za interoperabilnost* bitno je postići interoperabilnost na tehničkom, sintaksnom i semantičkom nivou (Fraunhofer ISST, 2009). *European Interoperability Framework (EIF)* definiše interoperabilnost na tehničkom, semantičkom, organizacionom i pravnom nivou (ISA, 2010). *C4 Interoperability Framework (C4)* definiše četiri nivoa interoperabilnosti po čijim početnim slovima je okvir dobio ime: konekcija (eng. connection), komunikacija (eng. communication), konsolidacija (eng. consolidation) i kolaboracija (eng. collaboration) (Tarabanis, 2006). Odabrani okviri koji su relevantni za predmet istraživanja disertacije i njihovi nivoi su detaljno analizirani i opisani u četvrtom poglavlju.

2.2.2. Prepreke interoperabilnosti

Prema (ATHENA D.A4.2, 2007), ***prepreke interoperabilnosti*** predstavljaju fundamentalan koncept za definisanje domena interoperabilnosti. Brojne prepreke interoperabilnosti zavise od specifične oblasti primene. Jedan od ciljeva INTEROP projekta je bila identifikacija opštih prepreka interoperabilnosti (Arne-Jorgen Berre et al., 2004). Pod preprekama interoperabilnosti (eng. barriers) se podrazumeva nekompatibilnost dva sistema ili komponente da razmenjuju i koriste informacije ili određene servise. Identifikovane su tri kategorije opštih prepreka: (1) konceptualne, (2) tehnološke i (3) organizacione.

Prema (Poler, Van Sinderen, & Sanchis, 2009), **konceptualne prepreke** se uglavnom tiču sintaksnih i semantičkih informacija koje se razmenjuju između dva poslovna sistema. Ovi problemi se tiču modelovanja na visokom nivou apstrakcije, ali i na nivou informacija. U opštem slučaju, konceptualne prepreke mogu da se klasifikuju na tri različita tipa koja su prikazana na slici 2.1: (1) prepreke po pitanju sadržaja, (2) sintaksne prepreke i (3) semantičke prepreke (Poler et al., 2009).



Slika 2.1 Tri osnovna tipa konceptualnih prepreka interoperabilnosti (Poler et al., 2009)

Prepreke po pitanju sadržaja se tiču obuhvata modela poslovnog sistema. Heterogenost koja dovodi do problema interoperabilnosti se tiče onih koncepata koji postoje u samo jednom od poslovnih sistema. *Sintaksne prepreke* se tiču jezika za modelovanje poslovnih sistema, dok se *semantičke prepreke* odnose na značenje termina koji se koriste. Modeli poslovnog sistema koji teže da predstave jedan deo sitema ili sistem kao celinu mogu da budu: neformalni (prirodni jezik), polu-formalni (jezici sa grafičkim formalizmom) ili formalni (matematički jezici). Značenje koncepata u modelu treba da bude jasno i precizno definisano.

Tehnološke prepreke se odnose na upotrebu računara ili *Information and Communication Technology (ICT)* za komunikaciju i razmenu informacija. Tehnološke prepreke se odnose na nekompatibilnost informacionih tehnologija (arhitekture i platforme, infrastrukture, itd.). Tiču se standarizacije predstavljanja, skladištenja, razmene i obrade podataka upotrebom računara. Komunikacionu prepreku predstavljaju protokoli koji se koriste prilikom razmene informacija. Prepreka u pogledu sadržaja uključuje standarde, tehnike i alate koji se koriste. Na primer, upotreba različitih tehnika za predstavljanje informacija, nekompatibilnost alata koji se koriste za kodiranje/dekodiranje informacija koje se razmenjuju. Infrastrukturne prepreke se tiču upotrebe nekompatibilnih platformi.

Organizacione prepreke se tiču nekompatibilnosti organizacione strukture i menadžment tehnika koje su implementirane u dva poslovna sistema. Različiti načini definisanja odgovornosti i autoriteta kao rezultat imaju različite organizacione strukture, što dovodi do problema interoperabilnosti. Organizacione prepreke se odnose na definisanje odgovornosti i autorizacije kako bi se obezbedila realizacija interoperabilnosti. Može da se posmatra kao „ljudska tehnologija“ ili „ljudski faktor“ i odnosi se na ljudsko i organizaciono ponašanje koje može da bude nekompatibilno sa uslovima koji su potrebni za interoperabilnost. *Odgovornost*

mora da bude jasno i precizno definisana. *Autorizacija* je organizacioni koncept koji definiše ko ima pravo obavljanja određenog posla. Na primer, neophodno je definisati ko je autorizovan da kreira, modifikuje ili upravlja podacima, procesima, servisima, itd. Organizaciona struktura se odnosi na stil organizovanja odgovornosti, autorizacije i donošenja odluka. Na primer, organizacije mogu da budu centralizovane ili decentralizovane. Drugi primer se odnosi na strukturu organizacija koja može da bude: hijerarhijska, matrična ili mrežna. Organizacione prepreke predstavljaju dodatnu kategoriju prepreka. Dok su konceptualne prepreke koncentrisane na informacione probleme, tehnološke na mašinske, organizacione se tiču problema čiji su glavni uzročnici ljudi.

2.2.3. Pristupi interoperabilnosti

Za definisanje istraživanja u domenu interoperabilnosti, nije dovoljno identifikovati prepreke i rešenja za otklanjanje prepreka već je potrebno definisati pristup otklanjanju prepreka. Da bi se ostvarila interoperabilnost neophodno je da se bitni entiteti povežu na specifičan način. Prema ISO 14258 (Concepts and Rules for enterprise models) identifikovana su tri osnovna pristupa rešavanju problema interoperabilnosti: integrisani (eng. integrated), ujedinjeni (eng. unified) i federalni (eng. federated). Za *integrisni pristup* je karakteristično postojanje opšteg formata za sve modele. Ovaj fomat mora da bude na istom nivou detalja kao i modeli. Opšti format nije obavezno standardizovan, ali mora da bude opšte prihvaćen od strane svih partnera u kolaboraciji. Za *ujedinjeni pristup* opšti format postoji samo na meta nivou. Meta-model nije izvršan entitet, kao što je slučaj sa integrisanim pristupom. On predstavlja sredstvo za definisanje semantičke ekvivalencije kako bi se omogućilo mapiranje između modela. Za *federalni pristup* ne postoji opšti format. Kako bi se ostvarila interoperabilnost, partneri moraju da dele zajedničku ontologiju.

2.3. Aspekti interoperabilnosti

Različiti modeli se koriste da opišu različite aspekte softverskog sitema. Na osnovu preporuka ATHENA Referentnog modela za konceptualnu integraciju identifikovana su četiri osnovna aspekata interoperabilnosti: informacioni aspekt,

aspekt servisa, aspekt procesa i nefunkcionalni aspekti (ATHENA D.A4.2, 2007). U disertaciji se smatra da ova grupa aspekata predstavlja dovoljno dobru osnovu za diskusiju specifikacije interoperabilnosti poslovnih sistema na konceptualnom nivou, iako predložena lista aspekata nije konačna, i moguće je identifikovati dodatne aspekte.

2.3.1. Aspekt procesa

U literaturi postoje brojne definicije poslovnih procesa. Izdvojićemo definiciju koju je dao Davenport, budući da je često citirana: "poslovni proces predstavlja struktuiran, merljiv set aktivnosti koji je dizajniran da proizvede specifičan izlaz za specifičnog korisnika ili specifično tržište"(Davenport, 1993).

Ziemann definiše *poslovni proces* kao "sekvencu poslovnih funkcija koja ima za cilj kreiranje izlaza (eng. output) za interne ili eksterne korisnike " (Ziemann, 2010). On ističe da se poslovni proces sastoji od različitih aktivnosti ili funkcija. U kontekstu istraživanja informacionih sistema, *funkcija* predstavlja transformaciju ulaznih u izlazne objekte sa ciljem realizacije eksplicitno definisanog cilja. Slično procesu, funkcija ima za cilj realizaciju predefinisanog cilja. Na nižem nivou detalja poslovni procesi mogu da se posmatraju kao poslovne funkcije. Jedna funkcija može da bude složena i da se dekomponuje na određeni skup podfunkcija. Za razliku od procesa, opis funkcije ne definiše sekvencu izvršenja.

S obzirom na to da se termin *poslovni proces* koristi u brojnim kontekstima, u literaturi postoji veliki broj različitih kriterijuma za njegovu klasifikaciju (Ziemann, 2010). Budući da je tema disertacije specifikacija aspekata interoperabilnosti, naš primarni fokus čine **inter-organizacioni** poslovni procesi. U literaturi se koriste različiti temini koji se odnose na inter-organizacione procese: kros-organizacioni poslovni procesi (eng. cross-organizational business processes), kolaborativni poslovni procesi (eng. collaborative business processes), poslovna kolaboracija (eng. business collaboration), e-kolaboracija (e-collaboration), B2B integracija (eng. B2B-Integration) i poslovna interoperabilnost (business interoperability) (Ziemann, 2010).

Pre nego što definišemo pojam **kros-organizacionog** poslovnog procesa, ukazaćemo na osnovne sličnosti i razlike između **intra- i inter-organizacionih** poslovnih procesa. **Inter-organizacioni** scenario saradnje karakteriše: manji stepen deljenja implicitnog i eksplicitnog znanja, niži nivo poverenja između partnera, heterogene poslovne kulture, mogućnost konkurencije, različiti ciljevi i kraće vreme međusobne saradnje. U uslovima moderne saradnje slične karakteristike se odnose i na **intra-organizacione** scenarije, za koje su karakteristični profitni centri i kraće vreme koje je na raspolaganju za realizaciju projekata. Bitno je istaći da **kros-organizacioni** scenario saradnje zahteva da se eksplicitno opišu i ograniče parametri za zajednički rad na inter-organizacionom nivou.

U literaturi postoji veliki broj različitih definicija i tipova kros-organizacionih procesa. Forme i definicije pojma kros-organizacionih procesa u velikom broju slučajeva nisu međusobno konzistentne (Ziemann, 2010). Shvatanje pojma kolaboracije u ovoj tezi je bazirano na stavu da i kolaboracija i kooperacija mogu da se interpretiraju na sledeći način: "zajednički rad sa ciljem realizacije određene svrhe" (Unhelkar, Ghanbary, & Younessi, 2010). Ipak, u odnosu na kooperaciju, kolaboracija podrazumeva bližu formu saradnje, gde partneri imaju jednake uloge i mogu da rade nezavisno na realizaciji cilja.

Ziemann definiše **kolaborativni poslovni proces** kao "poslovni proces čije se aktivnosti izvršavaju od strane dve ili više autonomnih organizacija" (Ziemann, 2010). Jedna od bitnih karakteristika modela kolaborativnog poslovnog procesa je da model treba da obezbedi partnerima koji učestvuju u kolaboraciji sve potrebne informacije za zajedničko izvršenje procesa. Pritom je bitno naglasiti da partneri treba da vode računa o tome da ne otkriju više informacija nego što je potrebno. Ovo je ujedno i jedan od osnovnih zahteva za modelovanje kolaborativnih poslovnih procesa. Ostali zahtevi su sistematizovani i biće detaljno diskutovani u petom poglavlju (5.5.1). U skladu sa definisanim zahtevima potrebno je precizirati razliku između privatnih (eng. private), javnih (eng. public) i globalnih (eng. global) procesa.

2.3.1.1. Određenje pojma privatnog i javnog procesa

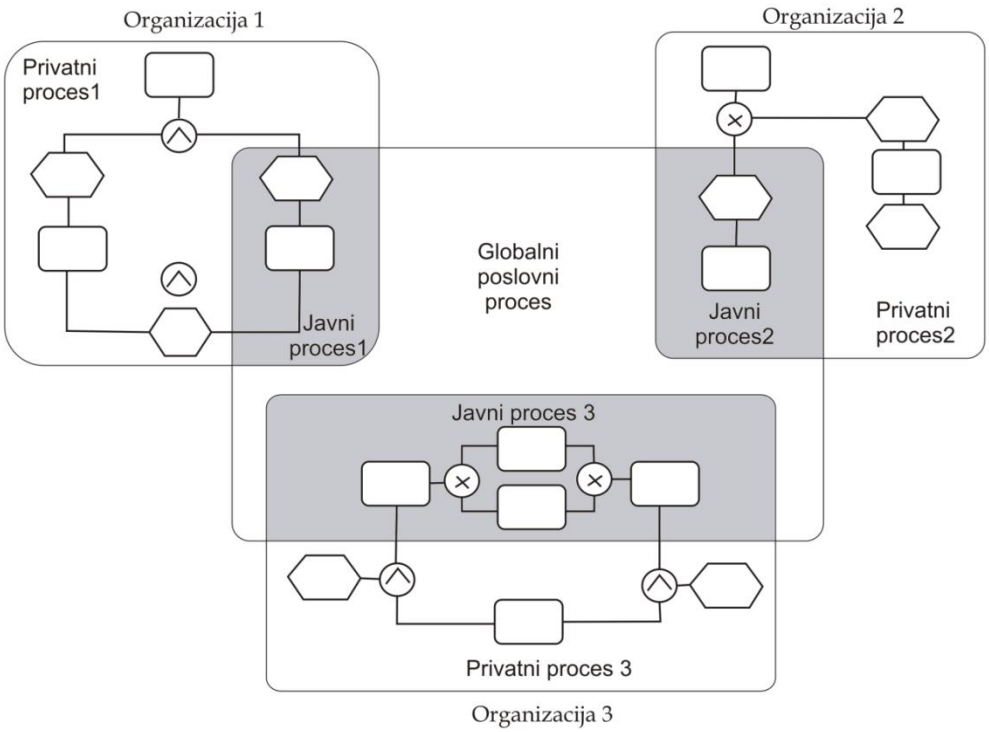
Privatni procesi se izvršavaju unutar jedne organizacije i mogu da sadrže informacije koje nisu javne. Oni predstavljaju najbolju poslovnu praksu i smatraju se intelektualnim vlasništvom (Schulz & Orlowska, 2004). Važno je napomenuti da neke od aktivnosti privatnog procesa mogu da služe za uspostavljanje kolaboracije sa poslovnim partnerima. Na primer, procesi kao što su slanje ili prijem poruke predstavljaju interne aktivnosti privatnog procesa. Ziemann definiše privatni proces kao "poslovni proces koji se kreira za internu upotrebu i reprezentuje samo one aktivnosti koje se izvršavaju interno. U opštem slučaju, modeli privatnih procesa ne bi trebalo da budu dostupni eksternim partnerima"(Ziemann, 2010).

Javni proces može da se izvede iz privatnog procesa, apstrakcijom svih poverljivih elemenata. Drugim rečima, javni proces treba da sadrži uglavnom aktivnosti koje su bitne za interakciju između poslovnih partnera u kolaboracionom procesu. U literaturi se koriste brojni termini za javni proces: apstraktni proces (eng. abstract process), workflow pogled (eng. workflow view), interfejs koreografije (eng. choreography interface), pogled procesa (eng. view process), itd. Grupa autora ograničava javni proces samo na aktivnosti koje su neophodne za komunikaciju između partnera u kolaborativnom procesu (Weske, 2012a; Ziemann, 2010).

Javni proces može da se posmatra kao interfejs jedne organizacije, koji uključuje samo one informacije koje su neophodne za interakciju sa jednim ili više partnera. Bitno je naglasiti da javni proces definiše sekvencu izvršenja aktivnosti. U skladu sa navedenim karakteristikama javnog procesa, Ziemann predlaže sledeću definiciju **javnog procesa**: "javni proces je poslovni proces koji uključuje one elemente privatnog procesa koji su relevantni i javni za jednog ili više kolaboracionih partnera. Njegova svrha je da opiše interakcije organizacije A sa partnerskim organizacijama (B, C, D,..) sa tačke gledišta organizacije A" (Ziemann, 2010).

Pored poređenja pojmova privatnog i javnog procesa, bitno je objasniti pojam **globalnog poslovnog procesa** (slika 2.2). Globalni proces opisuje aktivnosti više organizacija koje učestvuju u kolaborativnom procesu sa neutralne perspektive. Naime, osnovna razlika u odnosu na javni proces se ogleda u činjenici da javni proces

opisuje sekvencu interakcija sa perspektive jedne organizacije. Ziemann definiše globalni poslovni porces kao:“kolaborativni poslovni proces koji se sastoji samo od onih elemenata koji su javni i koji su relevantni za *sve* organizacije koje učestvuju u kolaboraciji. Njegova svrha je da opiše dozvoljene interakcije između procesa” (Ziemann, 2010).



Slika 2.2 Privatni, javni i globalni pogled na proces. Preuzeto i modifikovano prema (Ziemann, 2010)

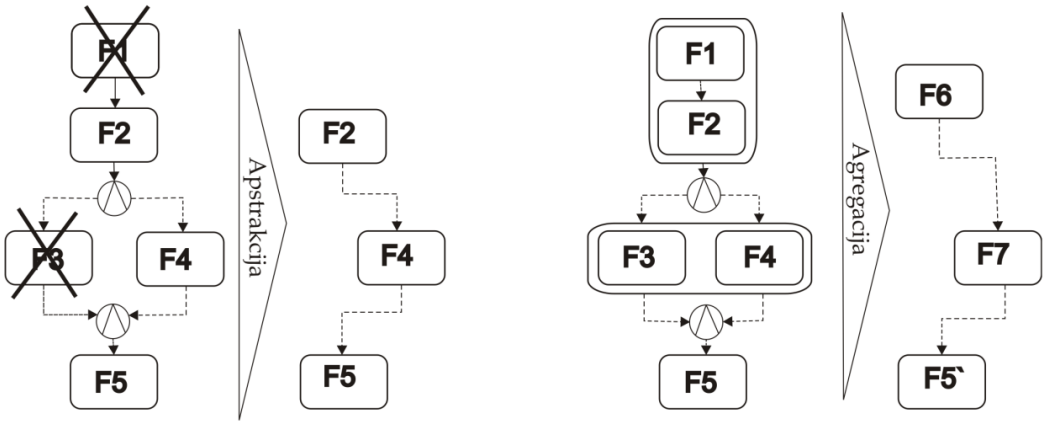
Sa aspekta treće organizacije, javni proces obuhvata interakcije sa prvom i drugom organizacijom. Važno je istaći da su interakcije između prve i druge organizacije van opsega javnog procesa koji je definisan sa aspekta treće organizacije. Teza se bavi problemima specifikacije kolaborativnih procesa sa aspekta jednog učesnika, tako da je problem realizacije gobalnih pogleda van opsega istraživanja.

2.3.1.2. Operacije između privatnih i javnih procesa

Svrha kreiranja javnog procesa je da se kolaboracioni partneri informišu o delovima internog procesa koji su relevantni za kolaboraciju. Minimum informacija koje javni proces treba da obezbedi partnerima koji su uključeni u kolaboraciju je u kojoj

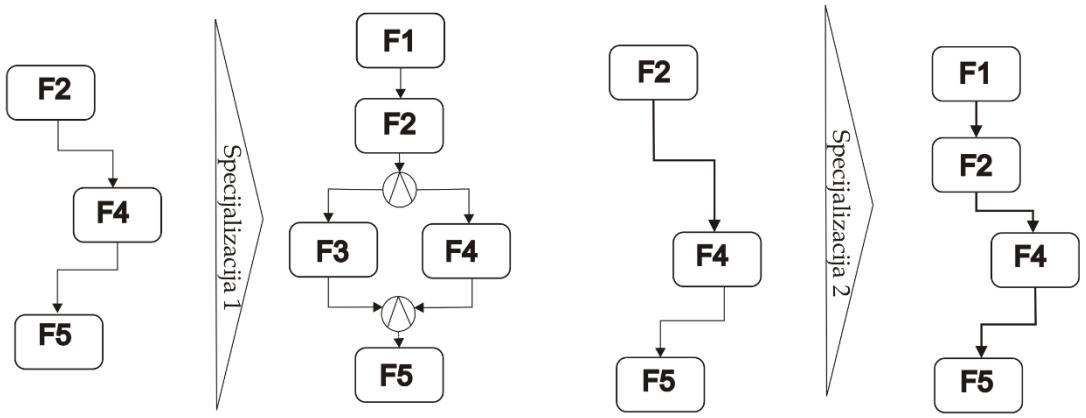
sekvenci i koji tip poruka je potrebno razmeniti. Kao što je već diskutovano, javni proces može da uključi one ne interakcione elemente koji su bitni za razumevanje kolaboracije. Prema tome, da bi se kreirao javni proces na osnovu privatnog procesa, potrebno je iz definicije privatnog procesa ukloniti one elemente koji nisu relevantni za kolaboraciju. Dve tehnike koje se za ovu svrhu predlažu u literaturi su: **apstrakcija** i **agregacija privatnog procesa**.

Apstrakcija modela privatnog procesa se vrši izostavljanjem odgovarajućih elemenata iz definicije procesa. Kao rezultat primene ove metode, apstrahovan proces ne sadrži sve informacije iz originalnog modela. Jedan model poslovnog procesa može da bude apstrahovan na različite načine (Ziemann, 2010). **Agregacija modela poslovnog procesa,** se vrši grupisanjem elemenata privatnog modela, koji se predstavljaju novim elementom u javnom modelu (slika 2.3).



Slika 2.3 Apstrakcija i agregacija poslovnog procesa. Preuzeto i modifikovano prema (Ziemann, 2010)

Pored apstrakcije i agregacije, kao treća operacija za skrivanje informacija privatnog poslovnog procesa se predlaže **generalizacija**. Generalizacija se slično apstrakciji i agregaciji, vrši nad individualnim elementima privatnog poslovnog procesa (slika 2.4).

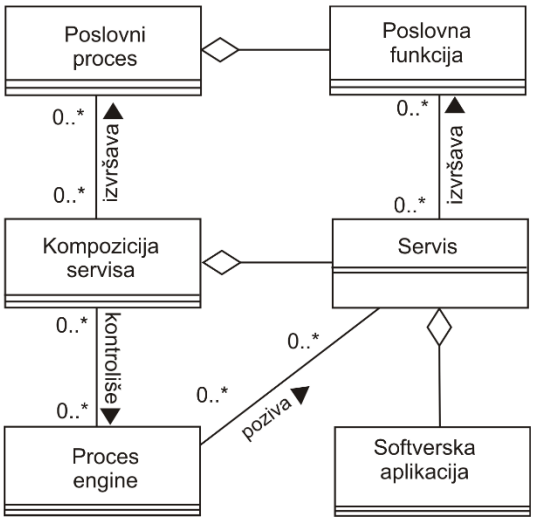


Slika 2.4 Dva privatna procesa kao specijalizacija jednog javnog poslovnog procesa. Preuzeto i modifikovano prema (Ziemann, 2010)

2.3.2. Aspekt servisa

U prethodnom poglavlju je opisan način definisanja i date su preporuke za modelovanje kolaborativnih poslovnih procesa. U ovom poglavlju će biti analiziran aspekt servisa, koji predstavlja prelazni koncept između koncepata poslovnog nivoa i nivoa izvršenja. Teza se ne bavi detaljima implementacije, već daje opšte preporuke i predlaže način da se specifikuje implementacija aspekata interoperabilnosti.

Ziemann definiše servis kao: "softversku komponentu koja implementira poslovnu funkciju, i opisana je pomoću razumljivog, mašinski-čitljivog interfejsa koji može da se pronađe i pozove na mreži"(Ziemann, 2010). Na slici 2.5 je ilustrovana veza između servisa, poslovnog procesa i izvršnog okruženja. Servisi su opisani pomoću poslovnih funkcija, koje su deo poslovnog procesa. Osnovna uloga servisa je da implementiraju poslovne funkcije. Servisi mogu da budu deo kompozicije servisa (eng. service composition), koja predstavlja realizaciju poslovnog procesa na tehničkom nivou. Kompozicija servisa može da se izvrši pomoću proces *engine*-a, odnosno opis kompozicije servisa kotroliše rad proces *engine*-a. Tokom izvršenja kompozicije servisa, vrši se poziv pojedinačnih servisa koji ulaze u njen sastav.



Slika 2.5 Veza između poslovnih procesa, servisa i aplikacija. Preuzeto i modifikovano prema (Ziemann, 2010)

Slika 2.5 ilusturuje relaciju servisa u odnosu na poslovne procese i na izvršno okruženje. Servisi implementiraju poslovne funkcije, pomoću kojih su opisani. Poslovne funkcije su deo poslovnog procesa, dok servisi mogu da budu deo kompozicije servisa. Tokom izvršenja kompozicije servisa, proces *engine* poziva pojedinačne servise koji ulaze u njen sastav. Sam servis može da bude realizovan pomoću različitih softverskih aplikacija.

2.3.3. Aspekt informacija

Interoperabilnost informacija se odnosi na mogućnost razmene podataka nezavisno od tehnologije implementacije softverskih komponenti, platforme i izvršnog okruženja. Dve bitne vrste interoperabilnosti podataka su sintaksna i semantička interoperabilnost. Usaglašavanje poslovnih podataka na sintaksnom i semantičkom nivou je neophodan uslov za uspešnu iteroperabilnost B2B poslovnih sistema (Aničić, 2006). *Sintaksna interoperabilnost* se realizuje uspostavljanjem i prihvaćanjem zajedničke forme za reprezentaciju podataka primenom jezika kao što su Extensible Markup Language (XML) ili Electronic Data Interchange (EDI). *Semantička interoperabilnost* se odnosi na sposobnost automatske ili polu-automatske, smisaone i tačne interpretacije značenja razmenjenih podataka. Realizacija semantičke interoperabilnosti zavisi od načina prikazivanja realnih poslovnih koncepata i konteksta u kojem se značenje podataka interpretira.

Kako bi se omogućilo prevazilaženje mogućih neusaglašenosti interpretacije razmenjenih podataka, često se predlaže uspostavljanje referentne ontologije (M. Vujasinovic et al., 2009; Marko Vujasinovic, Ivezic, Barkmeyer, & Marjanovic, 2010). Prema (M. Vujasinovic et al., 2009) problem interoperabilnosti B2B sistema se veže za neadekvatnu interoperabilnost na semantičkom nivou. U pristupu za specifikaciju aspekata interoperabilnosti koji se predlaže u disertaciji se za prevazilaženje problema semantičke interoperabilnosti koristi zajednička referentna ontologija. Gruber definiše referentnu ontologiju kao eksplicitnu i formalnu konceptualizaciju specifičnog poslovnog domena (Gruber, 1993). Referentna ontologija je prepoznata kao ključna osnova za realizaciju interoperabilnosti između B2B sistema koji učestvuju u kolaboraciji (Chandrasekaran, Josephson, & Benjamins, 1999).

2.3.4. Nefunkcionalni aspekti

Osnovna uloga nefunkcionalnih aspekata je da opišu nivo kvaliteta funkcionalnosti poslovnog sistema. Aspekt **sigurnosti** opisuje sposobnost rešenja da zaštiti resurse poslovnog sistema i obezbedi kontrolu pristupa. Aspekt sigurnosti uključuje: autentikaciju, autorizaciju i enkripciju podataka. Aspekt **skalabilnosti** se odnosi na mogućnost rešenja da se prilagodi povećanom obimu poslovnih zadataka. Aspekt **evolucije** ukazuje na mogućnost sistema da reaguje na promenu zahteva. Aspekt **performanse** se odnosi na mogućnost rešenja da brzo obavi poslovni zadatak, da obezbedi i vrati potrebne informacije na vreme. Aspekt **dostupnosti** se odnosi na vremenski interval u kojem je rešenje dostupno, na primer 7x24 (sedam dana u nedelji, dvadeset i četiri sata dnevno). Aspekt **portabilnosti** se odnosi na mogućnost korišćenja rešenja na različitim hardverskim platformama, operativnim sistemima i izvršnim okruženjima uz minimalne izmene. U nefunkcionalne aspekte sistema se ubraja i **interoperabilnost** poslovnih sistema.

2.4. Zahtevi za interoperabilnošću

Centralna tema disertacije je specifikacija aspekata interoperabilnosti u različitim metodološkim pristupima razvoju informacionih sistema. Za uspešnu specifikaciju

je bitno precizno definisati polazni skup zahteva za svaki od identifikovanih aspekata. U literaturi postoje brojne definicije i sistematizacije softverskih zahteva.

Zahtev može da se definiše kao „iskaz koji specificira funkciju, mogućnost ili karakteristiku koju proizvod ili sistem mora da zadovolji u određenom kontekstu“(Mallek, Daclin, & Chapurlat, 2011). Mallek i saradnici su predložili tri osnovne klase zahteva za interoperabilnošću: (1) kompatibilni zahtevi, (2) interoperativni zahtevi i (3) reverzibilni zahtevi (Mallek et al., 2011).

Kompatibilan zahtev se definiše kao iskaz koji specificira funkciju, mogućnost ili karakteristiku koju sistem mora da zadovolji (Mallek et al., 2011). Autori u (Mallek et al., 2011) ističu da kompatibilni zahtevi nisu zavisni od vremena i odnose se na barijere interoperabilnosti (konceptualne, tehnološke i organizacione) za svaki od nivoa interoperabilnosti (podataka, servisa, procesa i poslovni). Osnovni cilj definisanja kompatibilnih zahteva je da se obezbedi harmonizacija između partnera (organizacija, metodi, alati, itd.) kako bi bila moguća efikasna kolaboracija. Ipak, kompatibilnost se odnosi na statički pogled na sistem i predstavlja potreban, ali ne i dovoljan uslov za formulaciju inteoperabilnih zahteva. Činjenica je da se tokom kolaboracije između dva partnera javljaju brojni drugi problemi interoperabilnosti. Na primer, dva poslovna sistema mogu da koriste isti jezik za reprezentaciju podataka, ali jedan od njih nije u stanju da pošalje podatke. U ovom slučaju, „dva interoperabilna sistema su kompatibilna, ali dva kompatibilna sistema nisu obavezno interoperabilna“(Mallek et al., 2011). Ovaj iskaz, opravdava razlog za uvođenje nove kategorije interoperabilnih zahteva pod nazivom *interoperativni* zahtevi.

Interoperativan zahtev se definiše kao „iskaz koji specificira funkciju, mogućnost ili karakteristiku, koja je zavisna od vremena a odnosi se na performansu interakcije, koju sistem mora da zadovolji“(Mallek et al., 2011). Ovi zahtevi su fokusirani na mogućnost poslovnog sistema da prilagode organizaciju, način funkcionisanja i ponašanje prilikom interakcije.

Reverzibilan zahtev se definiše kao „iskaz koji specificira funkciju, mogućnost ili karakteristiku koja se odnosi na sposobnost poslovnog sistema da povrati svoju

autonomiju i da se vrati u početno stanje po završetku kolaboracije“(Mallek et al., 2011). Reverzibilnost se odnosi na sposobnost poslovnog sistema da sa lakoćom održava ili povrati svoju autonomiju i performanse (bilo pozitivne ili negativne) po završetku kolaboracije.

Statičke zahteve za interoperabilnošću, koji ne zavise od vremena, je potrebno verifikovati tokom čitavog životnog veka kolaboracije. Sa druge strane, dinamički zahtevi koji zavise od vremenske dimenzije, treba da budu verifikovani samo u odgovarajućim fazama kolaboracije. Prema tome, kompatibilni zahtevi su statički, interopertivni zahtevi su dinamički, dok reverzibilni zahtevi mogu da imaju oba aspekta. Opis, formalizacija i razumevanje zahteva može da bude otežano iz više razloga: kompleksnost,veliki broj zahteva, zahtevani nivo granularnosti itd.

Sommerville i Sawyer su dali sledeću definiciju zahteva: “zahtevi specifikuju šta treba da bude implementirano. Oni opisuju željeno ponašanje sistema, ili potrebno svojstvo. Oni mogu da definišu ograničenja koja su bitna za proces razvoja sistema” (Sommerville & Sawyer, 1997). Ova definicija je interesantna, jer ukazuje na različitost tipova informacija koje se zajedničkim imenom označavaju kao “zahtevi”. Zahtevi obuhvataju korisnički pogled na potrebno spoljno ponašanje sistema, kao i tehničke karakteristike sistema bitne sa aspekata programera.

Prema (Wiegers & Beatty, 2013), softverski zahtevi obuhvataju tri različita nivoa: poslovni zahtevi, korisnički zahtevi i funkcionalni zahtevi. Pored ova tri osnovna tipa zahteva, svaki sistem ima skup nefunkcionalnih zahteva. *Poslovni zahtevi* opisuju razlog zbog kojeg organizacija implementira informacioni sistem (“*zašto*”). Fokus je na opisu očekivanih pogodnosti koje bi implementacija sistema trebalo da obezbedi. *Korisnički zahtevi* opisuju ciljeve ili zadatke koje korisnik očekuje od informacionog sistema. Korisnički zahtevi obično uključuju i opis karakteristika i atributa proizvoda koji su bitni za zadovoljstvo krajnjih korisnika. Neki od načina za reprezentaciju korisničkih zahteva su: slučajevi korišćenja (Kulak & Guiney, 2003), korisničke priče (Cohn, 2004), i događaj-odgovor (eng. event-response) tabele. Korisnički zahtevi opisuju šta će korisnik biti u mogućnosti da uradi pomoću sistema. *Funkcionalni zahtevi* specificiraju *šta* je potrebno da se implementira kako

bi se realizovali korisnički zahtevi, a u skladu sa poslovnim zahtevima. Usaglašenost tri osnovna tipa zahteva je ključna za uspeh projekta. Nefunkcionalni zahtevi ne opisuju šta sistem radi, već koliko dobro vrši svoju funkcionalnost. Neki od autora smatraju da su nefunkcionalni zahtevi sinonim za *atribute kvaliteta*, što je restriktivno (Bass, Clements, & Kazman, 2012; Rozanski & Woods, 2011). *Nefunkcionalni zahtevi* obuhvataju bitne karakteristike sistema kao što su: performansa izvršenja, sigurnost, portabilnost, interoperabilnost, itd. Različita ograničenja dizajna i implementacije takođe spadaju u nefunkcionalne zahteve. Bass i saradnici predlažu klasifikaciju na *eksterne* i *interne* atribute kvaliteta (Bass et al., 2012). *Eksterni atributi kvaliteta* opisuju karakteristike softverskog sistema koje mogu da se prate tokom izvršenja softvera. Eksterni atributi su bitni za krajnje korisnike i doprinose njihovom većem zadovoljstvu proizvodom. *Interni atributi* su bitni u fazi razvoja i održavanja softverskog rešenja i odnose se na lakše održavanje, korigovanje, unapeđenje i migraciju na nove platforme.

Prema Wiegers-u, *interoperabilnost* kao interni atribut kvaliteta, ukazuje na spremnost sistema da razmeni podatke i servise sa drugim softverskim sistemom, kao i koliko jednostavno može da se intergriše sa eksternim hardverskim uređajima (Wiegers & Beatty, 2013). Autori su definisali listu pitanja koja ima za cilj da olakša identifikaciju zahteva za interoperabilnošću. Za problematiku kojom se bavi disertacija relevantna su sledeća pitanja: (1) „Sa kojim sistemima naš sistem stupa u interakciju?“, (2) „Koje servise i podatke je potrebno razmeniti?“ i (3) „Koji standardni format podataka je potrebno koristiti kako bi se obezbedila razmena sa drugim sistemima?“.

3. METODOLOŠKI PRISTUPI ZA RAZVOJ INFORMACIONIH SISTEMA

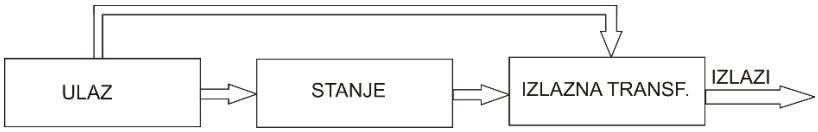
U ovom poglavlju su dati osnovni pojmovi i definicije koji su bitni za razumevanje različitih metodoloških pristupa za razvoj informacionih sistema. Posebna pažnja je posvećena objašnjenju principa sistemsko-teorijskog modela životnog ciklusa informacionog sistema, koji je odabran kao teorijski okvir predloženog pristupa za specifikaciju aspekata interoperabilnosti. Takođe, u ovom poglavlju je dat opšti pregled metodologija za razvoj informacionih sistema i ukazano je na značaj relevantne podrške za specifikaciju interoperabilnosti. Zatim su prikazane osnovne faze i osobine FON Labis metodologije za razvoj informacionih sistema i Larmanove metode koje su odabrane za validaciju predloženog pristupa specifikacije aspekata interoperabilnosti.

3.1. Osnovni pojmovi

Standardna metodologija razvoja informacionog sistema (IS) formalizuje proces kojim se realizuje neki IS. Prema (PMOV, 2000), savlađivanje složenosti procesa razvoja IS se postiže: (1) precizno definisanim redosledom faza za realizaciju IS i (2) specifičnim modelima i alatima koji se za svaku od faza koriste. Pri kreiranju metodologije za razvoj informacionih sistema, treba imati u vidu da je njena poželjna karakteristika opštost, odnosno mogućnost primene na sisteme bilo koje vrste (PMOV, 2000).

Prema (PMOV, 2000), **sistem** se najopštije definiše kao skup objekata i njihovih međusobnih veza, gde objekti u sistemu mogu da budu: fizički objekti, koncepti, događaji i drugo. Autori navode da se dejstvo okoline na sistem opisuje preko ulaza u sistem, a dejstvo sistema na okolinu preko njegovih izlaza (PMOV, 2000). Jedan od načina za prikazivanje dinamike i ponašanje realnog sistema je ilustrovan na slici 3.1. Prema (PMOV, 2000), „stanje sistema u jednom trenutku vremena predstavlja skup objekata sistema, njihove međusobne veze i vrednosti atributa objekata“.

Autori definišu stanje sistema kao skup informacija o prošlosti i sadašnjosti sistema, koje opisuju njegove fundamentalne karakteristike Izlazna transformacija na osnovu ulaza koji menjaju stanje sistema, definiše njegove izlaze (PMOV, 2000).



Slika 3.1 Realni sistem. Preuzeto i modifikovno prema (PMOV, 2000)

Imajući u vidu navedene karakteristike realnog sistema, autori ukazuju na činjenicu da **informacioni sistem** može da se definiše kao model realnog sistema u kome deluje. Oni ističu da osnovu svakog informacionog sistema predstavlja njegova baza podataka. Baza podataka može da se definiše kao "kolekcija međusobno povezanih podataka koja modelira objekte, veze objekata i atribute objekata posmatranog realnog sistema" (PMOV, 2000). Ako je baza podataka dobro projektovana, onda se iz IS može dobiti bilo koja informacija potrebna za upravljanje (izlaz).

Postupak projektovanja IS se svodi na modelovanje realnog sistema, a kao intelektualna sredstva se predlažu model podataka i model procesa (PMOV, 2000). Prema (PMOV, 2000) **model podataka** predstavlja: „intelektualno sredstvo za opis statičkih karakteristika sistema, pomoću kojeg se prikazuju objekti sistema, njihovi atributi i međusobne veze". Svaki model podataka treba da ima definisane tri osnovne komponente: strukturu modela, semantička ograničenja na vrednost podataka i operacije nad konceptima strukture (Lazarević, Marjanović, Aničić, & Babarogić, 2006; PMOV, 2000). Dinamika realnog sistema se opisuje pomoću **modela procesa**. Model porocesa se definiše kao: „intelektualno sredstvo za opis dejstva ulaza na stanje sistema i izlazne transformacije, preko programa nad definisanim modelom podataka" (Lazarević et al., 2006; PMOV, 2000).

Prema (PMOV, 2000), svaki model podataka treba da zadovolji dva bitna kriterijuma: (1) da poseduje koncepte koji su bitni za modelovanje realnih sistema i (2) da su njegovi koncepti, struktura, ograničenja i operacije jednostavni za implementaciju. Imajući u vidu ova dva bitna kriterijuma, autori predlažu klasifikaciju modela podataka na tri generacije (PMOV, 2000). *Prvu generaciju* čine

konvencionalni programski jezici, poznati kao jezici treće generacije koji se mogu tretirati kao modeli podataka (PMOV, 2000). Autori ističu da apstrakciju podataka u njima predstavljaju tipovi podataka kojima raspolažu, i da su jedine agregacije koje se koriste rekordi i vektori. Autori u (PMOV, 2000), ukazuju na to da ovi jezici nisu pogodni za modelovanje realnog sistema usled nemogućnosti definisanja ograničenja nad vrednostima pojedinih tipova i činjenice da se sve operacije moraju realizovati implementacijom odgovarajućih procedura. Prema (PMOV, 2000), *drugu generaciju* čine hijerarhijski, mrežni i relacioni modeli baze podataka. Iako semantički bogatiji od jezika prve generacije, autori ističu da ovi modeli nisu dovoljno pogodni za opis složenih objekata koji se ne mogu lako predstaviti konceptom rekorda. *Treću generaciju* čine semantički bogati modeli podataka i objektni modeli podataka koji podržavaju (PMOV, 2000): „različite vrste apstrakcija, poseduju mogućnost definisanja ograničenja na vrednost podataka i moćne operacije nad objektima visokog nivoa apstrakcije“. U praksi se često koriste: Prošireni model objekti i veze (PMOV), Model entiteti-veze, Semantic Data Model (SDM), Semantičke mreže i različiti objektno-orijentisani modeli podataka.

Avison i Fitzgerald, definišu **metodologiju za razvoj informacionih sistema** kao: „kolekciju procedura, tehnika, alata, i dokumentacije koja treba da pomogne sistemskim programerima da implementiraju novi informacioni sistem“ (Avison & Fitzgerald, 2006). Metodologija ima više faza, a svaka faza se sastoji od odgovarajućeg skupa podfaza. Za svaku fazu, odnosno podfazu metodološkog pristupa se predlaže odgovarajući skup tehnika. Autori naglašavaju da se određena tehnika može koristiti u više različitih metodoloških pristupa (Avison & Fitzgerald, 2006).

Različiti metodološki pristupi kombinuju postojeći skup tehnika na specifičan način. Postavlja se pitanje da li je moguće definisati uniformni metodološki pristup za sve faze životnog ciklusa softvera? Ovo pitanje je detaljno diskutovano u (Lazarević & Nešković, 1997), gde autori predlažu primenu opšteg **sistemsko-teorijskog pristupa** u razvoju softvera. Opšti sistemsko-teorijski pristup se bazira na ideji da se u razvoju softvera mogu definisati tri osnovne faze, koje su prisutne u većini metodoloških pristupa za razvoj informacionih sistema. Budući da je sistemsko-

teorijski pristup izabran za osnovu analize aspekata interoperabilnosti u ovoj tezi, biće detaljnije prodiskutovan u ostatku ovog poglavlja.

3.2. Sistemsko-teorijski model životnog ciklusa IS

Teorijski okvir pristupa za specifikaciju aspekata interoperabilnosti koji se predlaže u tezi je **sistemsko-teorijski model životnog ciklusa informacionog sistema** koji sadrži sledeće tri osnovne faze (Lazarević & Nešković, 1997):

- **Identifikacija sistema**, koja podrazumeva opisivanje sistema kao skupa funkcija koje transformišu skup ulaza u skup izlaza, odnosno definisanje funkcionalnog modela softvera preko njegovih stvarnih ulaza i izlaza;

- **Realizacija sistema**, kojom se definiše specifičan pogled na realni sistem preko izabranog modela (teorije) kojim se prikazuje ponašanje sistema u prostoru stanja. Ovde se vrši detaljnije modeliranje sistema u izabranom prostoru stanja, i po mogućstvu nalaženje njegove minimalne realizacije. Pod minimalnom realizacijom se podrazumeva nalaženje minimalnog skupa informacija o prošlosti i sadašnjosti sistema, koji je dovoljan za definisanje budućeg stanja sistema.

- **Implementacija sistema**, kojom se definiše način implementacije modela sistema u izabranom tehnološkom okruženju.

U svakom od navedenih koraka daje se specifikacija koja predstavlja opis sistema na različitim nivoima apstrakcije. Očigledno je da **realizacija sistema** predstavlja jedan detaljniji opis u odnosu na **identifikaciju sistema** dok se **implementacija sistema** može posmatrati kao dalji detaljni opis u izabranom tehnološkom okruženju (slika 3.2)(Labis, 2015).

Slika 3.2 Sistemsko-teorijski model životnog ciklusa

Za identifikaciju sistema može se koristi bilo koji model koji omogućava funkcionalnu specifikaciju sistema, dok se za realizaciju sistema bira model koji je najpogodniji za datu vrstu sistema i dati nivo apstrakcije. Stoga se, na različitim apstraktnim nivoima, mogu birati različiti modeli iz različitih metodologija. U sistemsko-teorijskom okviru je pokazano da metodološka uniformnost u razvoju softvera nema opravdanja (Labis, 2015). Proces razvoja IS se sada može formulisati kao postupak nalaženja pomenute tri apstraktne specifikacije, odnosno kao postupak kojim se od *identifikacije* sistema stiže do *implementacije* sistema. Postupak detaljnijeg opisa sistema na nižem apstraktnom nivou se ponavlja sve dok se ne dođe do nivoa čija je realizacija jasna u nekom tehnološkom okruženju.

3.2.1. Identifikacija sistema

U fazi *identifikacije sistema* se na osnovu korisničkih zahteva vrši specifikacija informacionog sistema. Osnovni cilj faze identifikacije je da se izgradi poslovni model sistema (eng. business model), koji podrazumeva definisanje modela funkcija. Za analizu sistema u fazi identifikacije može da se koristiti bilo koja konvencionalna ili objektna metoda za funkcionalnu specifikaciju sistema. Neke od popularnih konvencionalnih metoda za funkcionalnu specifikaciju sistema su: Strukturna sistemska analiza (SSA), Structured Analysis and Design Technique (SADT) i Integration Definition for Function Modeling (IDEF0)(Avison & Fitzgerald, 2006; Isaias & Issa, 2015). Od objektno-orijentisanih metoda za analizu i funkcionalnu specifikaciju sistema posebno se izdvaja UML Model slučajeva korišćenja (Larman, 2004; J. Martin & Odell, 1997).

Rezultat faze identifikacije sistema je funkcionalni model sistema. Zbog kompleksnosti informacionih sistema, čiji objektno-orijentisani model može da sadrži i nekoliko hiljada različitih objekata, preporuka je da prvi modeli u razvoju budu funkcionalni (Lazarević & Nešković, 1997). Autori ističu da je funkcionalna dekompozicija sistema kao pristup specifikaciji sistema pogodna tehnika za komunikaciju sa korisnicima koji su navikli na funkcionalni pogled na sistem i bliski su im pojmovi kao što su: funkcija sistema, proces, itd. (Lazarević & Nešković, 1997).

Na najvišem nivou apstrakcije, funkcionalni model sistema predstavlja sistem kao „crnu kutiju“ (slika 3.3.). Drugim rečima, celokupan informacioni sistem se predstavlja kao funkcija koja skup ulaza transformiše u skup izlaza, za čije predstavljanje je pogodan dijagram konteksta SSA. Bitno je naglasiti da se funkcionalnost sistema predstavlja na način na koji je vide spoljni objekti. Funkcionalni model sistema predstavlja i model zahteva, jer je cilj da pokaže potpuno, precizno i nedvosmisleno kako će objekti van sistema (korisnici, akteri) koristiti posmatrani sistem.

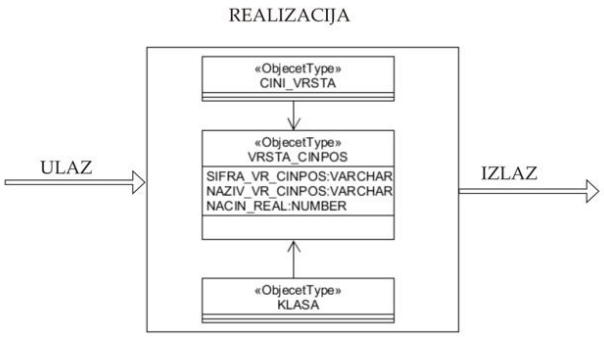
Slika 3.3 Identifikacija sistema

Bez obzira na izbor tehnike za funkcionalnu specifikaciju sistema, cilj je da se identifikuju esencijalni procesi. Suštinski, esencijalni procesi odražavaju svrhu posmatranog sistema i ne zavise od konkretne tehnologije u kojoj se vrši implementacija (SSA, 2000). Razlikujemo dve vrste esencijalnih procesa: (1) *osnovne ili fundamentalne* procese koji opravdavaju postojanje sistema, odnosno generišu izlazne tokove ka okruženju i (2) *procese održavanja* koji prihvataju i skladište podatke od fundamentalnih procesa ili iz okruženja (SSA, 2000). Autori u (SSA, 2000) ističu da se u praksi obično javljaju složeni procesi koji objedinjuju ove dve vrste esencijalnih procesa. Nalaženje logičkog modela IS može da se ostvari (SSA, 2000): (1) **direktnim modelovanjem** na osnovu poznavanja suštinskih procesa realnih sistema; (2) **snimanjem,** odnosno izvlačenjem logičkog iz postojećeg fizičkog modela. Autori navode da je sa praktične tačke gledišta teško definisati striktnu granicu između ova dva pristupa, pa se obično preporučuje njihova kombinacija. Navedeni postupci su detaljno objašnjeni u skripti (SSA, 2000) i u tezi neće biti dalje diskutovani.

3.2.2. Realizacija sistema

Pod realizacijom sistema se podrazumeva “detaljnije modeliranje sistema u nekom izabranom prostoru stanja i po mogućstvu nalaženje neke njegove minimalne

realizacije" (slika 3.4)(Lazarević & Nešković, 1997). Autori smatraju da je u ovoj fazi neophodno: (1) definisati model za realizaciju sistema; (2) izgraditi neku zadovoljavajuću ili minimalnu realizaciju sistema u izabranom modelu.



Slika 3.4 Realizacija sistema

Prema (Lazarević & Nešković, 1997) mogu da se definišu dve opšte vrste modela za realizaciju sistema: (1) modeli podataka i (2) objektni modeli. Pod minimalnom realizacijom sistema autori podrazumevaju nalaženje minimalnog skupa informacija o prošlosti i sadašnjosti sistema koji je dovoljan da bi se definisalo buduće ponašanje sistema (budući izlazi) na osnovu budućih poznatih ulaza. Realizaciji sistema se može pristupiti na dva načina (Lazarević & Nešković, 1997):

- Direktnim modeliranjem na osnovu poznavanja realnog sistema, tj. direktnim opisivanjem realnog sistema u definisanom modelu;
- Integracija podmodela, gde se pod modelom podrazumeva model stanja sistema dobijen iz strukture podataka jednog ulaza ili izlaza sistema.

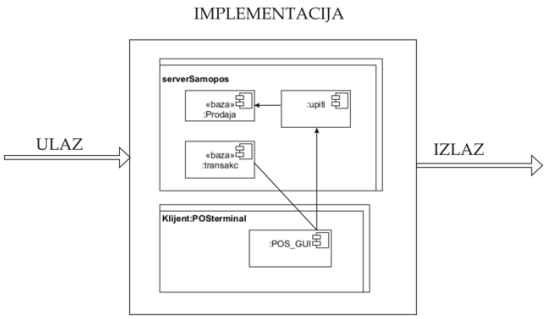
Autori ističu da se u praksi pokazalo da je najbolji rezultat zagarantovan kombinacijom ova dva pristupa. Ako se za realizaciju sistema koriste modeli podataka postupci izgradnje sistema i minimalne realizacije su poznati (Lazarević & Nešković, 1997). Na primer, u relacionom modelu to je postupak normalizacije relacija, dok u modelu objekti-veze postoje definisani standardi za formiranje podmodela i njihovu integraciju (Lazarević & Nešković, 1997). Autori u (Lazarević & Nešković, 1997), ukazuju na to da u objektno-orijentisanim pristupima sa tačke realizacije sistema nije precizirano kako se sistem dekomponuje na objekte. Oni ističu da se u izgradnji objektnog modela preporučuje analiza verbalnog opisa

sistema i gramatičkih konstrukcija u tom verbalnom opisu, na osnovu kojih se definišu objekti, njihove veze, atributi i relacije.

U fazi realizacije sistema se definišu dva osnovna modela: model strukture i model dinamike. Za prikaz strukture sistema u fazi realizacije je moguće koristiti neki od semantički bogatih modela podataka kao što je PMOV ili UML Dijagram klasa, odnosno Objekata. Za opis dinamike sistema u fazi realizacije se preporučuje upotreba dijagrama interakcije, promene stanja ili dijagrama aktivnosti (Lazarević et al., 2006).

3.2.3. Implementacija sistema

Pod implementacijom sistema se podrazumeva prevođenje modela definisanih u fazi realizacije sistema u odnosu na izabrano tehnološko okruženje (slika 3.5.)(Lazarević & Nešković, 1997). Ova faza se zasniva na Model Driven Development (MDD) pristupu i transformacijama modela. U MDD pristupu sistem se opisuje na različitim apstraktnim nivoima pomoću tri vrste modela: Računarski nezavisni modeli (Computer Independent Models-CIM), Platformski nezavisni modeli (Platform Independent Models-PIM) i Platformski zavisni modeli (Platform Specific Models-PSM).



Slika 3.5 Imlpementacija sistema

3.3. Pregled metodologija za razvoj informacionih sistema

U literaturi postoji veliki broj specifičnih metodoloških pristupa koji su se javili tokom razvoja informacionih sistema kao naučne discipline. Postoje brojne klasifikacije metodoloških pristupa koje se predlažu od strane različitih autora i

međunarodnih tela za standardizaciju (ATHENA A1.1, 2004; Avison & Fitzgerald, 2006; Arne-Jorgen Berre et al., 2004; Isaias & Issa, 2015). Međutim, kao tri osnovna pristupa se izdvajaju: (1) strukturni (funkcionalni), (2) objektno-orijentisani i (3) agilni pristupi. Ostali specifični i manje korišćeni pristupi su često izvedeni iz ova tri osnovna pristupa. Zbog toga ćemo u ovom poglavlju predstaviti osnovne karakteristike ova tri pristupa.

3.3.1. Strukturni pristupi

Za strukturne metodološke pristupe razvoju informacionih sistema je karakteristično da se zasnivaju na funkcionalnoj dekompoziciji sistema. Naime, funkcionalna dekompozicija se koristi kao osnovni metodološki princip savladavanja složenosti sistema. Sistem se posmatra kao skup međusobno povezanih funkcija koje transformišu ulaze u izlaze, pri čemu se funkcija sa višeg nivoa apstrakcije dekomponuje na funkcije na nižem nivou (Lazarević et al., 2006; SSA, 2000). Strukturni metodološki pristupi su u početku bili bazirani na tzv. „vodopad“ modelu životnog ciklusa, koji je postepeno težio ka „transformacionom“ modelu (Lazarević & Nešković, 1999). U strukturnom pristupu funkcionalna dekopozicija se koristi kao osnovni princip ne samo u početnim fazama analize i specifikacije sistema, već i kasnije za definisanje strukture programa (Nešković, 2006; SSA, 2000). Prema (Nešković, 2006), ovakva metodološka uniformnost nije opravdana i za direktnu posledicu ima pojavu objektno-orijentisanog pristupa.

Najpoznatije u literaturi i u praksi najčešće korišćene strukturne metodologije su: Structured Systems Analysis (SSA), Information System Work and Analysis of Change (ISAC), Structured Analysis and Design Techniques (SADT), Structured Analysis and Design Method (SSADM), Information Engineering (IE) i Rapid Application Development (RAD) (Avison & Fitzgerald, 2006; Isaias & Issa, 2015; Pfleeger & Atlee, 2006).

3.3.2. Objektno-orijentisani pristupi

U literaturi postoje brojni objektno-orijentisani pristupi, čija je zajednička karakteristika da za savladavanje složenosti sistema koriste objektnu

dekompoziciju (Blaha & Rumbaugh, 2004; Booch et al., 2007; Coad & Yourdon, 1990; Jacobson, Booch, & Rumbaugh, 1999a; J. Martin & Odell, 1997; Yourdon & Argila, 1996).

U objektno-orijentisanim pristupima sistem se posmatra kao kolekcija međusobno povezanih objekata (Jacobson et al., 1999a; Lazarević et al., 2006). Pod pojmom objekta se podrazumeva bilo koji fizički objekat ili koncept iz realnog sveta koji se predstavlja kao apstraktni tip podataka (Booch et al., 2007). Prema (Blaha & Rumbaugh, 2004), objekat je entitet koji poseduje skup stanja i skup operacija preko kojih ta stanja mogu da se prikazuju i menjaju. Kao osnovni nedostatak objektno-orijentisanih pristupa se navodi ukidanje funkcionalne dekompozicije kao mehanizma za savladavanje složenosti sistema (Nešković, 2006). Ovaj nedostatak dolazi do izražaja naročito u fazi analize sistema, pri specifikaciji zahteva. Funkcionalna dekompozicija sistema je daleko prirodnija i prikladnija za komunikaciju sa korisnicima i identifikaciju zahteva.

Za objektno-orijentisane pristupe je karakterističan iterativno-inkrementalni pristup razvoju informacionih sistema. Iterativno-inkrementalni životni ciklus se sastoji od više iteracija, pri čemu svaka iteracija predstavlja jedan relativno mali, po pravilu korisniku značajan deo projekta (inkrement) (Lazarević & Nešković, 1997; J. Martin & Odell, 1997).

U literaturi i praksi često korišćene objektne metodologije su: Object Modeling Technique (OMT), Object Oriented Analysis and Design (OOAD), Object Oriented Information Engineering (OOIE), Object Oriented Analysis and Design (OOADA), Object Oriented Software Engineering (OOSE), Object Oriented System Analysis (OSA), Semantic Object Modeling Approach (SOMA), Object Oriented Analysis/Object Oriented Design (OOA/OOD), Larmanova metoda životnog ciklusa softvera i Rational Unified Process (RUP) (Avison & Fitzgerald, 2006; Pfleeger & Atlee, 2006) .

3.3.3. Agilni pristupi

Agilni metodološki pristupi su nastali kao posledica sve većih zahteva za neprekidnim inovacijama u uslovima opšte globalizacije koja je zahvatila sve industrijske grane. Prema (Highsmith & Cockburn, 2001), agilnost se odnosi na mogućnost poslovnog sistema da reaguje na promene sa ciljem ostvarenja dobiti u turbulentnom okruženju. Agilne metode su zasnovane na iterativnom pristupu i glavni fokus im je na brzom razvoju softverskog proizvoda uz postojani kvalitet (Cockburn, 2006; Cohn, 2004; R. C. Martin, 2002). Agilni manifesto definiše niz principa koji su karakteristični za agilne pristupe (Beck et al., 2001). Agilni pristupi su posebno pogodni za projekte koji zahtevaju kratke razvojne cikluse, brzu isporuku gotovog softverskog rešenja i blisku saradnju naručioca i izvođača. U poznate agilne metodologije spadaju: eXtreme Programing, Adaptive Software Development (ASD), Scrum Object Technology, Lean Development i Cristal (Pfleeger & Atlee, 2006).

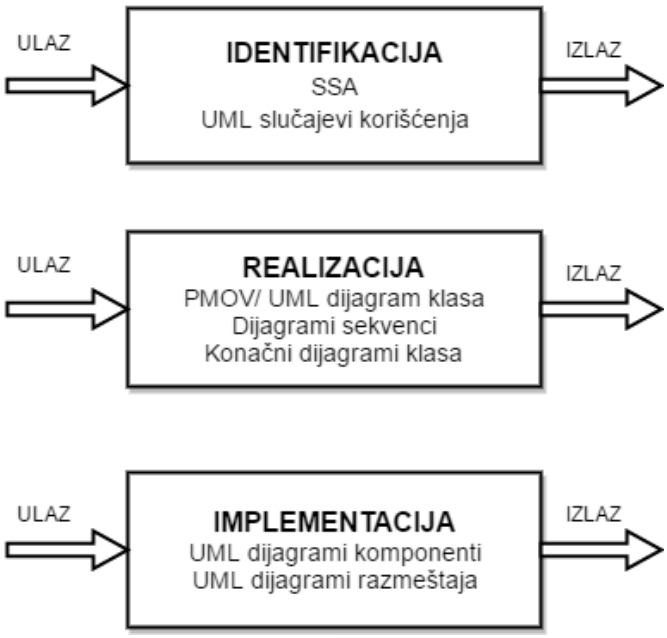
3.4. FON Labis metodologija za projektovanje informacionih sistema

Fonova metodologija razvoja informacionih sistema, koja se bazira na objektno-orijentisanom i strukturnom pristupu definiše način integracije ova dva pristupa (Labis, 2015). Fon Labis metodologija razvoja informacionih sistema je nastala 90-tih godina prošlog veka na Fakultetu organizacionih nauka na osnovu višegodišnjeg iskustva iz oblasti razvoja informacionih sistema. Razvili su je članovi Laboratorije za informacione sisteme na čelu sa Prof. dr Branislavom Lazarevićem. Fon Labis se može definisati kao objektno-orijentisana metodologija skladno uklopljenih objektnih metoda i tehnika. Karakteriše je iterativno-inkrementalni razvoj sa kraćim razvojnim ciklusima kojima je omogućen efikasan odgovor na stalne promene. Fon Labis model životnog ciklusa je baziran na sistemsko-teorijskom modelu životnog ciklusa, ali je bitno unapređen jer definiše postupak postepenog razvoja sistema i njegove postepene implementacije (Labis, 2015). FON Labis metodologija ima tri osnovne faze koje su definisane u skladu sa fazama sistemsko-teorijskog životnog ciklusa: identifikaciju, realizaciju i implementaciju sistema.

U **fazi identifikacije** sistema *analiza zahteva* se ostvaruje metodom SSA, čiji se rezultati zapisuju korišćenjem tehnike Dijagrama tokova podataka (DTP). Specifična primena metode SSA se ogleda u precizno definisanom kriterijumu za dekompoziciju sistema. Naime, procesi se dekomponuju sve dotle dok između sebe komuniciraju isključivo preko skladišta podataka (SSA, 2000). Rezultat faze identifikacije je model funkcija poslovnog sistema. Model sistema se savladava postupno, tako što se posmatra jedna po jedna funkcija. Detaljna analiza modela funkcija podrazumeva izradu pojedinačnih slučajeva korišćenja za svaku od funkcija. U fazi analize sistema i specifikacije zahteva se daje struktuirani verbalni opis, dok se u fazi realizacije ovaj opis dopunjava formalnim opisom kolaboracije objekata preko kojih se posmatrani slučaj korišćenja realizuje (Lazarević et al., 2006).

U **fazi realizacije**, koja podrazumeva detaljnije modelovanje sistema, prvo se definiše *model podataka*. Za prikaz **strukture sistema** se predlaže upotreba semantički bogatog modela kao što je PMOV, koji omogućava integralni opis i strukture i ponašanja realnog sistema. Pored PMOV-a moguće je koristiti i različite objektno-orijentisane modele podataka, kao što je UML dijagram klasa. Poštujući princip minimalne realizacije sistema, prvo se definišu osnovni objekti sistema, njihovi atributi i međusobne veze. Za opis **dinamike sistema** u fazi relizacije se preporučuje upotreba *dijagrama sekvenci*, koji se kreiraju u skladu sa pravilima logičke arhitekture koju je potrebno prethodno definisati. Na osnovu dijagrama sekvenci se kreiraju *konačni dijagrami klasa*, koji pored atributa sadrže i specifikaciju operacija.

U **fazi implementacije** se kreiraju dijagrami komponenti i dijagrami rasporeda u višeslojnim arhitekturama poslovnih objekata. Osnovne faze FON Labis metodologije, kao i tehnike koje se preporučuju za realizaciju opisanih koraka su ilustrovane na slici 3.6.



Slika 3.6 Faze i koraci FON Labis metodologije

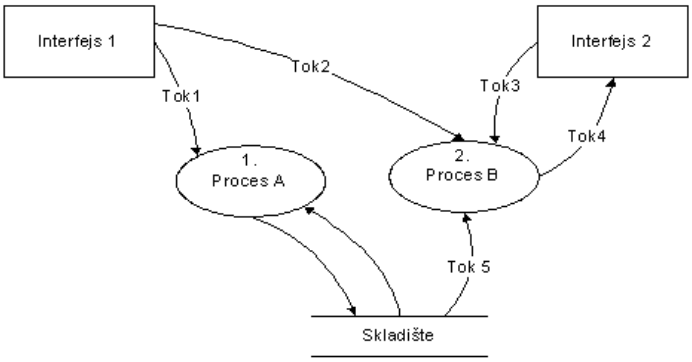
3.4.1. Izrada modela procesa

3.4.1.1. SSA-Strukturna sistemska analiza

Za specifikaciju sistema u prvom koraku faze *identifikacije* se primenjuje metoda *Strukturne sistemske analize*. SSA je "jedna potpuna, samosvojna metoda za analizu i funkcionalnu specifikaciju informacionog sistema" (Lazarević et al., 2006). FON Labis metodologija je jedan od primera povezivanja metode SSA sa metodama drugih faza u specifičnu metodologiju celokupnog razvoja IS. Kao bitan nedostatak objektno-orijentisanih pristupa se ističe činjenica da ne postoji postupak analogan SSA, kojim bi se došlo do skupa funkcija kao modela sistema (Lazarević et al., 2006).

Prema (Lazarević et al., 2006), „SSA posmatra informacioni sistem kao funkciju (proces obrade) koja na bazi ulaznih podataka i stanja sistema predstavljenog preko skladišta podataka generiše izlazne podatke." Lazarević i saradnici napominju da se ulazni podaci dovode u proces obrade, a izlazni iz njega odvode preko tokova podataka. Isto tako, proces obrade koristi i menja podatke o stanju sistema razmenom tokova podataka sa skladištem. Imajući u vidu zahtev da specifikacija treba da se oslobodi svih implementacionih detalja, autori napominju da su od interesa samo sadržaj i struktura toka, a ne i medijum nosilac toka" (SSA, 2000).

Osnovni koncepti za specifikaciju IS u SSA su funkcije, odnosno procesi obrade podataka, tokovi podataka, skladišta podataka i interfejsi. Njihov međusobni odnos se prikazuje preko **Dijagrama toka podataka (DTP)**. Na Slici 3.7 je prikazan opšti DTP koji predstavlja osnovne grafičke simbole SSA metode: (1) krug ili elipsa reprezentuje proces obrade podataka, odnosno funkciju (2) pravougaonik predstavlja interfejs, (3) usmerena linija označava tok podataka i (4) dve paralelne linije ilustruju skladište podataka (Lazarević et al., 2006).



Slika 3.7 Osnovni koncepti Dijagrama toka podataka (Lazarević et al., 2006)

Dijagram toka podataka na vrhu hijerarhije, koji obično sadrži samo jednu funkciju, se naziva dijagram konteksta, a funkcije na najnižem nivou (procesi koji se dalje ne dekomponuju) se nazivaju primitivne funkcije (Lazarević et al., 2006; SSA, 2000). Kao što je već napomenuto, strukturna sistemska analiza je donekle modifikovana, a izmena se odnosi na precizno definisanje kriterijuma za dekompoziciju sistema.

Jednu potpunu specifikaciju IS, u metodi Strukturne sistemske analize, čine (Lazarević et al., 2006): (1) hijerarhijski organizovan skup dijagrama toka podataka, (2) rečnik podataka koji opisuje sadržaj i strukturu svih tokova i skladišta podataka i (3) specifikacija logike primitivnih procesa. Pravila za kreiranje dijagrama tokova podataka su detaljno opisana u (Lazarević et al., 2006; SSA, 2000). Napomenućemo samo da je najznačajnije pravilo koje se mora poštovati pri dekompoziciji procesa *pravilo balansa tokova* koje glasi „ulazni i izlazni tokovi na celokupnom DTP-u koji je dobijen dekompozicijom nekog procesa P moraju odgovarati ulaznim i izlaznim tokovima toga procesa P na dijagramu višeg nivoa“ (Lazarević et al., 2006). Pri tome se uzima u obzir dekompozicija tokova predstavljena u Rečniku podataka. U Rečniku

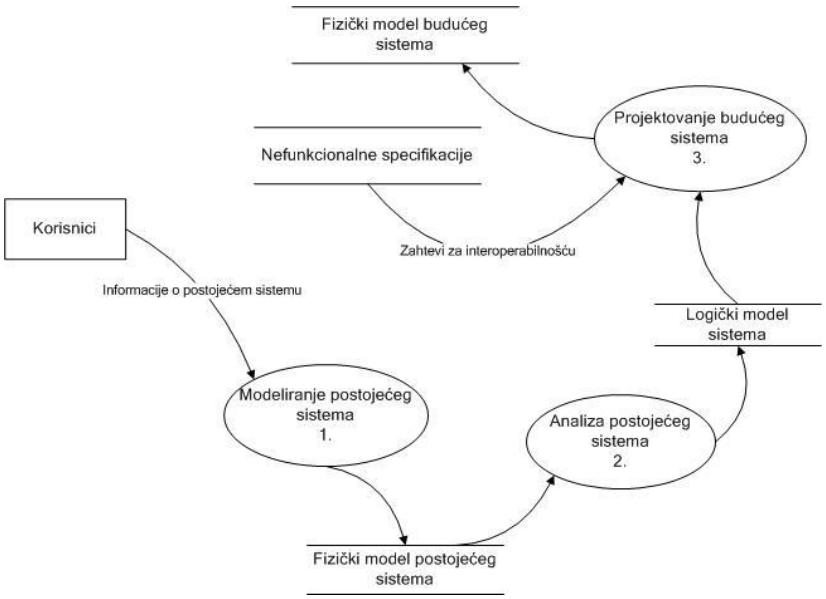
podataka SSA se „opisuju struktura i sadržaj tokova i skladišta podataka i prikazuje, ako je to potrebno, njihova dekompozicija“ (SSA, 2000).

3.4.1.2. Metodološki aspekti modelovanja procesa

U ovom poglavlju su opisani osnovni pristupi i praksa izgradnje funkcionalnih modela koju predlaže FON Labis metodologija. Fon Labis metodologija ne definiše precizne korake i stroga pravila koja treba poštovati pri modelovanju procesa. Preduslov uspešnog modelovanja zahteva za funkcionisanjem realnog sistema je njegovo dobro poznavanje, ali u velikoj meri i iskustvo i sposobnost analitičara.

Prva preporuka se odnosi na definisanje razlike između **logičkog i fizičkog modela** procesa (SSA, 2000). Autori naglašavaju da SSA treba da identifikuje i da precizno opiše logički model procesa, odnosno da definiše „ŠTA“ budući IS treba da obezbedi. Odgovor na pitanje „KAKO“ će se suštinski procesi realnog sistema koji su definisani logičkim modelom realizovati je van opsega SSA. Naime, specifikacija fizičkog modela procesa i njegova implementacija je tema relevantna za sledeće faze ravoja IS.

Za definisanje logičkog modela funkcija FON Labis metodologija predlaže kombinaciju dva pristupa (SSA, 2000): (1) direktno modelovanje na osnovu poznavanja suštinskih procesa realnog sistema i (2) izdvajanje logičkog iz fizičkog modela procesa. Postupni postupak za **izdvajanje logičkog iz fizičkog modela procesa** je prikazan DTP-om na slici 3.8.



Slika 3.8 Specifikacija IS na osnovu snimanja stanja i zahtevi za interoperabilnošću. Preuzeto i modifikovano prema (SSA, 2000)

Fizički model postojećeg sistema se definiše na osnovu informacija o postojećem sistemu koje su rezultat snimanja postojećeg stanja. U odgovarajućem skladištu *Fizički model postojećeg sistema* se pamte informacije o dve vrste različitih procesa: suštinskim procesima i dopunskim, odnosno posredničkim procesima. Dopunski procesi su rezultat konkretne implementacije sistema i zavise od tehnologije i organizacije. Drugi proces, *Analiza postojećeg sistema* je ključan za uspešnu specifikaciju IS. On zahteva veštinu da se prepoznaju i uklone svi dopunski procesi, tokovi i skladišta podataka. Prema (SSA, 2000) logički model sistema treba da sadrži samo suštinske procese. Pri specifikaciji IS je bitno uzeti u obzir i nefunkcionalne zahteve koji zavise od buduće tehnologije i organizacije sistema. Nefunkcionalni zahtevi se pamte u skladištu *Nefunkcionalne specifikacije*. Na DTP-u prikazanom na slici 3.8 je posebno izdvojen tok *Zahtevi za interoperabilnošću*, sa ciljem da se ukaže na to da su od nefunkcionalnih karakteristika sistema u tezi ključni zahtevi za interoperabilnošću. Zahtevi za interoperabilnošću su detaljno analizirani i klasifikovani u poglavlju pet (5.1.1). Kao osnovni nedostatak opisanog pristupa, autori ističu da detaljno i kompletno snimanje postojećeg stanja zahteva puno vremena i truda (SSA, 2000). Oni dalje navode da do problema i nesuglasica može da dođe i pri definisanju logičkog na osnovu fizičkog modela postojećeg sistema. Iz

tog razloga se predloženi pristup u praksi najčešće kombinuje sa *metodom direktnog modeliranja logičkog sistema*.

Prema (SSA, 2000), za primenu **metode direktnog modeliranja logičkog sistema** su poželjna dva preduslova: dobro poznavanje postojećeg sistema i postojanje nekog opšteg teorijskog modela vrste sistema koji se analizira. Za analizu specifičnosti karakteristika konkretnog sistema se obično koriste dva dopunska metodološka pristupa: (1) analiza odziva sistema na specifične događaje i (2) analiza životnog ciklusa osnovnih delatnosti i resursa u sistemu (SSA, 2000).

Metoda direktnog modeliranja se sastoji u identifikaciji suštinskih procesa, koji se sastoje od skupa akcija kojim sistem reaguje na jedan i samo jedan događaj (spoljni, vremenski ili interni) (SSA, 2000). Ovi procesi komuniciraju isključivo preko suštinske memorije, koja predstavlja osnovne fizičke ili konceptualne objekte posmatranog sistema. Autori (SSA, 2000) predlažu sledeće korake za direktno modeliranje sistema: (1) identifikacija svrhe ili cilja postojanja IS, (2) identifikacija suštinskih ili fundamentalnih procesa sistema, (3) identifikacija neophodnih informacija i (4) identifikacija procesa održavanja. Navedeni koraci su detaljno opisani u (SSA, 2000), i njihova primena neće biti detaljno diskutovana u tezi.

Drugi pristup koji se predlaže za direktno modelovanje logičkog sistema je „analiza životnih ciklusa i osnovnih delatnosti i resursa u sistemu“. Ovaj pristup podrazumeva podelu funkcija u sistemu koji se analizira na sledeće grupe (Lazarević et al., 2006): (1) funkcije organizovanja, planiranja i upravljanja, (2) funkcije obavljanja osnovnih delatnosti i (3) funkcije upravljanja resursima. Opšta preporuka je da se pri analizi sistema, na prvom nivou definišu ove tri grupe funkcija.

Jedan od osnovnih problema vezanih za primenu metode SSA se odnosi na pitanje kako vršiti dekompoziciju sistema, odnosno kako zaključiti da proces ne treba dalje dekomponovati. FON Labis metodologija predlaže sledeće opšte pravilo koje glasi: „dekompoziciju treba vršiti dok se neki proces može prirodno dekomponovati na suštinske paralelne procese koji međusobno komuniciraju isključivo preko suštinskih skladišta“ (SSA, 2000). Naime, predlaže se da se dekompozicija okonča

kada se dođe do procesa koji su po svojoj prirodi sekvencijalni. Pored opšteg kriterijuma dekompozicije je definisan i pomoćni kriterijum koji glasi: „dekompoziciju treba vršiti sve dok se ne dobiju procesi sa jednim ulaznim i/ili izlaznim tokom podataka“ (SSA, 2000).

3.4.2. Izrada modela podataka

Prema (Lazarević et al., 2006), „konceptualni model predstavlja suštinske karakteristike sistema za koji se projektuje baza podataka, opisane na visokom nivou apstrakcije, preko modela podataka.“ Autori ističu da konceptualni model „predstavlja celokupan mogući informacioni sadržaj baze podataka, nezavisno od toga koji je deo tog sadržaja i kako implementiran“. Kao sredstvo za izradu jedinstvenog modela podataka u fazi *realizacije*, FON Labis metodologija predlaže kombinaciju PMOV-a kao konvencionalne metode, i UML dijagrama klasa kao relevantne objektne metode.

3.4.2.1. PMOV-Prošireni model objekti i veze

Za izradu modela podataka u fazi *Realizacije* FON Labis metodologija predlaže upotrebu Proširenog modela objekti i veze (PMOV). On obezbeđuje interpretaciju podataka o posmatranom realnom sistemu koja se u svakom modelu podataka ostvaruje kroz njegove tri osnovne komponente (Lazarević et al., 2006; PMOV, 2000):

- **Strukturu podataka**, tj. skup podataka za opis statičkih karakteristika sistema koje obuhvataju: objekte sistema, njihove atribute i međusobne veze.
- **Ograničenja,** tj. semantička ograničenja na vrednost podataka koja u svakom stacionarnom stanju moraju biti zadovoljena. Ova dodatna ograničenja na podatke koja nisu obuhvaćena samom strukturom podataka se nazivaju pravilima integriteta modela podataka.
- **Operacije** nad konceptima strukture preko kojih je moguće definisati dinamiku sistema, odnosno dati interpretaciju podataka kroz obradu podataka u modelu podataka.

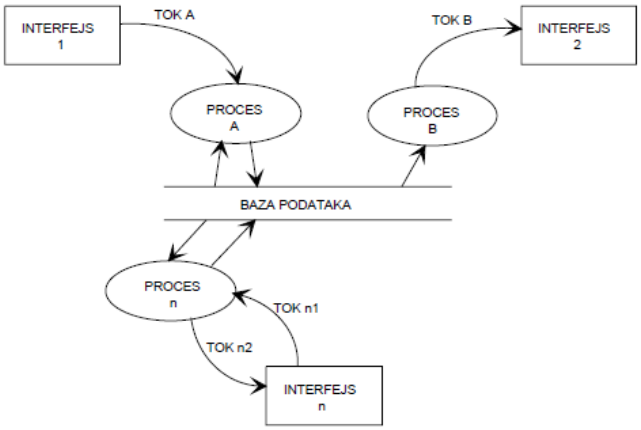
Pomoću standardnog Modela objekti i veze (MOV-a) je moguće opisati samo strukturu sistema, konceptima kao što su: objekti i njihovi atributi, međusobne veze objekata i ograničenja. PMOV koji predstavlja proširenje MOV-a, uključuje skup dozvoljenih operacija nad konceptima modela i pravila integriteta za one operacije koje mogu da naruše odgovarajuća ograničenja (PMOV, 2000). Dva osnovna tipa pravila integriteta kojima se ostvaruju ograničenja su: (1) RESTRICTED tip kojim se odbija izvršenje operacije koja narušava ograničenje i (2) CASCADES tip kojim se aktivira operacija koja oporavlja narušeno ograničenje.

3.4.2.2. Metodološki aspekti modelovanja podataka

PMOV ili UML dijagram klasa koji se preporučuju za definisanje modela podataka realnog sistema, predstavljaju samo alat za opisivanje skupa povezanih objekata i događaja u njemu. FON Labis metodologija daje samo opšte preporuke i smernice za složeni postupak izrade jedinstvenog modela podataka realnog sistema. Postoji više metodoloških pristupa konceptualnom modelovanju, a u razvoju modela nekog konkretnog sistema oni se gotovo uvek kombinuju (Lazarević et al., 2006): integracija podmodela; direktno modelovanje na bazi verbalnog opisa sistema; konkretizacija opštih (generičkih) modela, odnosno korišćenje uzora ("paterna"); normalizacija relacija i transformacija jednog modela u drugi ("direktno" i "inverzno" inženjerstvo). Navedeni pristupi su detaljno opisani u (Lazarević et al., 2006) i tezi neće biti detaljnije diskutovani.

3.4.2.2.1. Specifikacija baze podataka

Na osnovu dosadašnjeg izlaganja, pokazano je kako je moguće izvršiti specifikaciju nekog IS pomoću SSA i PMOV-a. Pomoću metode SSA je izvršena hijererhijska dekompozicija sistema do nivoa primitivnih funkcija, dok je pomoću PMOV-a specifikovana jedinstvena baza podataka IS. S obzirom na to da procesi na DTP-u komuniciraju samo preko skladišta, a da PMOV definiše jedinstvenu bazu podataka IS, autori su pokazali da fizički DTP može da se prikaže kao na slici 3.9 (PMOV, 2000). Na slici 3.9 su prikazani samo primitivni procesi i oni predstavljaju aplikacije budućeg sistema. Dijagrami dekompozicije na višim nivoima predstavljaju samo menije, odnosno način izbora pojedinih aplikacija.



Slika 3.9 Fizički DTP IS zasnovanog na integrisanoj bazi podataka (PMOV, 2000)

Imajući u vidu opštu arhitekturu budućeg fizičkog sistema koja je prikazana na slici 3.9, autori ističu da jedna potpuna specifikacija IS obuhvata (PMOV, 2000): (1) specifikaciju baze podataka, (2) specifikaciju aplikacija i (3) nefunkcionalne specifikacije.

Specifikacija baze podataka podrazumeva sledeća tri koraka koja su detaljno opisana u (PMOV, 2000). U prvom koraku se kreira dijagram proširenog modela objekti i veze, i definišu se pravila dinamičkog strukturnog integriteta. U drugom koraku se vrši definicija svih atributa i domena sa ograničenjima. Poželjno je definisati i akcije koje se pozivaju pri narušavanju integriteta. U trećem koraku se vrši specifikacija svih vrednosnih i odgovarajućih dinamičkih pravila integriteta.

3.4.3. Specifikacija aplikacija

Rezultat primene metode SSA je skup primitivnih funkcija, na osnovu kojih se vrši identifikacija budućih aplikacija informacionog sistema. FON Labis metodologija predlaže da se specifikacija aplikacija radi prototipski, na osnovu složenih aplikativnih objekata. Koncept složenog objekta je više godina razvijan i uspešno primenjivan u Laboratoriji za informacione sisteme FON-a. Prema (PMOV, 2000), složeni objekat se karakteriše spoljnim izgledom (ekranska forma ili format izveštaja) i operacijama koje se nad njim mogu izvršavati. Autori preporučuju da se u prototipu operacije nad složenim objektima definišu kao opcije u meniju, i da se precizna specifikacija operacija da u formi pseudokoda ili da se opiše detaljnim opisom slučaja korišćenja. Specifikacija aplikacije podrazumeva definisanje

strukture ulaznih, odnosno izlaznih tokova i načina na koji ulazni tok ažurira bazu podataka, odnosno način na koji se iz baze dobija izlazni tok (SSA, 2000). Struktura ulaznih i izlaznih tokova je opisana u rečniku podataka SSA. Ovi tokovi su složeni objekti koji se mogu izvesti iz strukture baze podataka (baznih objekata), budući da se baza podataka projektuje na osnovu njih.

Opšta arhitektura aplikacije je prikazana na slici 3.10. U svakoj aplikaciji se formira neki složeni objekat, a zatim se on koristi da se preko njega izvrši ažuriranje baze podataka i/ili se prikazuje korisniku. Na osnovu DTP-a opšte arhitekture aplikacije može da se zaključi da specifikacija aplikacija obuhvata (PMOV, 2000): (1) definisanje struktura njenih složenih objekata, (2) definisanje načina formiranja složenog objekta, odnosno njegovih komponenti, (3) definisanje operacija ažuriranja koje se preko složenog objekta izvršavaju i (4) specifikaciju spoljnjeg izgleda složenog objekta.

Slika 3.10 Opšta arhitektura aplikacije (PMOV, 2000)

3.4.3.1. Izrada slučajeva korišćenja

Sa tačke gledišta analize sistema i izgradnje njegovog poslovnog modela, *slučaj korišćenja* se definiše kao specifikacija interakcije između sistema i jednog ili više aktera i sistema, zajedno sa opisom akcija sistema u ovoj interakciji (Lazarević et al., 2006). Sa tačke gledišta specifikacije aplikacija *slučaj korišćenja* se definiše kao specifičan način na koji će aktor da koristi jednu buduću aplikaciju IS koji se projektuje. Aktor se i u jednom i u drugom slučaju definiše kao uloga koju igra neki entitet van sistema u jednoj ili više interakcija sa sistemom (Lazarević et al., 2006).

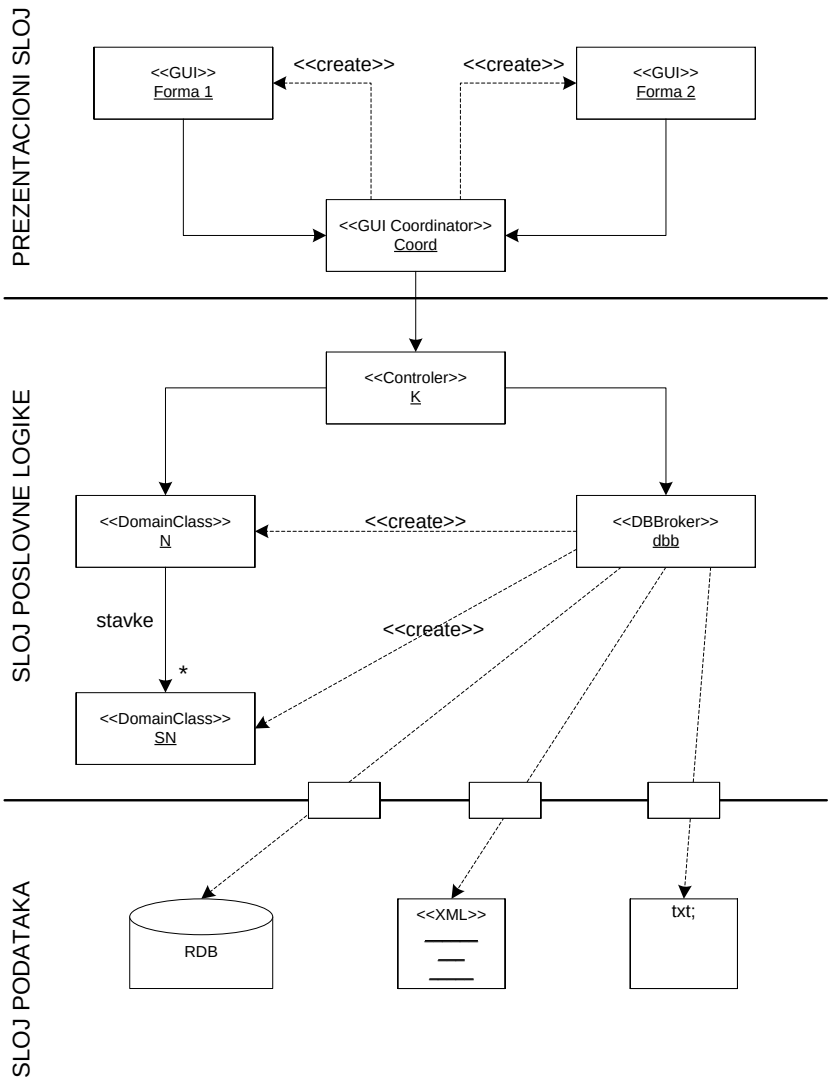
Formalno *Model slučajeva korišćenja* može da se definiše kao graf sa dva tipa čvorova: (1) čvorovi tipa aktor i (2) čvorovi tipa slučaj korišćenja (Larman, 2004). Direktna asocijacija između dva aktora ili dva slučaja korišćenja nije dozvoljena. Svaki slučaj korišćenja treba da bude detaljno opisan. Jedan slučaj korišćenja predstavlja skup sekvenci događaja. Jedna sekvenca događaja se naziva scenario (Larman, 2004; Lazarević et al., 2006). Postoji jedan osnovni scenario i skup mogućih izuzetaka, odnosno alternativnih funkcionisanja. Uobičajeno je da se posebno daje opis osnovnog scenarija, a posebno opis svih alternativnih načina funkcionisanja (Cockburn, 2001; Larman, 2004; Lazarević et al., 2006).

3.4.3.2. Specifikacija dijagrama sekvenci

Realizacija svakog od slučajeva korišćenja se specificira preko dijagrama sekvenci u skladu sa pravilima koja su definisana izabranom logičkom arhitekturom. Kao arhitekturni patern se koristi Use Case (UC) Model-View-Controler (MVC) koji se uklapa u opšte usvojenu troslojnu arhitekturu aplikacija (slika 3.11). Predložena troslojna arhitektura aplikacija se sastoji od tri osnovna sloja:

- **Prezentacioni sloj** koji predstavlja ulazno-izlaznu reperezentaciju softverskog sistema;
- **Sloj poslovne logike** koji opisuje strukturu i ponašanje softverskog sistema;
- **Sloj podataka** koji služi za fizički smeštaj podataka.

Za dizajn se može koristiti patern *Database Brokera*, ali nije obavezan, s tim da onda funkcionalnost treba delegirati Kontroloru i samim perzistentnim objektima. Forma komunicira samo sa svojim Kontrolorom i od njega dobija sve potrebne podatke za prikaz korisniku. Između Forme i Kontrolora se ne smeju razmenjivati objekti, što znači da parametri i povratne vrednosti funkcija treba da budu nekog od osnovnih tipova podataka (long, int, double, string itd.), niz ili kolekcija gde su elementi osnovnih tipova podataka ili tipa kursora kao što je Recordset, ResultSet, DataSet, DataTable ili DataRow. Kontrolor implementira logiku posmatranog slučaja korišćenja, upravlja transakcijom i komunicira sa bazom podataka kako bi obezbedio sve potrebne podatke koje zahteva Forma. Kontrolor otpočinje i potvrđuje transakciju nad bazom podataka.



Slika 3.11 Opšta logička arhitektura aplikacija

3.4.3.3. Generisanje konačnog dijagrama klasa

U ovom koraku se na osnovu dijagrama sekvenci, vrši generisanje konačnog dijagrama klasa. Za razliku od logičkog konceptualnog modela podataka, koji se kreira u prvom koraku faze *realizacije*, konačni dijagram klasa uključuje klase koje su definisane na osnovu predložene logičke arhitekture, na primer Forma, Kontrolor i Domenske klase. Takođe, pored definicije atributa, klase uključuju i metode koje su specificirane na dijagramu sekvenci.

3.4.4. Implementacija sistema

Za fazu implementacije sistema su karakteristična dva koraka. U prvom koraku se na osnovu fizičkog modela podataka generiše odgovarajuća šema baze podataka u izabranom okruženju. U drugom koraku se vrši transformacija u programski kod. Na osnovu specifikacije, tj. konačnog dijagrama klasa, a u skladu sa MVC paternom implementacija date specifikacije može da se izvrši na više načina:

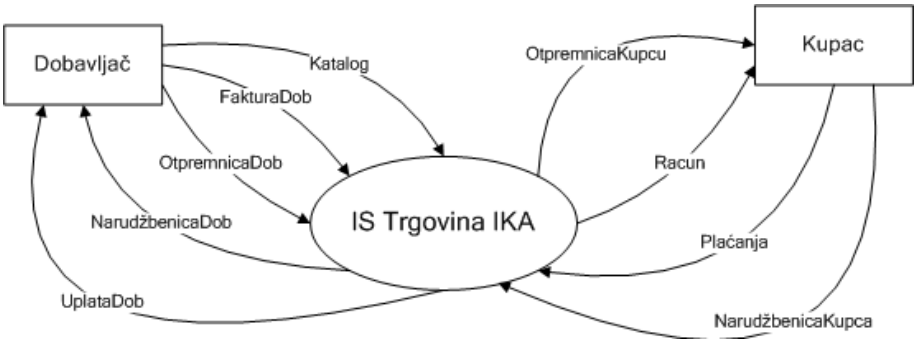
- Generisanje skupa klasa i specifikacija njihovih metoda uz manuelno ili ručno kodiranje metoda;
- Izbor nekog od <<middleware>> okvira koji su zasnovani na višeslojnim arhitekturama sistema, pa je moguće standardizovati način transformacije u odgovarajuce implementaciono okruženje.

3.4.5. Primena FON Labis metode

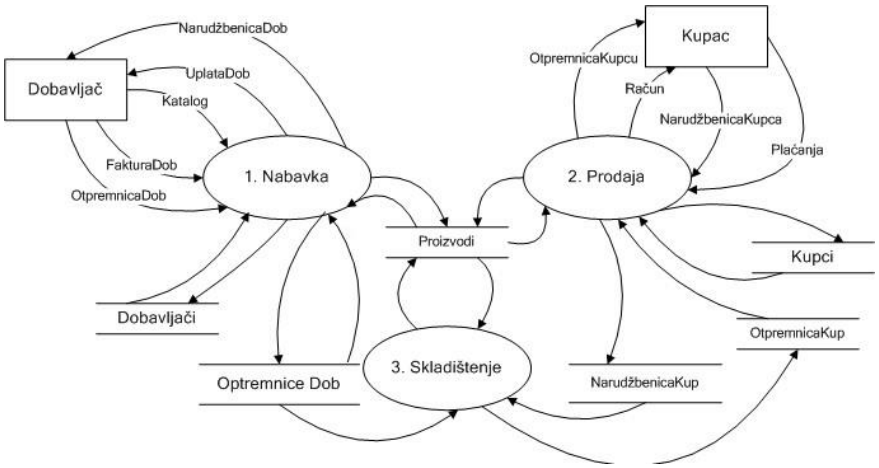
U ovom poglavlju će na jednostavnom primeru biti prikazani osnovni koraci faza identifikacije i realizacije koji su prethodno objašnjeni. U fazi identifikacije sistema (slika 3.6.) se vrši hijerarhijska dekompozicija sistema i idenitifikuju se slučajevi korišćenja.

Strukturna sistemska analiza

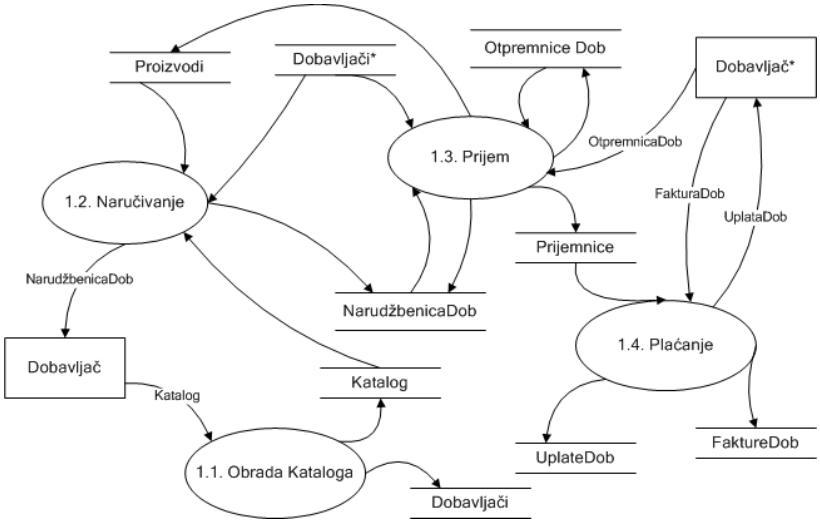
Na slikama 3.12-3.14 je predstavljen primer dekompozicije IS “Trgovina IKA”, preko dijagrama toka podataka. Na dijagramu konteksta definisana su dva interfejsa, sa kojima IS komunicira, Kupac i Dobavljač (3.12). Prikazani su i tokovi podataka preko kojih se ova komunikacija obavlja. Zatim je dat dijagram prvog nivoa dekompozicije, sa funkcijama Nabavka, Prodaja i Skladištenje (3.13). Na slici 3.14 je prikazana dekompozicija procesa Nabavka.



Slika 3.12 Dijagram konteksta



Slika 3.13 Dijagram prvog nivoa dekompozicije



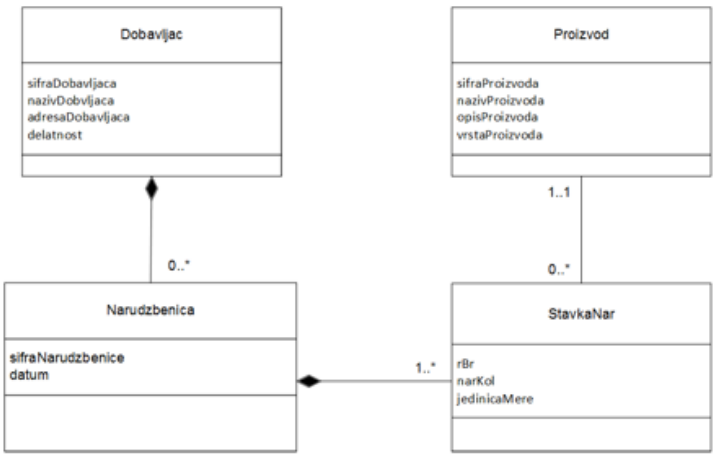
Slika 3.14 DTP Nabavke

U rečniku podataka su prikazane strukture skladišta Dobavljači, Proizvodi i opisan je tok podataka NarudžbenicaDob. U primeru nisu opisani svi tokovi podataka i

skladišta, s obzirom na to da je njegova svrha da ilustruje osnovne korake FON Labis metodologije na jednostavnom primeru.

NarudžbenicaDob: <BrojNarDob, DatumNar, ŠifraDobavljača, NazivDobavljača, AdresaDobavljača, {<RedniBroj, ŠifraProizvoda, NazivProizvoda, NarKol, JedinicaMere >}>
Dobavljači: <ŠifraDobavljača, NazivDobvljača, AdresaDobaljača, Delatnost>
Proizvodi: <ŠifraProizvoda, NazivProizvoda, OpisProizvoda, VrstaProizvoda>

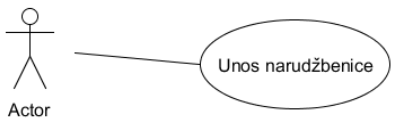
Na osnovu rečnika podataka se kreira konceptualni dijagram klasa, koji je prikazan na slici 3.15.



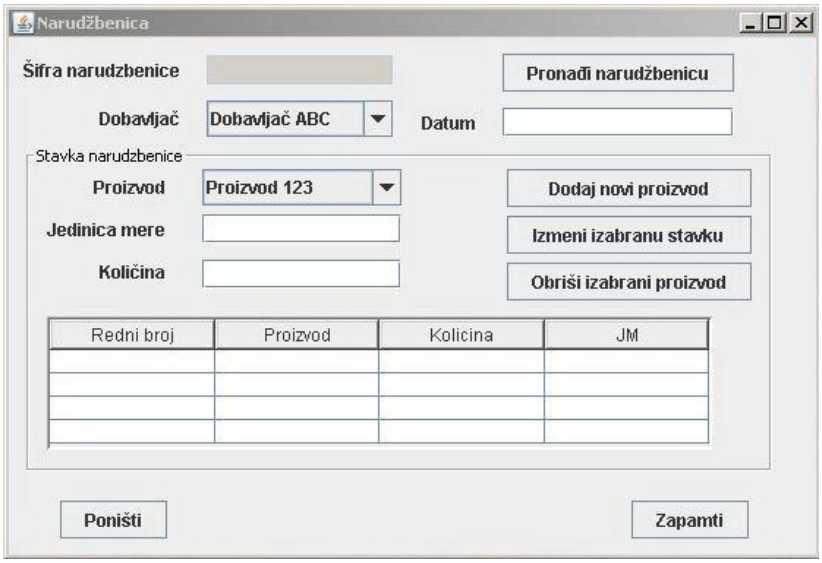
Slika 3.15 Konceptualni dijagram klasa

Slučajevi korišćenja

Na slici 3.16 prikazan je jednostavan slučaj korišćenja za Unos narudžbenice, nakon čega sledi opis scenarija na osnovu ekranske forme za Unos Narudžbenice koja predstavlja prototip složenog objekta (slika 3.17).



Slika 3.16Dijagram slučajeva korišćenja za Unos narudžbenice



Slika 3.17 Ekranska forma za Unos narudžbenice

Naziv SK: Unos narudžbenice
Akteri: Službenik
Preduslov: -
Postuslov: -
Osnovni scenario:

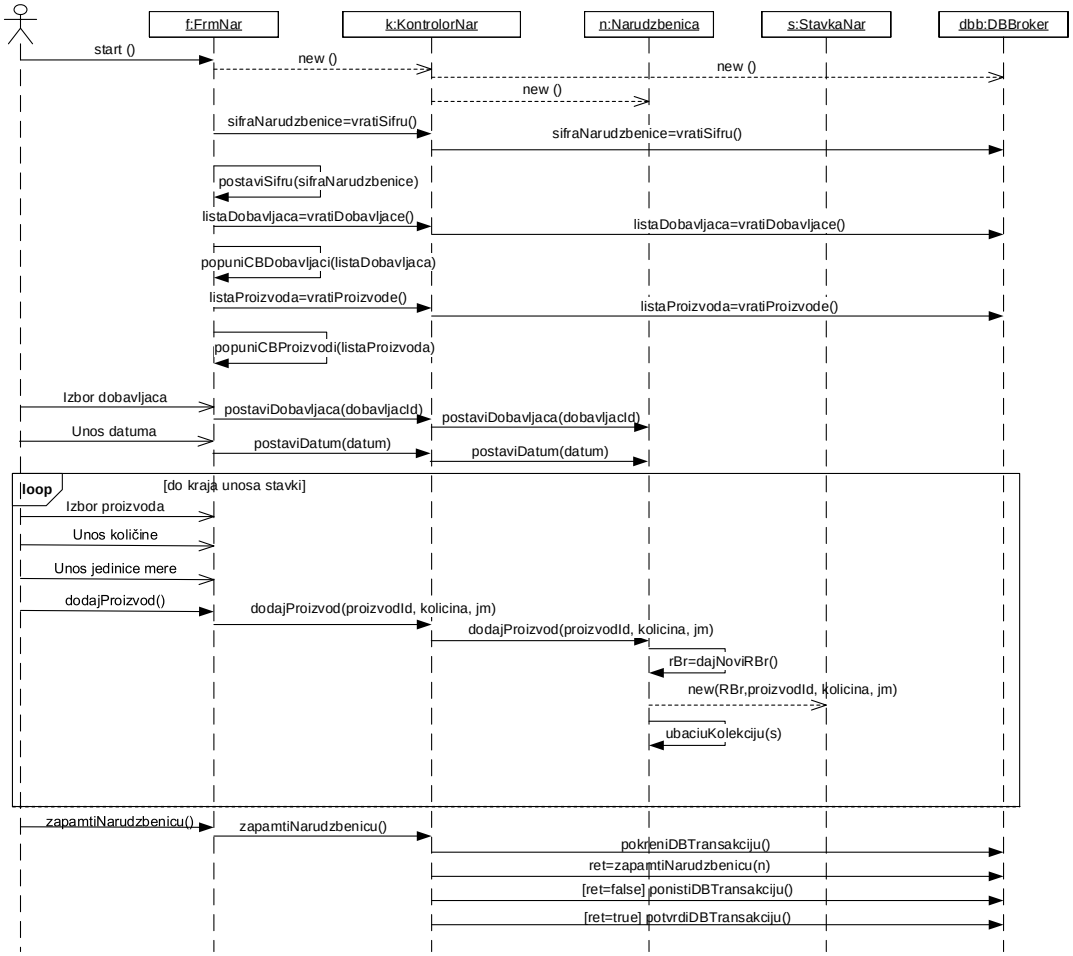
1. Službenik pokreće formu za unos nove narudžbenice.
2. Sistem instancira potrebne objekte za izvršenje slučaja korišćenja i dodeljuje šifru narudžbenice.
3. Službenik unosi datum nardužbenice.
4. Službenik iz combo box-a bira dobavljača.
5. Službenik unosi osnovne podatke za stavku narudžbenice (bira proizvod i unosi količinu i jedinicu mere).
6. Službenik pritiska dugme "Dodaj novi proizvod".
7. Službenik ponavlja korake 5 i 6 sve dok ima novih stavki narudžbenice za unos.
8. Službenik pritiska dugme "Zapamti".
9. Sistem prihvata poziv i obezbeđuje pamćenje svih objekata u bazu podataka pod transakcijom.

Alternativni scenario:

9.1. Sistem nije u mogućnosti da sačuva narudžbenicu u bazi podataka.

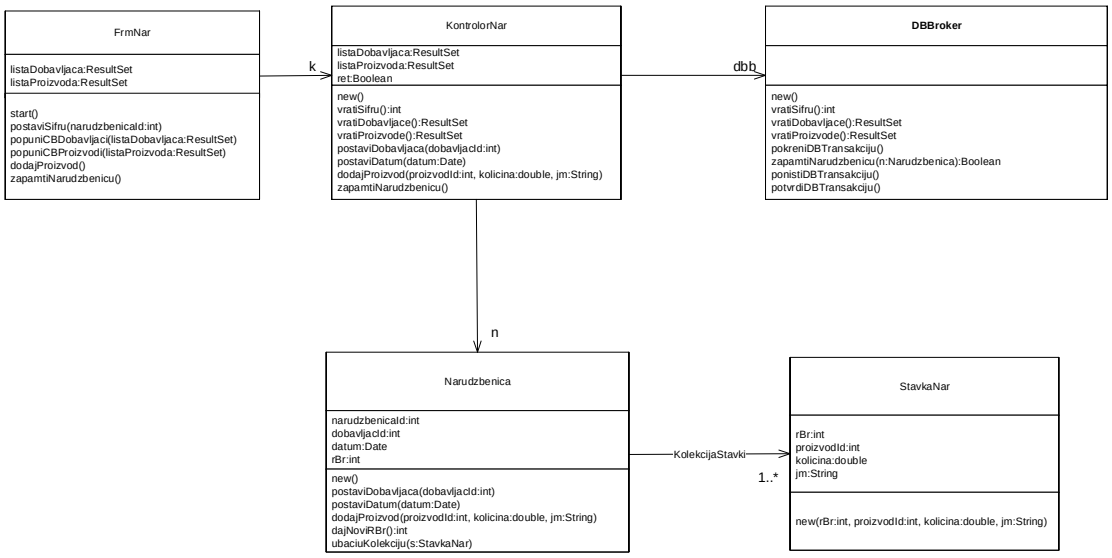
Dijagram sekvenci

Na slici 3.18 je prikazan primer dijagrama sekvenci za Slučaj korišćenja *Unos narudžbenice*. Dijagram sekvenci na slici 3.18 ilustruje primenu pravila koja su definisana opštom logičkom arhitekturom.



Slika 3.18 Dijagram sekvenci za slučaj korišćenja Unos narudžbenice

Na slici 3.19. je prikazana transformacija dijagrama sekvenci u konačni dijagram klasa.



Slika 3.19 Konačni dijagram klasa za slučaj korišćenja Unos narudžbenice

3.5. Larmanova metoda razvoja softvera

Larmanova metoda razvoja softvera je bazirana na iterativno inkrementalnom modelu životnog ciklusa softvera (Larman, 2004). Ona predstavlja objektno-orijentisanu metodu projektovanja softvera, koja za opis modela koristi Unified Modeling Language (UML) (Pilone & Pitman, 2005). Iterativno-inkrementalni model životnog ciklusa Larmanove metode podrazumeva da je razvoj softverskog sistema podeljen na više mini-projekata, odnosno iteracija. Svaka iteracija se sastoji iz pet faza: (1) specifikacija zahteva, (2) analiza, (3) projektovanje, (4) implementacija i (5) testiranje. Sistem se u svakoj iteraciji postepeno usavršava i obogaćuje se novim funkcionalnostima. Larmanova metoda je poznata i kao iterativni i evolucioni razvoj, jer povratne informacije od strane budućih korisnika i odgovarajuće adaptacije vode do odgovarajućeg sistema.

3.5.1. Specifikacija zahteva

Zahtevi predstavljaju uslove ili svojstva koje sistem, ili šire gledajući projekat mora da ispuni (Jacobson, Booch, & Rumbaugh, 1999b). Larman naglašava da je za uspeh projekta faza specifikacije zahteva ključna, odnosno da je bitno definisati „sistematičan pristup za otkrivanje, dokumentaciju, organizovanje i praćenje *promenljivih* zahteva sistema“ (Larman, 2004). Kao pogodnu tehniku za identifikaciju i dokumentaciju zahteva Larman predlaže UML *Model slučajeva korišćenja* (Use-Case Model). *Model slučajeva korišćenja* se sastoji od skupa slučajeva korišćenja, aktora i njihovih međusobnih veza (Vlajić, 2005). Slučaj korišćenja opisuje deo funkcionalnosti softverskog sistema, odnosno skup željenih korišćenja sistema od strane aktora. Larman naglašava da slučajevi korišćenja predstavljaju verbalni opis korišćenja sistema od strane korisnika za realizaciju željenih ciljeva. Naime, bitno je fokusirati se na tekst slučaja korišćenja, a ne na njegovu grafičku reprezentaciju.

Prema Larmanu, scenario se definiše kao „specifična sekvenca akcija i interakcija između aktora i sistema“ (Larman, 2004). Scenario predstavlja instancu slučaja korišćenja, odnosno jednu putanju kroz slučaj korišćenja koja može da bude uspešna ili neuspešna. Prema tome, slučaj korišćenja može da se neformalno definiše kao

„kolekcija povezanih uspešnih ili neuspešnih scenarija koji opisuju kako korisnik koristi sistem za realizaciju svojih ciljeva“ (Larman, 2004). RUP definiše slučaj korišćenja kao „skup instanci slučajeva korišćenja, gde svaka instanca predstavlja sekvencu akcija koje sistem izvršava sa ciljem postizanja merljivih rezultata koji imaju vrednost za određenog korisnika“(RUP, 2016). Cockburn definiše slučaj korišćenja kao „ugovor o ponašanju sistema“ (Cockburn, 2001). Prema Larmanu, slučajevi korišćenja se primarno koriste za specifikaciju funkcionalnih i bihevioralnih zahteva (Larman, 2004).

Za opis slučajeva korišćenja mogu da se koriste različiti formati i nivoi formalizma: (1) *kratak (eng. brief) opis* je najčešće u formi jednog paragrafa koji sumira osnovni scenario slučaja korišćenja; (2) *običan* (*eng. casual) opis* se sastoji od više paragrafa, koji opisuju različite scenarije; (3) *detaljan* (eng. *fully dressed*) *opis* sadrži detaljan opis svih koraka i njihovih varijanti, kao i pomoćne sekcije, na primer preduslovi (Larman, 2004).

UML standard poseduje osnovnu notaciju za prikaz interakcije između korisnika i sistema koji se modeluje. Ipak, notacija ne specificira format za tekstualni opis interakcije. Moguće je koristiti proizvoljene šablone za tekstualni opis slučajeva korišćenja. Larman ukazuje na bitnu razliku između slučajeva korišćenja visokog nivoa (eng. high-level) i proširenih slučajeva korišćenja (eng. expanded).

Slučajevi korišćenja visokog nivoa uključuju: ime slučaja korišćenja, listu korisnika, osnovni scenario koji opisuje interakciju između korisnika i sistema. Svrha ovih slučajeva korišćenja je da definišu granice sistema, bez ulaženja u detalje. Ovaj tip slučajeva korišćenja može da bude od koristi u ranim fazama projektovanja informacionog sistema, kao što je planiranje.

Larman ukazuje na činjenicu da se za detaljan opis proširenih slučajeva korišćenja može koristiti bilo koji šablon i da ne postoji najbolji format. U svojoj knjizi (Cockburn, 2001) Cockburn je predložio šablon za detaljan opis slučajeva korišćenja koji je stekao veliku popularnost u praksi. Ovaj šablon uključuje sledeće sekcije: ime slučaja korišćenja, okvir, nivo, primarni korisnik, zainteresovana strana i njeni interesi, preduslovi, postuslovi, osnovni scenario, proširenja, specijalni zahtevi,

specijalni uslovi, tehnološki zahtevi, frekvencija izvršenja i sporna pitanja. Neki od autora za opis slučajeva korišćenja preferiraju konverzacioni format u vidu dve kolone, koji naglašava interakciju između korisnika i sistema (Constantine & Lockwood, 1999; Wirfs-Brock, 1993).

Prema Larmanu bez obzira na format za opis slučajeva korišćenja koji se koristi, kjučno je detaljno i pažljivo opisati osnovni scenario i njegova moguća proširenja. **Osnovni scenario** se u literaturi naziva još i „srećna putanja" (eng. happy path), osnovni tok (eng. basic flow) ili tipičan tok (eng. typical flow). Osnovni scenario opisuje sve korake uspešne putanje, vodeći računa o njihovom redosledu izvršenja. U većini slučajeva osnovni scenario ne sadrži uslove i grananja. Moguća su tri različita tipa koraka: (1) interakcija između učesnika, (2) validacija (obično od strane sistema) i (3) promena stanja od strane sistema (na primer snimanje ili ažuriranje). **Ekstenizije ili alternativni tokovi** predstavljaju proširenje osnovnog scenarija, i ukazuju na ostale uspešne ili neuspešne grane.

Radi bolje preglednosti i lakšeg razumevanja, Larman savetuje da se vodi računa o sekvencijalnoj numeraciji svih koraka osnovnog scenarija. Koraci alternativnog scenarija treba da budu numerisani u skladu sa odgovarajućim korakom osnovnog scenarija koji proširuju. Na primer, *Koraku 2* osnovnog scenarija mogu da se pridruže *Koraci 2a* i *2b* alternativnog scenarija. Koristeći predloženi način numeracije, šablon obezbeđuje jasan okvir za naredne faze procesa: sistem sekvencne dijagrame i ugovore o operacijama. Jedna od osnovnih karakteristika Larmanovog procesa je da su faze logički povezane i da se modeli u narednim fazama izvode na osnovu dobro postavljenih modela prethodne faze.

Larman se slaže sa stavom Flower-a i Cockburn-a, vodećih stučnjaka za pisanje slučajeva korišćenja, da se pri kreiranju slučajeva korišćenja treba fokusirati na tekstualni opis (Cockburn, 2001; Fowler, 1996). Ipak, dijagram slučajeva korišćenja je pogodan za sticanje opšte slike o sistemu i njegovom širem kontekstu. Pomoću dijagrama slučajeva korišćenja je moguće definisati granice sistema, ilustrovati eksterne učesnike i način na koji koriste sistem. Neki od saveta po pitanju notacije za model slučajeva korišćenja koje predlaže Larman su: primarne učesnike prikazati

sa leve strane; sekundarne učesnike prikazati sa desne strane dijagrama; za učesnike koji reprezentuju kompjuterski sistem se predlaže uvođenje stereotipa <<actor>>.

3.5.2. Analiza

Faza analize se oslanja na dobro definisane i objašnjene slučajeve korišćenja. Osnovni zadatak faze analize je opis logičke stukture i ponašanja softverskog sistema. Za opis logičke strukture Larman preporučuje izradu domenskog, odnosno konceptualnog modela podataka. Ponašanje softverskog sistema se opisuje pomoću sistemskih dijagrama sekvenci i ugovora o sistemskim operacijama.

3.5.2.1. Domenski model

Domenski model predstavlja vizuelnu reprezentaciju konceptualnih klasa ili stvarnih objekata u domenu (Fowler, 1996; J. Martin & Odell, 1997). Za domenski model se u literaturi koriste i nazivi „konceptualni model“ i „domenski objektni model“. Unifed Process (UP) definiše domenski model kao „reprezentaciju stvarnih konceptualnih klasa, koje nisu softverski objekti“ (Larman, 2004). RUP definiše UP domenski model, kao specijalizaciju UP Business Object Model–a (BOM) i ističe da je njegov primarni cilj da obezbedi razumevanje osnovnih koncepata poslovnog domena (RUP, 2016).

Larman predlaže da se za reprezentaciju domenskog modela koristi UML dijagram klasa koji treba da prikaže: domenske objekte ili konceptualne klase, veze između konceptualnih klasa i odgovarajuće atribute. Na domenskom dijagramu klasa ne treba definisati metode, odnosno operacije (Larman, 2004). Domenski model predstavlja „vizuelni rečnik“ stvarnih koncepata, i ne treba da uključi softverske objekte kao što su na primer Java klase i njihove metode.

Domenski model ilustruje domenske klase, koje se neformalno definišu kao ideja, stvar ili objekat. Martin i saradnici definišu konceptualnu klasu pomoću termina: *simbol*, *intenzija* i *ekstenzija* (J. Martin & Odell, 1997). *Simboli* predstavljaju reči ili slike koje reprezentuju konceptualnu klasu. *Intenzija* je definicija konceptualne klase, dok *ekstenzija* predstavlja skup primera na koje se konceptualna klasa odnosi.

Domenski model ne može da se poistoveti sa modelom podataka, tako da je opravdano imati u domenskom modelu klase koje nemaju atribute, ili čija je priroda čisto bihevijoralna.

S obzirom na to da domenski model prikazuje konceptualne klase, centralni problem je kako identifikovati one klase koje su relevantne za dati domen. Larman preporučuje tri strategije za identifikaciju konceptualnih klasa (Larman, 2004):

1) Ponovna upotreba ili modifikacija postojećih modela, predstavlja najlakši pristup s obzirom na to da u literaturi postoji veliki broj opštih domenskih modela, kao i modela podataka koji mogu da se prevedu u domenske modele (Fowler, 1996; Hay, 2006)
2) Upotreba postojećih lista kategorija konceptualnih klasa, može da ubrza kreiranje domenskog modela.
3) Identifikacija konceptualnih klasa na osnovu imenica iz tekstualnog opisa slučajeva korišćenja.

Identifikacija konceptualnih klasa na osnovu lingvističke analize, koju je predložio Abbott se pokazala kao izuzetno efikasna metoda koja je jednostavna za primenu (Abbott, 1983). Ova metoda se sastoji u identifikaciji imenica i izraza u tekstualnim opisima domena. Imenice koje su identifikovane analizom, predstavljaju kandidate za konceptualne klase ili atribute. Prilikom primene ovog metoda je bitno posebno obratiti pažnju na reči prirodnog jezika koje mogu da imaju dvosmisleno značenje. Larman ističe da mehaničko mapiranje imenica na klase nije moguće (Larman, 2004). Detaljni slučajevi korišćenja koji su rezultat faze specifikacije zahteva, predstavljaju dovoljno dobru osnovu za primenu lingvističke analize.

Konceptualne klase se sastoje od atributa, koji opisuju osobine klase. Svaki atribut ima određeni tip podatka, i pridružuje mu se konkretna vrednost za konkretno pojavljivanje klase. Konceptulane klase su međusobno povezane asocijacijama. Za svaki kraj asocijacije može da se definiše uloga koncepta koji učestvuje u asocijaciji.

3.5.2.2. Sistemski dijagrami sekvenci

Sistemski dijagram sekvenci se kreira za određeni slučaj korišćenja i prikazuje događaje koji uspostavljaju interakciju između aktora i softverskog sistema (Vlajić, 2005). Na sistemskom dijagramu sekvenci se prikazuju eksterni učesnici koji su u direktnoj interakciji sa sistemom, sistem koji je prikazan koristeći princip „crne kutije" i sistemski događaji koje učesnik generiše.

Prema Larmanu, sistemski dijagram sekvenci se kreira za određeni scenario slučaja korišćenja i prikazuje: događaje koje eksterni korisnik generiše, njihov redosled, i inter-sistemske događaje (Larman, 2004). Budući da se svi sistemi tetiraju kao „crne kutije", naglasak je na događajima koji se razmenjuju između korisnika i sistema. Larman preporučuje da se sistemski dijagram sekvenci kreira za osnovni scenario svakog slučaja korišćenja, kao i za one alternativne scenarije koji se često izvršavaju ili su dosta kompleksni.

Sistemski dijagrami sekvenci opisuju *„šta"* sistem radi, ne ulazeći u detalje *„kako"* je to moguće realizovati. Prema Larmanu softverski sistem može da reaguje na tri tipa događaja: (1) eksterni događaji koje generiše učesnik (ljudi ili kompjuter), (2) vremenski događaji i (3) greške ili izuzeci (Larman, 2004).

3.5.2.3. Ugovori o operacijama

Ugovori o operacijama se prave za operacije koje su prikazane na sistemskom dijagramu sekvenci. Sistemske operacije koje opisuju ponašanje softverskog sistema imaju potpis koji sadrži ime metode i opciono ulazne i izazne argumente (Vlajić, 2005). Sistemske operacije su deo javnog interfejsa sistema koji je prikazan pomoću koncepta „crne kutije". *Ugovori* opisuju ponašanje sistemske operacije, odnosno *„šta"* operacija treba da radi, bez objašnjenja *„kako"* je to moguće realizovati. Bitno je naglasiti da je jedan ugovor uvek vezan za jednu sistemsku operaciju. Ugovori se sastoje od sledećih sekcija (Larman, 2004):

- **Operacija** – ime operacije i parametri;
- **Veza sa slučajem korišćenja** – imena slučajeva korišćenja u kojima se poziva sistemska operacija;

- **Preduslovi** – sistem mora da bude u određenom stanju pre izvršenja sistemske operacije;
- **Postuslovi**- posle izvršenja sistemske operacije sistem mora da bude u određenom stanju ili se rezultat operacije poništava.

Postuslovi predstavljaju najbitniju sekciju ugovora o operacijama. *Postuslovi* opisuju promene stanja objekata domenskog modela nakon izvršenja sistemske operacije. Larman daje niz preporuka za kreiranje ugovora o operacijama (Larman, 2004):

- Sistemske operacije za koje se kreiraju ugovori treba identifikovati na osnovu sistemskog dijagrama sekvenci.
- Kreiranje ugovora se posebno preporučuje za kompleksne operacije ili one operacije čija upotreba nije jasna. Kreiranje ugovora za ostale operacije sa sistemskog dijagrama sekvenci je opciono.
- Za opis postuslova treba koristiti tri osnovne kategorije: (1) kreiranje ili brisanje pojavljivanja, (2) promenu atributa i (3) pravljenje ili uništavanje asocijacija.

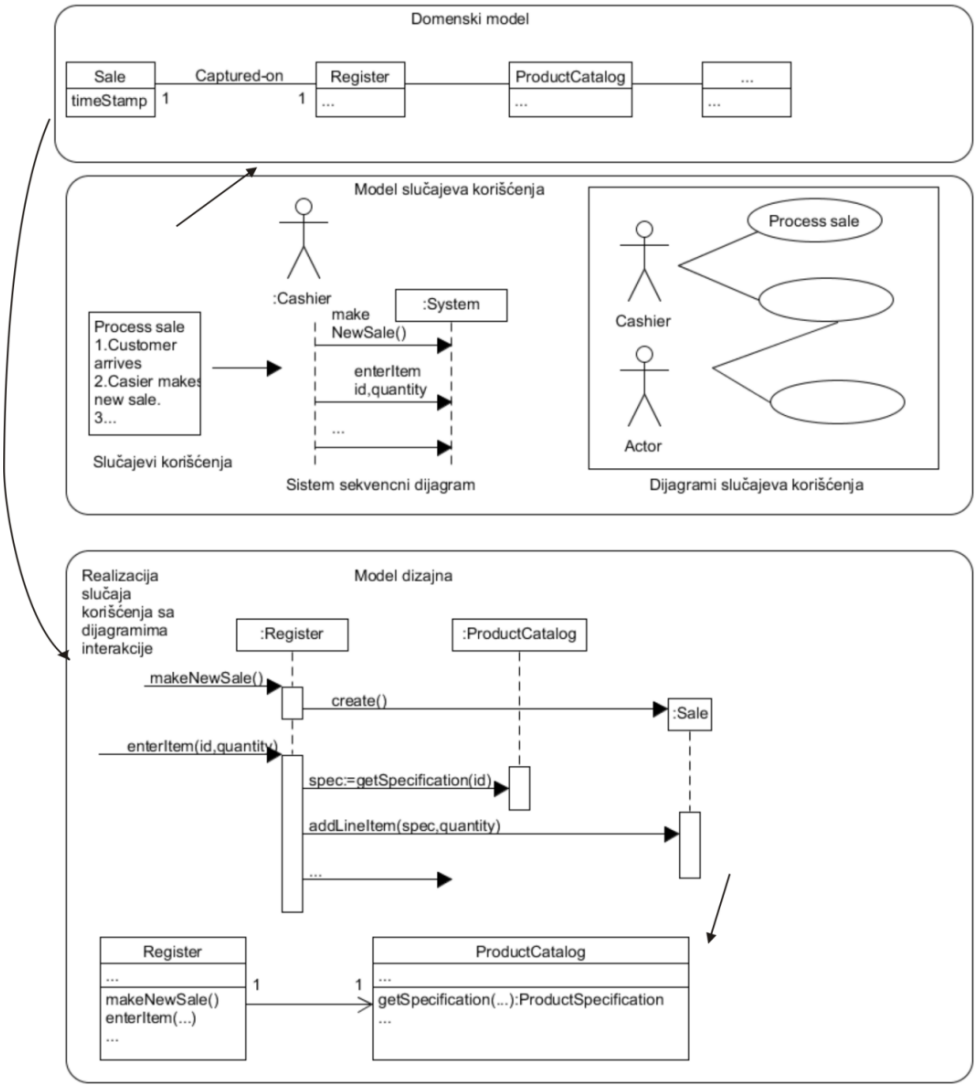
Postuslove je potrebno opisati koristeći prošlo vreme, jer je bitno da se istakne da je operacija završena, odnosno da se objekat već nalazi u novom stanju. Na primer, *Stavka narudžbenice je kreirana,* a ne *Kreiranje stavke narudžbenice*.

3.5.3. Projektovanje

Faza projektovanja opisuje fizičku strukturu i ponašanje softverskog sistema. Ugovori o sistemskim operacijama definišu granicu između faza analize i projektovanja softverskog sistema. Dok se u fazi analize pomoću ugovora o operacijama specificira *„šta“* treba da se omogući, faza projektovanja treba da obezbedi odgovor na pitanje *„kako“* je to moguće postići. Pre opisa načina ponašanja softverskog sistema, bitno je definisati njegovu arhitekturu.

Larman predlaže da se za svaki sistemski događaj koji je definisan na sistemskom dijagramu sekvenci kreira dijagram interakcije, koji opisuje način na koji objekti međusobno komuniciraju. Kao pogodne dijagrame za opis dinamike sistema Larman izdvaja dijagram sekvenci ili dijagram kolaboracije. Na dijagramu sekvenci se

prikazuje niz poruka koje objekti međusobno razmenjuju sa ciljem izvršenja određene operacije, pri čemu se uzima u obzir vremenska dimenzija. Dijagram kolaboracije predstavlja dijagram objekata na kojem su prikazane veze između njih, poruke koje šalju jedan drugom i podaci koje međusobno razmenjuju. Nakon kreiranja odgovarajućeg dinamičkog modela sistema, odnosno dijagrama sekvenci ili kolaboracije predlaže se da se na osnovu njih izgeneriše odgovarajući dijagram klasa koji opisuje statičku dimenziju sistema. Na slici 3.20 je prikazana veza između odgovarajućih modela koji se predlažu u fazama specifikacije zahteva, analize i projektovanja softverskog sistema Larmanove metode.



Slika 3.20 Relacije imeđu osnovnih tehnika. Preuzeto i modifikovano prema (Larman, 2004)

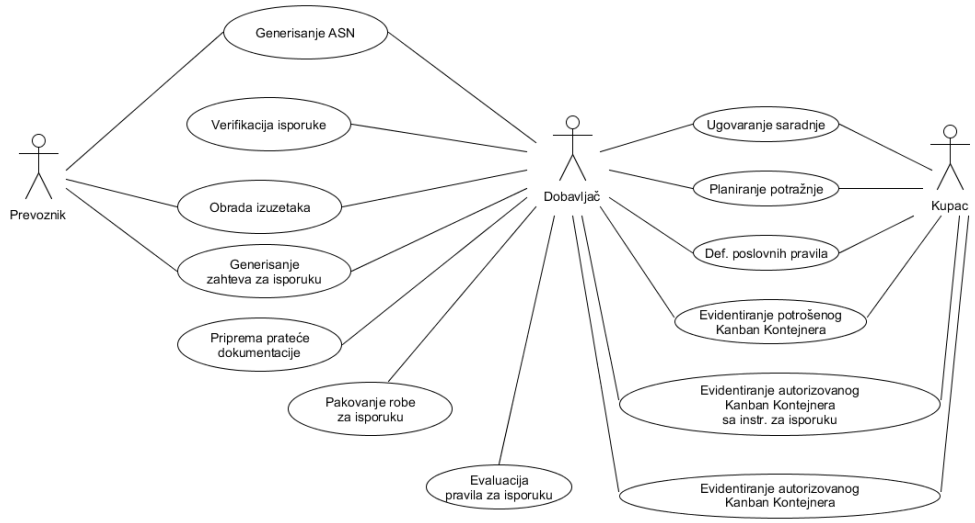
U Larmanovoj metodi razvoja softvera se za kreiranje dijagrama interakcije u fazi projektovanja preporučuje primena paterna. Na osnovu paterna se projektuje saradnja između objekata i definišu se njihove odgovornosti. Prema Larmanu paterni su imenovani parovi problem/rešenja, koji definišu savete i principe kod dodeljivanja odgovornosti objektima (Larman, 2004). Larman preporučuje upotrebu niz General Responsibility Assignment Software Patterns (GRASP) paterna: Kreator uzor, Uzor visoke povezanosti (eng. high cohesion), Uzor niske povezanosti (eng. low cohesion), Kontroler uzor, Uzor podele domena od prezentacije (eng. model-view separation pattern), itd.

3.5.4. Implementacija i testiranje

Shodno odabranoj arhitekturi softverskog sistema, i detaljnoj specifikaciji u fazi projektovanja vrši se implementacija softverskih komponenti u proizvoljnom programskom okruženju. Larman preporučuje da se testiranje izvrši kroz nekoliko nezavisnih jednica testiranja, koje se definišu za svaku softversku komponentu i podrazumevaju izradu: (1) test slučajeva koji opisuju šta test treba da obavi, (2) test procedura koje opisuju kako će se izvršiti test i (3) test komponenti koje treba da automatizuju test procedure ukoliko za to postoji mogućnost (Larman, 2004).

3.5.5. Primena Larmanove metode

U ovom poglavlju ćemo ilustrovati primenu Larmanove metode na primeru eKanban poslovnog scenarija. IV&I eKanban poslovni scenario je detaljno opisan u poglavlju 6.1. Na slici 3.21 je prikazan model slučajeva korišćenja koji se sastoji od tri aktora (Dobavljač, Prevoznik i Kupac), skupa slučajeva korišćenja koji su identifikovani na osnovu tekstualne analize verbalnog problema i veza između slučajeva korišćenja i aktora.



Slika 3.21 Model slučajeva korišćenja za eKanban poslovni scenario

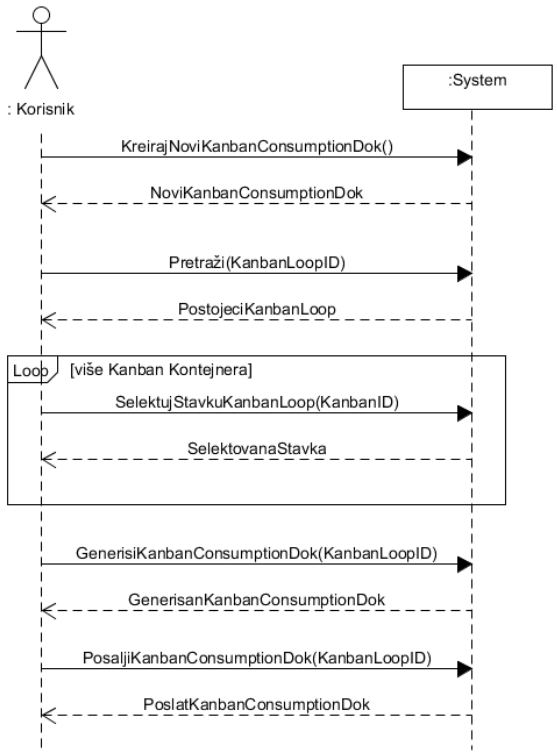
Svaki slučaj korišćenja se opisuje preko jednog glavnog scenarija, a može da sadrži i više alternativnih senarija. Opis slučaja korišćenja Evidentiranje potrošenog Kanban kontejnera je prikazan u tabeli 3.1.

Tabela 3.1 Slučaj korišćenja SK1: Evidentiranje potrošenog Kanban Kontejnera

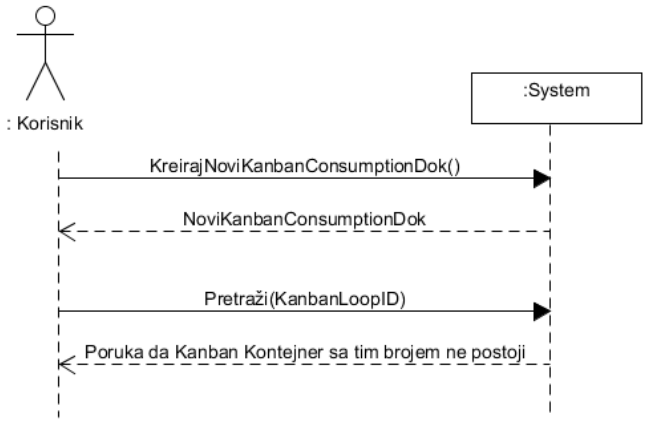
Slučaj korišćenja SK1: Evidentiranje potrošenog Kanban Kontejnera		
Okvir	eKanban IV&I poslovni scenario	
Nivo	Korisnički cilj	
Primarni aktor	Kupac	
Interesne strane	Dobavljač	
Preduslovi	Sistem je uključen i kupac je ulogovan pod svojom šifrom. Sistem prikazuje formu za kreiranje dokumenta o potrošnji Kanban kontejnera (SyncKanbanConsumption dokument). Postoji aktivan Kanban loop sa barem jednim Kanban kontejnerom čiji je status „PUN“.	
Garancija uspeha	Status Kanban kontejnera u IV&I alatu i/ili ostalim alatima koji primaju Kanban signal je postavljen na „PRAZAN“. Dobavljač je informisan o potrošnji Kanban kontejnera.	
Osnovni scenario	**Akcija učesnika**	**Odgovor sistema**
	1. Kupac poziva sistem da kreira novi SyncKanban Consumption dokument.	2. Sistem kreira novi SyncKanbanConsumption dokument.
		3. Sistem prikazuje Kupcu novi SyncKanbanConsumption dokument.

	4. Kupac bira broj Kanban Loop-a u okviru kojeg je došlo do potrošnje Kanban kontejnera.	
	5. Kupac poziva sistem da proveri da li Kanban Loop sa tim brojem postoji.	6. Sistem proverava postojanje Kanban Loop-a.
		7. Sistem prikazuje Kanban Loop ukoliko postoji. Za dati Kanban Loop sistem prikazuje sve njegove stavke, odnosno Kanban kontejnere.
	8. Kupac selektuje sve potrošene Kanban kontejnere u okviru Kanban Loopa-a koji je odabran	
	9. Kupac poziva sistem da generiše SyncKanban Consumption dokument.	10. Sistem generiše SyncKanbanConsumption dokument.
		11. Sistem prikazuje generisani SyncKanbanConsumption dokument Kupcu.
	12. Kupac kontroliše ispravnost generisanog dokumenta.	
	13. Kupac poziva sistem da pošalje generisani dokument Dobavljaču.	14. Sistem šalje dokument Dobavljaču.
		15. Sistem izveštava kupca da je dokument poslat Dobaljaču.
Alternativni scenario	4.1. Ukoliko Kanban Loop sa zadatim brojem ne postoji, sistem prikazuje Kupcu poruku da Kanban Loop ne postoji. Prekida se izvršenje scenarija. 9.1. Ukoliko sistem ne može da generiše SyncKanban Consumption dokument, prikazuje odgovarajuću poruku Kupcu. Prekida se izvršenje scenarija.	

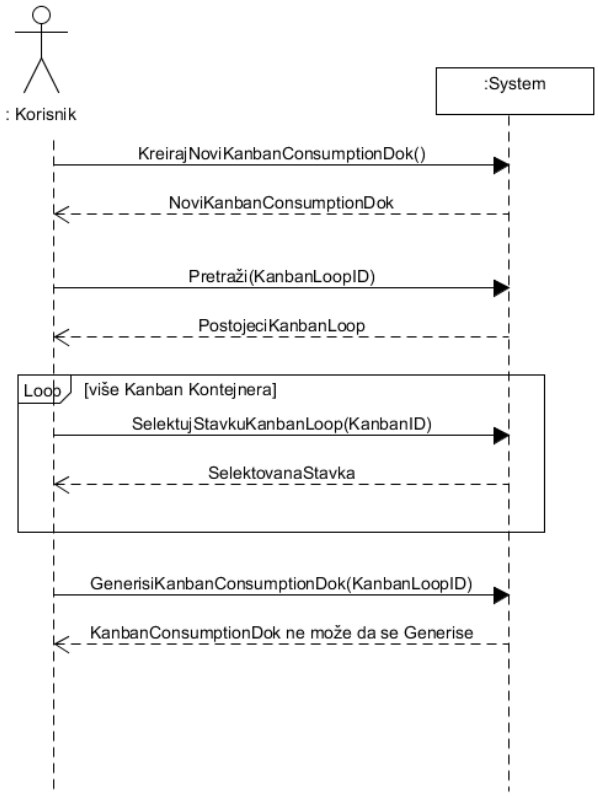
U narednom koraku se na osnovu detaljnog opisa osnovnog i alternativnih scenarija za svaki slučaj korišćenja prave posebni sistemski dijagrami sekvenci. Sistemski dijagrami sekvenci opisuju ponašanje softverskog sistema, odnosno interakciju između aktora i sistema. Na slici 3.22 je prikazan sistemski dijagram sekvenci za osnovni scenario slučaja korišćenja *Evidentiranje potrošenog Kanban Kontejnera*. Alternativna scenarija za posmatrani slučaj korišćenja su prikazana na slikama 3.23 i 3.24 respektivno.



Slika 3.22 Sistemski dijagram sekvenci (SDS) za slučaj korišćenja Evidentiranje potrošenog Kanban kontejnera



Slika 3.23 SDS-alternativni scenario za korak 4



Slika 3.24 SDS-alternativni scenario za korak 9

Kao rezultat analize scenarija identifikovano je ukupno pet sistemskih operacija koje treba projektovati:

1) NoviKanbanConsumptionDok *KreirajNoviKanbanConsumptionDok*();
2) PostojeciKanbanLoop *Pretrazi* (KanbanLoopID);
3) SelektovanaStavka *SelektujStavkuKanbanLoop* (KanbanID);
4) KanbanConsumptionDok *GenerisiKanbanConsumptionDok* (KanbanLoopID);
5) PoslatKanbanConsumptionDok *PosaljiKanbanConsumptionDok* (KanbanLoopID);

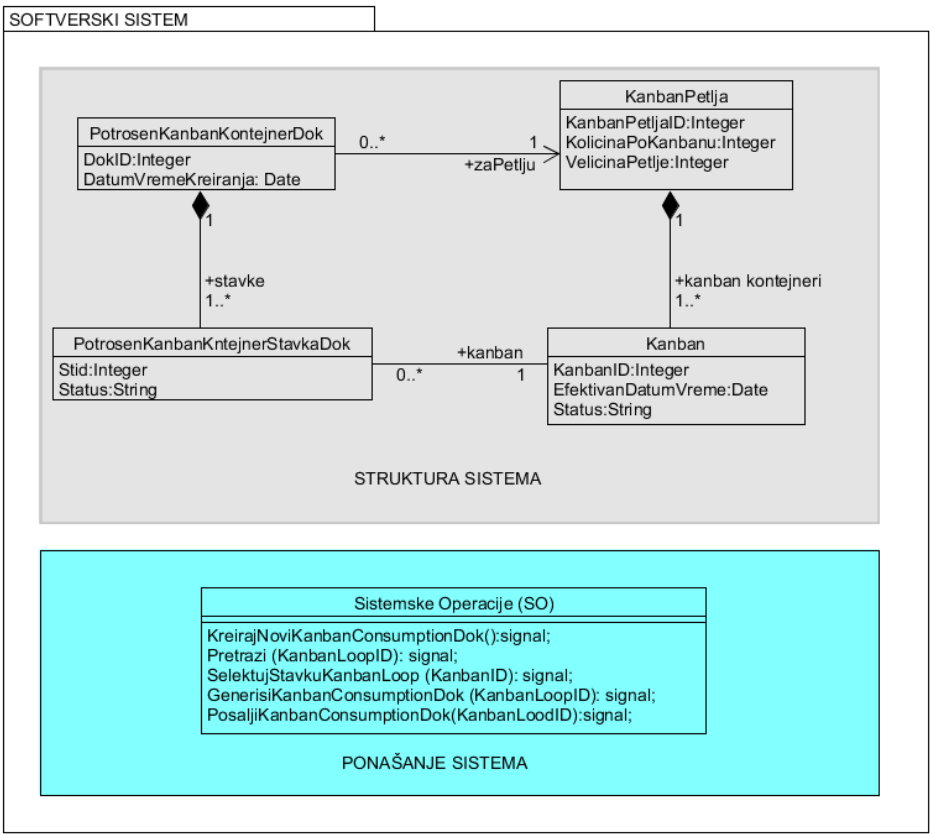
Za svaku od identifikovanih sistemskih operacija je potrebno napraviti poseban Ugovor.

Tabela 3.2 Ugovori o operacijama

UGOVORI O OPERACIJAMA	
UGOVOR UG1	
Operacija	KreirajNoviKanbanConsumptionDok():signal;
Veza sa SK	SK1: Evidentiranje potrošenog Kanban kontejnera
Preduslovi	-

Postuslovi	Napravljen je novi KanbanConsumptionDok
UGOVOR UG2	
Operacija	Pretrazi (KanbanLoopID): signal;
Veza sa SK	SK1: Evidentiranje potrošenog Kanban kontejnera
Preduslovi	-
Postuslovi	Pronađen je Kanban Loop u okviru kojeg je došlo do potrošnje Kanban kontejnera
UGOVOR UG3	
Operacija	SelektujStavkuKanbanLoop (KanbanID): signal;
Veza sa SK	SK1: Evidentiranje potrošenog Kanban kontejnera
Preduslovi	-
Postuslovi	Selektovani su svi Kanban kontejneri koji su potrošeni
UGOVOR UG4	
Operacija	GenerisiKanbanConsumptionDok (KanbanLoopID): signal;
Veza sa SK	SK1: Evidentiranje potrošenog Kanban kontejnera
Preduslovi	-
Postuslovi	Generisan je Kanban Consumption dokument Statusi potrošenih Kanban kontenera su postavljeni na "PRAZAN"
UGOVOR UG5	
Operacija	PosaljiKanbanConsumptionDok(KanbanLoodID):signal;
Veza sa SK	SK1: Evidentiranje potrošenog Kanban kontejnera
Preduslovi	-
Postuslovi	Poslat je Kanban Consumption dokument, i na taj način je Dobavljač informisan o potrošnji Kanban kontejnera.

U primeru je pokazano na koji način se pomoću sistemskih dijagrama sekvenci i ugovora o operacijama opisuje dinamika softverskog sistema. Za opis strukture softverskog sistema po Larmanovoj metodi se predlaže kreiranje domenskog modela. Osnovne klase domenskog modela su prikazane pomoću UML dijagrama klasa, a identifikovane su tekstualnom analizom verbalnog opisa IV&I eKanban poslovnog procesa. Deo domenskog modela eKanban poslovnog scenarija je prikazan na slici 3.25. Kao rezultat analize scenarija slučaja korišćenja i izrade konceptualnog modela dobija se logička struktura i ponašanje sofverskog sistema.



Slika 3.25 Struktura i ponašanja softverskog sistema koji su dobijeni primenom Larmanove metode

4. POSTOJEĆI PRISTUPI OBUHVATANJA INTEROPERABILNOSTI U RAZVOJU INFORMACIONIH SISTEMA

U ovom poglavlju je dat pregled postojećih pristupa koji se bave rešavanjem problema interoperabilnosti u razvoju informacionih sistema. U prvom delu su prikazane karakteristike reprezentativnih poslovnih arhitektura (eng. business architectures), koje su bile od koristi za sagledavanje relevantnih aspekata kolaborativnog poslovnog procesa. U drugom delu poglavlja fokus je usmeren na detaljnu analizu onih okvira interoperabilnosti, koji za cilj imaju reprezentaciju kolaborativnih poslovnih procesa na konceptualnom i tehničkom nivou. Nakon toga je dat pregled osnovnih okvira Servisno-orjentisane arhitekture (SOA). Dok su prva dva pristupa bila od koristi za identifikaciju poslovnih aspekata, SOA obezbeđuje bitne koncepte za implementaciju poslovnih procesa. Pregled i analiza relevantnih radova za predmet istraživanja disertacije je dat na kraju poglavlja.

Za sagledavanje bitnih karakteristika kolaborativnih poslovnih sistema, na osnovu kojih se mogu dati preporuke za njihovo modelovanje i implementaciju potrebno je uzeti u obzir tri polja: poslovne arhitekture, okvire za interoperabilnost i servisno-orijentisane arhitekture (slika 4.1).

Slika 4.1 Razvoj kolaborativnih poslovnih procesa koji je baziran na tri istraživačka polja

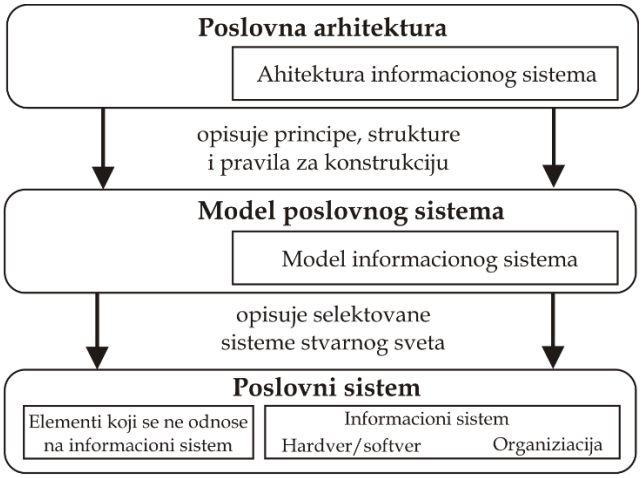
4.1. Poslovne arhitekture

Poslovne arhitekture (eng. enterprise architectures) predstavljaju sisteme koji na sveobuhvatan način opisuju poslovne procese i podržavaju njihovu realizaciju (Ziemann, 2010). Vernadat definiše poslovnu arhitekturu kao organizovanu kolekciju komponenti (alati, metodologije, jezici za modelovanje, modeli, itd.) (F. B. Vernadat, 2007). Sa njegove tačke gledišta, poslovna arhitektura opisuje veze između misije poslovnog sistema, vrste posla koji obavlja, vrste informacija koje koristi, infrastrukture, radne snage i potrebne informacione tehnologije. Prema (Schekkerman, 2003), za sistematičan razvoj kolaborativnih poslovnih procesa i mogućnost njihove automatizacije je bitno opisati procese ne samo na tehničkom, već i na poslovnom nivou.

Tražeći odgovor na pitanje koji aspekti interoperabilnosti su bitni za specifikaciju kolaborativnih poslovnih procesa, u disertaciji su uzete u razmatranje i poslovne arhitekture. Iako su poslovne arthitekture fokusirane na obezbeđenje adekvatne podrške za specifikaciju i realizaciju intra-organizacionih elementata, nisu isključene iz analize. Prema (Scheer, 1994; Ziemann, 2010), inter-organizacioni poslovni proces može da se posmatra kao specijalizacija poslovnog procesa. Drugim rečima, dimenzije koje se koriste za opis intra-organizacionih poslovnih procesa su relevantne i za inter-organizacione poslovne procese.

U literaturi se pojam poslovne arhitekture često koristi kao sinonim za arhitekturu informacionog sistema. Prema (Aier & Schönherr, 2005; Lankhorst, 2013), poslovne arhitekture i odgovarajući modeli poslovnih sistema treba da obuhvate samo one elemente koji su relevantni za razvoj informacionih sistema. Sa druge strane, (Winter & Fischer, 2006) propagiraju razliku između arhitektura informacionih sistema i poslovnih arhitektura. Oni ističu da poslovne arhitekture treba da uključe i specifikaciju onih elemenata koji su isključivo poslovno orijentisani, na primer organizacioni ciljevi, indikatori performansi, itd.

Prema (Ziemann, 2010), poslovna arhitektura sadrži širi opis poslovnog sistema koji uključuje opis i principa i strukture poslovnog modela. Na slici 4.2 je ilustrovan odnos između poslovne arhitekture, poslovnog modela i informacionog sistema.



Slika 4.2 Poslovna arhitektura, poslovni model i informacioni sistem. Preuzeto i modifikovano prema (Ziemann, 2010)

Grupa autora (Kosanke & Nell, 1997; François Vernadat, 1996) smatra da je modelovanje poslovnog sistema (eng. enterprise modeling) ključan element svake poslovne arhitekture, budući da predstavlja osnovu za razvoj poslovnog sistema. Interoperabilnost modela poslovnih sistema je detaljno analizirana u (Janković, 2007; M. Jankovic, Ivezic, Knothe, Marjanovic, & Snack, 2007).

U literaturi je prihvaćena podela na dva tipa poslovnih arhitektura (Bernus, Nemes, & Williams, 1996). Prvi tip poslovnih arhitektura (*tip 1*), opisuje fizičku strukturu određene komponente ili deo integrisanog poslovnog sistema, na primer kompjuterski sistem. Za predmet straživanja u ovoj disertaciji je relevantan drugi tip poslovnih arhitektura (*tip 2*), koji predstavlja referentnu arhitekturu koja ilustruje životni ciklus projekta za razvoj integrisanog poslovnog sistema. U poznate poslovne arhitekture *tipa 2* se ubrajaju: Zachman okvir, Architecture of Interoperable Information Systems (ARIS), Computer Integrated Manufacturing Open System Architecture (CIMOSA) i Generalized Enterprise Reference Architecture and Methodology (GERAM).

4.1.1. Pregled relevantnih poslovnih arhitektura

Zachman okvir koji je kreirao John Zachman, nosi naziv "*Okvir za arhitekturu informacionih sistema*", i jedan je od najpoznatijih okvira poslovnih arhitektura (J. Zachman, 2002, 2016). Svoju popularnost u velikoj meri duguje jednostavnosti i lakoj razumljivosti. Osnovu Zachman okvira čini dvodimenziona matrica, gde prvu

osu predstavljaju kolone koje se zovu *apstrakcije* (eng. abstractions) koje daju odgovor na sledeća pitanja o poslovnom sistemu: *šta* (podaci), *kako* (funkcije), *gde* (mreža), *ko* (ljudi), *kada* (vreme) i *zašto* (motivacija). Redovi se zovu *perspektive*, i svaki red predstavlja pogled koji je specifičan za jednu od interesnih grupa (eng. stakeholders). Svaka ćelija Zachmann-ove mreže definiše primitivni element i sadrži elementarni model arhitekture. Svaka ćelija može da reprezentuje jedan i samo jedan aspekt poslovnog sistema (Bente, Bombosch, & Langade, 2012). Svaki red reprezentuje različiti pogled (eng. view), odnosno jednu perspektivu određene interesne strane, na primer izvršna ili menadžment perspektiva. Zachman okvir može da se uporedi sa dobrom klasifikacionom šemom, i njegova struktura je čisto logička. Zachmanov okvir ne daje odgovor na rešenje problema, već predstavlja alat za razmišljanje (Sowa & Zachman, 1992).

Kao bitne karakteristike Zachman okvira izdvojićemo: *jednostavnost* (lak je za razumevanje i ne zahteva tehničko znanje); *razumljivost* (predstavlja poslovni sistem kao celinu); *jezik* (olakšava razmišljanje o kompleksnim pojmovima i njihovo precizno komuniciranje pomoću nekoliko termina koji nisu tehnički); *alat za planiranje* (doprinosi boljem donošenju odluka jer sporna pitanja mogu da se posmatraju u kontekstu čitavog poslovnog sistema uzimajući u obzir raspoložive alternative); *alat za rešavanje problema* (pruža mogućnost apstrakcije i izolovanja pojedinačnih promenljivih ne gubeći predstavu o poslovnom sistemu kao celini); *neutralnost* (u potpunosti je nezavistan od ostalih alata i metodologija) (J. A. Zachman, 1987).

Kao glavnu prednost Zachman okvira, sa aspekta interoperabilnosti istaćićemo kompletan skup koncepata i principa za modelovanje, dizajn i implementaciju poslovnih sistema na celovit način. Kao glavni nedostatak se ističe njegova nepogodnost za dizajn izvršnog nivoa, budući da u osnovi ima stratešku perspektivu i nedostatak operativnih informacija.

Osnovna karakteristika **ARIS okvira** je koncept pogleda (eng. view concept) (Scheer, 1992, 2013). Princip pogleda se bazira na ideji da poslovni sistem treba da se podeli na različite poglede, kako bi se lakše savladala njegova kompleksnost. ARIS

arhitektura predlaže definisanje pet različitih pogleda, koji su definisani na osnovu odgovarajućih dimenzija poslovnog sistema: organizacioni pogled, funkcionalni pogled, pogled podataka, pogled izlaza i kontrolni pogled. Za svaku dimenziju poslovnog sistema, definisana su tri nivoa granularnosti: definicija zahteva, specifikacija dizajna i opis implementacije. Navedeni nivoi odgovaraju CIM, PIM i PSM nivoima MDA pristupa respektivno (Scheer, Abolhassan, Jost, & Kirchmer, 2004). U poređenju sa ostalim okvirima, ARIS poseduje sistematičan pristup razvoju softvera koji je baziran na modelima. Posebna pažnja je posvećena specifikaciji zahteva, kojom su obuhvaćeni i zahtevi za interoperabilnošću.

Različiti pogledi ARIS arhitekture, podrazumevaju primenu različitih jezika za modelovanje. Na primer, za modelovanje kolaborativnih poslovnih procesa ARIS predlaže Event-Driven Process Chain (EPC) notaciju. Za potrebe modelovanja kolaborativnih poslovnih procesa su definisana odgovarajuća proširenja. Na primer, definisan je nov koncet uloge. Takođe je definisana odgovarajuća ekstenzija za podršku reprezentacije internih i eksternih pogleda. Sa aspekta interoperabilnosti između različitih alata za modelovanje, ARIS poseduje mehanizam za razmenu modela.

Pored detaljne analize Zachman i ARIS okvira, sagledane su i karakteristike GERAM, Graphs with Results and Actions Inter-related (Grai) (Doumeingts, Vallespir, & Chen, 1998), CIMOSA (ESPRIT Consortium AMICE, 1993) i The Open Group Architecture Framework (TOGAF) okvira.

4.1.2. Zaključna razmatranja

Poslovne arhitekture koje su uzete u razmatranje u ovom poglavlju, nude podršku za različite dimenzije poslovnog sistema. Za većinu arhitektura je karakteristično da imaju podršku za modelovanje na različitom nivou apstrakcije. Različite arhitekture predlažu da se sistem sagleda sa različitih nivoa. Nivoi koji se često uzimaju u razmatranje su: poslovni nivo, nivo informacija, aplikacioni i tehnološki nivo. Zajednička karakteristika svih navedenih okvira je da je glavni fokus da se obezbedi adekvatna podrška za reprezentaciju inter-organizacionih elemenata. Evidentno je da adekvatni koncepti za modelovanje kolaboracije između različitih poslovnih

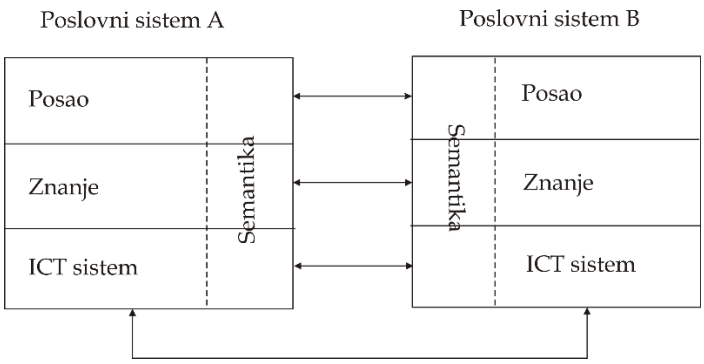
partnera nedostaju. Takođe, primarni fokus je na definisanju strateških pitanja i problema. Podrška za rešavanje operativnih pitanja nije adekvatna. Ne postoji adekvatna podrška za Model Driven Architecture (MDA) i SOA orijentisani razvoj.

4.2. Okviri za interoperabilnost

Dok poslovne arhitekture daju opsežan pregled intra-organizacionih elemenata, okviri koji će biti razmotreni u ovom poglavlju se koncentrišu na inter-organizacione poslovne procese na konceptualnom i tehničkom nivou.

4.2.1. IDEAS okvir

IDEAS okvir nosi naziv po projektu *Interoperability Development for Enterprise Application and Software* (IDEAS, IST-2001-37368) tokom kojeg je razvijen. IDEAS okvir koji je prikazan na slici 4.3 predstavlja konceptualni model interakcije između dva poslovna sistema na četiri različita nivoa (Blanc, 2005). Za uspešnu kooperaciju između različitih poslovnih partnera, je bitno ostvariti interoperabilnost na sva četiri nivoa (Arne-Jorgen Berre et al., 2004).



Slika 4.3 IDEAS okvir. Preuzeto i modifikovano prema (Arne-Jorgen Berre et al., 2004)

Poslovni nivo (eng. business level) sadrži opis poslovnog okruženja i poslovnih procesa, **nivo znanja** (eng. knowledge level) sadrži opis organizacionih uloga, veština i kompentencija radnika i ostale resurse bitne za sticanje znanja o poslovnim sistemu, dok **ICT sistem nivo** sadrži komunikacione komponente. Uvedena je i dodatna **semantička dimenzija** koja se odnosi na sve nivoe, i značajna je za postizanje međusobnog razumevanja između različitih poslovnih sistema koji žele da stupe u interakciju (Chen & Doumeingts, 2003). Nivoi IDEAS okvira su slični

poslovnom, nivou podataka, aplikacija i tehnologije koji su karakteristični za poslovne arhitekture koje su razmatrane u prethodnom poglavlju. Slično poslovnim arhitekturama, glavni nedostatak IDEAS okvira je činjenica da ne podržava interoperabilan razvoj vođen modelima, niti razliku između internih i eksternih pogleda.

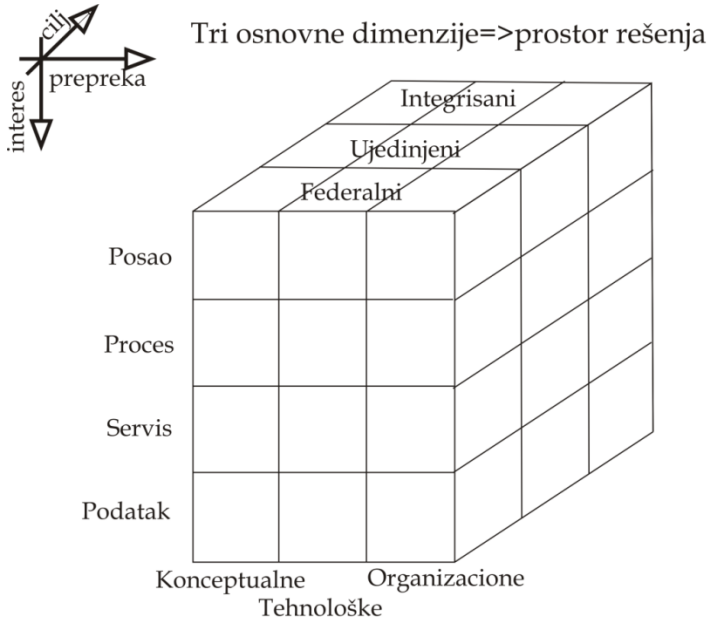
4.2.2. Enterprise Interoperability Framework (EIF)

EIF okvir ima tri osnovne dimenzije: nivoe interoperabilnosti, prepreke interoperabilosti i pristupe interoperabilnosti. Na slici 4.4 su prikazane dve osnovne dimenzije okvira: nivoi i barijere interoperabilnosti. U okviru EIF-a interoperabilnost se razmatra na četiri nivoa: podataka, servisa, procesa i poslovanja. Definisana su tri tipa prepreka: konceptualne, tehnološke i organizacione. Definicija nivoa interoperabilnosti i prepreka je data u drugom poglavlju teze.

Barijere / Nivoi	Konceptualne	Tehnološke	Organizacione
Posao			
Proces			
Servis			
Podatak			

Slika 4.4 EIF-dve osnovne dimenzije

Presek reda (*nivo interoperabilnosti*) i kolone (*prepreke interoperabilnosti*) definiše poddomen. EIF okvir definiše domen istraživanja kao skup poddomena od kojih je sačinjen. EIF definiše proširenje dvodimenzionalnog modela uvođenjem treće dimenzije koja određuje tri pristupa za rešavanje problema interoperabilnosti: integrisani, ujedinjeni i federalni (slika 4.5).

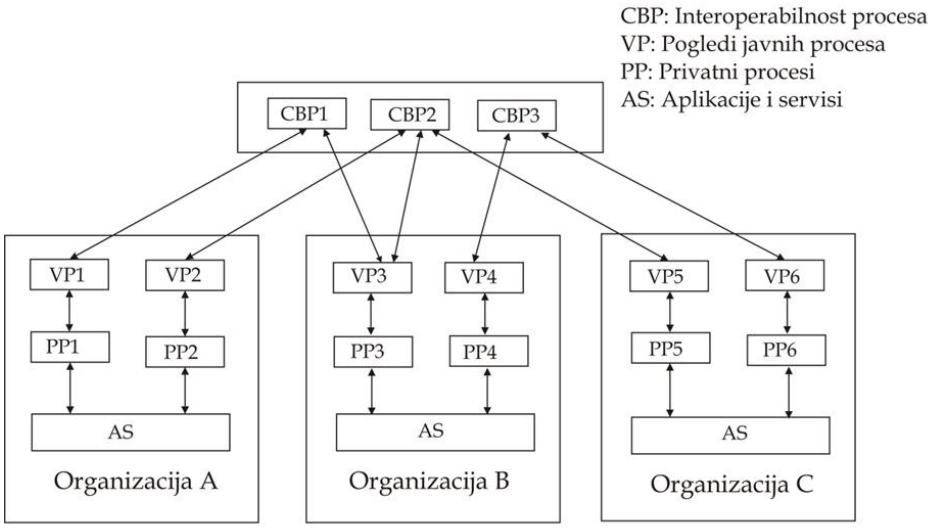


Slika 4.5 EIF okvir interoperabilnosti

EIF okvir interoperabilnosti pruža dobru osnovu za sistematizaciju osnovnih koncepata interoperabilnosti. Pri poređenju sa ostalim okvirima interoperabilnosti, bitno je naglasiti da jedino EIF definiše prepreke interoperabilnosti. Njegov osnovni princip je jedinstven, i zasnovan je na otklanjanju prepreka koje treba da budu jasno specificirane. EIF je standardizovan kao CEN/ISO 11354 standard.

4.2.3. ATHENA procesno-orijentisani okvir

U okviru ATHENA projekta su razvijena dva različita okvira: okvir koji se bavi problemima automatizacije kolaborativnih poslovnih procesa i okvir čiji je osnovni fokus na razvoju baziranom na modelima (eng. model-driven) u kontekstu SOA arhitekture koji će biti opisan u narednom poglavlju (slika 4.6) (ATHENA D.A2.1, 2005; ATHENA D.A4.2, 2007).



Slika 4.6 ATHENA procesno orjentisani okvir. Preuzeto i modifikovano prema (ATHENA D.A4.2, 2007)

ATHENA procesno-orijentisani okvir se fokusira na dimenziju procesa i koristi koncept javnih, privatnih i globalnih pogleda za automatizaciju kolaborativnih procesa. Procesno-orijentisani pristup je dizajniran sa ciljem da se obezbedi sistematičan način za selektivno prikazivanje internih informacija pri kolaboraciji između više partnera. Uvodi se poseban nivo pogleda procesa (eng. view process (VP)), kao dodatni nivo iznad nivoa privatnih procesa (PP). Nivo pogleda procesa (VP) se izvodi na osnovu privatnih procesa, apstrakcijom onih internih procesa koji po pravilu ne treba da budu izloženi prilikom kolaboracije. Pogledi procesa (VP) moraju da sadrže minimum informacija koje su dovoljne za uspešnu koordinaciju internih aktivnosti i odgovarajućih aktivnosti eksternih poslovnih partnera. Kao što je prikazano na slici 4.6 na osnovu jednog privatnog procesa, mogu da se izvedu različiti pogledi procesa (VP) u skladu sa potrebama kolaboracije sa različitim poslovnim parterima. Kolaborativni poslovni proces povezuje odgovarajuće poglede procesa (VP), u skladu sa potrebama specifičnog poslovnog scenarija, kao što je ilustrovano na slici 4.6. Bitno je naglasiti da jedan pogled procesa (VP) može da se koristi u različitim kros-organizacionim scenarijima (Cross-Organizational Business Proces (CBP)). Takođe, jedan kros-organizacioni scenario može da poveže poglede procesa (VP) jednog ili više poslovnih partnera. Teoretske osnove ATHENA procesno-orijentisanog okvira su u velikoj meri inspirisale specifikaciju privatnog i

javnog poslovnog procesa, odnosno treći korak faze identifikacije predloženog pristupa za specifikaciju aspekata interoperabilnosti.

Kao osnovne nedostatke ovog okvira izdvojićemo činjenicu da poseduje samo dimenziju procesa, dok ostale dimenzije kao što je organizaciona ili funkcionalna nisu podržane. Pristup nije fokusiran na definisanje potrebnih interfejsa za kolaboraciju, odnosno nije posvećena dovoljna pažnja definiciji javnog pogleda procesa.

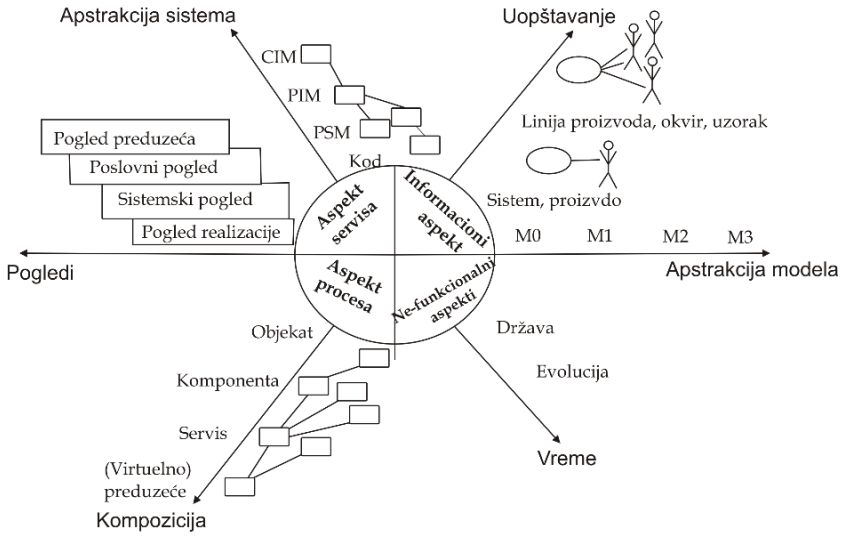
4.2.4. Model-Driven Interoperabilty (MDI) okvir

U oviru INTEROP projekta je razvijen MDI okvir koji je podeljen na tri dela: konceptualna integracija, tehnička integracija i aplikativna integracija (Arne-Jorgen Berre et al., 2004). **Konceptualna integracija** se fokusira na modele i veze između njih, **tehnička integracija** se fokusira na razvoj softvera i izvršno okruženje, dok se **aplikativna integracija** tiče standarda i domenskih modela.

Konceptualni pogled je razvijen sa MDD aspekta, sa fokusom na sistemske aspekte koji su definisani u skladu sa dimenzijama poslovnog sistema. Definisana su četiri sistemska aspekta: servisa, informacioni, procesni i nefunkcionalni. Sva četiri aspekta su usko povezana sa SOA konceptima, na primer aspekt servisa odgovara konceptu veb servisa, dok aspekt procesa omogućava realizaciju servisno-orijentisanih workflow-ova. INTEROP MDI okvir pored četiri osnovna aspekta, predlaže da se za potrebe harmonizacije CBP modela, uvede pet dodatnih dimenzija: apstrakcija sistema (eng. system abstraction), generička dimenzija (eng. genericity), tačka gledišta (eng. viewpoint), kompozicija (eng. composition), vreme (eng. time) i apstrakcija modela (eng. model abstraction). Predložene dimenzije i njihova relacija u odnosu na četiri osnovna aspekta su ilustrovane na slici 4.7.

Četiri predložena aspekta: procesa, servisa, informacija i nefunkcionalni aspekti ne predstavljaju konačnu listu aspekata koji su relevantni za modelovanje kolaborativnih poslovnih sistema. Iako je moguće identifikovati ostale aspekte, smatraju se dovoljno dobrom osnovom. Istraživanja sprovedena u okviru INTEROP

MDI i ATHENA projekata, su u velikoj meri uticala na izbor aspekata interoperabilnosti čija je specifikacija centralna tema disertacije.



Slika 4.7 Odnos integracionih dimenzija i sistemskih aspekata. Preuzeto i modifikovano prema (ATHENA D.A4.2, 2007)

Za konceptualni pogled je karakteristično da omogućava sagledavanje problema interoperabilnosti između dva različita poslovna sistema sa vertikalne i horizontalne perspektive. Dimenzija vertikalne interoperabilnosti se bavi rešavanjem internih problema interoperabilnosti, odnosno problema unutar jednog poslovnog sistema. Sa horizonalne tačke gledišta fokus je na rešavanju inter-organizacionih problema interoperabilnosti. Referentni model za konceptualnu integraciju se prioritetno bavi rešavanjem problema horizontalne interoperabilnosti. Definisan je niz paterna za rešavanje problema inter-organizacione interoperabilnosti, koji mogu da se primene i na rešavanje problema na nivou jednog poslovnog sistema. Mapiranja između različitih modela, mogu da se definišu koristeći meta-modele i ontologije.

4.2.5. Zaključna razmatranja

Analiza postojećih okvira interoperabilnosti je bila od koristi za sagledavanje bitnih aspekata koje je potrebno specificirati za uspešnu realizaciju interoperabilnosti informacionih sistema. Analizom je utvrđeno da većina okvira za interoperabilnost podrazumeva specifikaciju četiri osnovna sistemska aspekta: procesa, servisa, informacija i nefunkcionalnih aspekata. Iako je pokazano da je moguće identifikovati

i druge aspekte, za specifikaciju interoperabilnosti u disertaciji su odabrani: aspekt procesa, servisa i informacija. Nefunkcionalni aspekti koji se ne tiču interoperabilnosti, su van osega pristupa koji se predlaže u disertaciji, ali otvaraju interesantne mogućnosti za dalje istraživanje. Dok je podrška kolaborativnih okvira za reprezentaciju pogleda nad privatnim poslovnim procesima ograničena na aspekt procesa, u disertaciji se predlaže pristup koji predstavlja kombinaciju funkcionalnog i procesnog pogleda.

4.3. Servisno-orijentisani pristupi

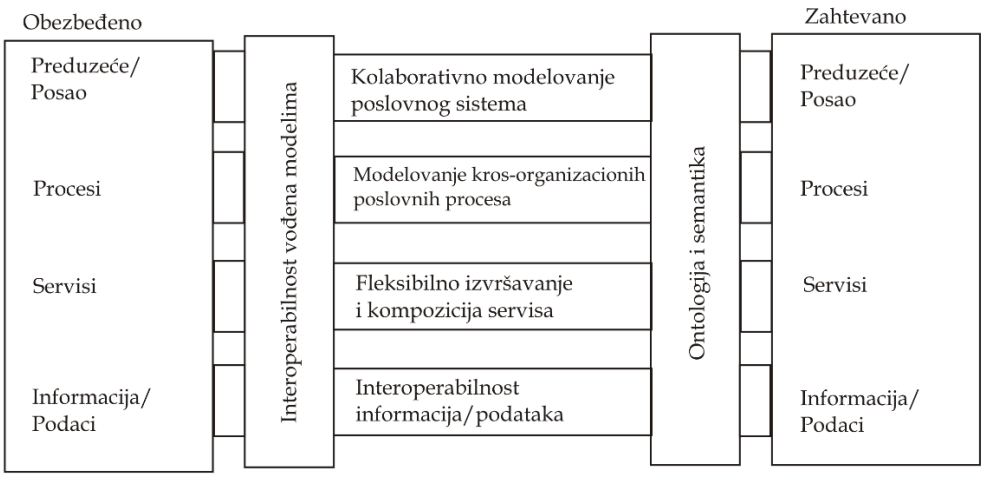
4.3.1. ATHENA Interoperability Framework (AIF)

Centralni okvir ATHENA projekta nije fokusiran na kolaborativne poslovne procese, već na razvoj vođen modelima u kontekstu SOA. SOA orijentisani AIF okvir je rezultat rada na poboljšanju i unapređenju IDEAS okvira. Dok se IDEAS okvir primarno fokusira na strukturiranje problema interoperabilnosti, osnovni cilj AIF okvira je sinteza rezultata istraživanja na ATHENA projektu kako bi se omogućilo pronalaženje adekvatnih rešenja u skladu sa poslovnim potrebama i tehničkim zahtevima. Glavna karakteristika ATHENA rešenja je da su vođena modelima (eng. model-driven). Fokusirana su na modelovanje interakcije i razmene informacija prilikom saradnje na poslovnom i na tehničkom nivou.

AIF je stukturiran u tri međusobno povezana dela (A.-J. Berre et al., 2007). **Konceptualna integracija** je deo koji se fokusira na koncepte, metamodele, jezike i relacije između modela. On obezbeđuje osnovu za sistematizaciju različitih aspekata interoperabilnosti softvera. **Aplikativana integracija** je deo koji se fokusira na metodologije, standarde i domenske modele. On obezbeđuje različita uptustva i softverska rešenja koja se mogu primeniti kako bi se rešila različita pitanja interoperabilnosti. **Tehnička integracija** se fokusira na razvoj softvera i različitih razvojnih okruženja. Obezbeđuje razvojne alate za razvoj softverskih modela i razvojna okruženja za njihovo izvršenje.

Referencijalni model za konceptualnu integraciju

Na slici 4.8 je prikazana pojednostavljena slika referencijalnog modela za konceptualnu integraciju.



Slika 4.8 Athena Interoperability Framework. Preuzeto i modifikovano prema (A.-J. Berre et al., 2007)

Potreba za saradnjom između poslovnih sistema je identifikovana na četiri različita nivoa koji su prikazani na slici 4.8. Interoperabilnost može da se posmatra posebno na svakom od nivoa.

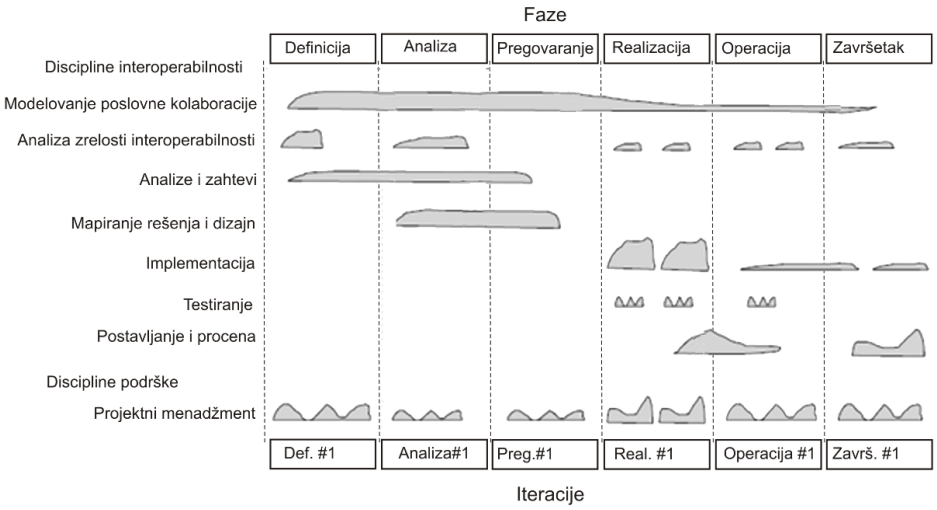
1) Interoperabilnost **informacija/podataka** omogućava zajednički rad razmenom heterogenih poruka i dokumenata između različitih entiteta koji učestvuju u kolaboraciji.
2) Interoperabilnost **servisa** omogućava identifikovanje, kompoziciju i zajedničko funkcionisanje različitih aplikacija, koje su dizajnirane i implementirane nezavisno. Fleksibilno modelovanje i izvršavanje servisa je podržano Platform-Independent Model for Service-Oriented Architecture (PIM4SOA) meta-modelom (ATHENA A6.3, 2006).
3) Interoperabilnost **procesa** ima za cilj da omogući zajednički rad različitih procesa. Proces definiše sekvencu servisa (funkcija) u skladu sa specifičnim potrebama poslovnih sistema. U distribuiranim poslovnim sistemima je bitno prethodno proučiti potrebne veze između procesa kako bi se omogućilo kreiranje zajedničkog procesa. Kolaborativno modelovanje poslovnih sistema je omogućeno Cross-organisatonal business process (CBP) meta-modelom (ATHENA A2.2, 2005).

4) Interoperabilnost **poslovnog sistema** se odnosi u prvom redu na harmonizaciju na organizacionom nivou, kako bi poslovanje između različitih partnera moglo da se odvija nesmetano (ne odnosi se na: komercijalni pristup, organizacionu kulturu, donošenje odluka i itd.). Kolaborativno modelovanje poslovnog sistema je podržano Process, Organisaton, Product and others (POP*) meta-modelom (ATHENA DA1.3.1, 2005).

Normalno na četri opisane dimenzije su postavljene dve dodatne dimenzije: dimenzija za razvoj vođen modelima (MDD) i dimenzija za opis semantike modela.

Referencijalni model za aplikativnu integraciju

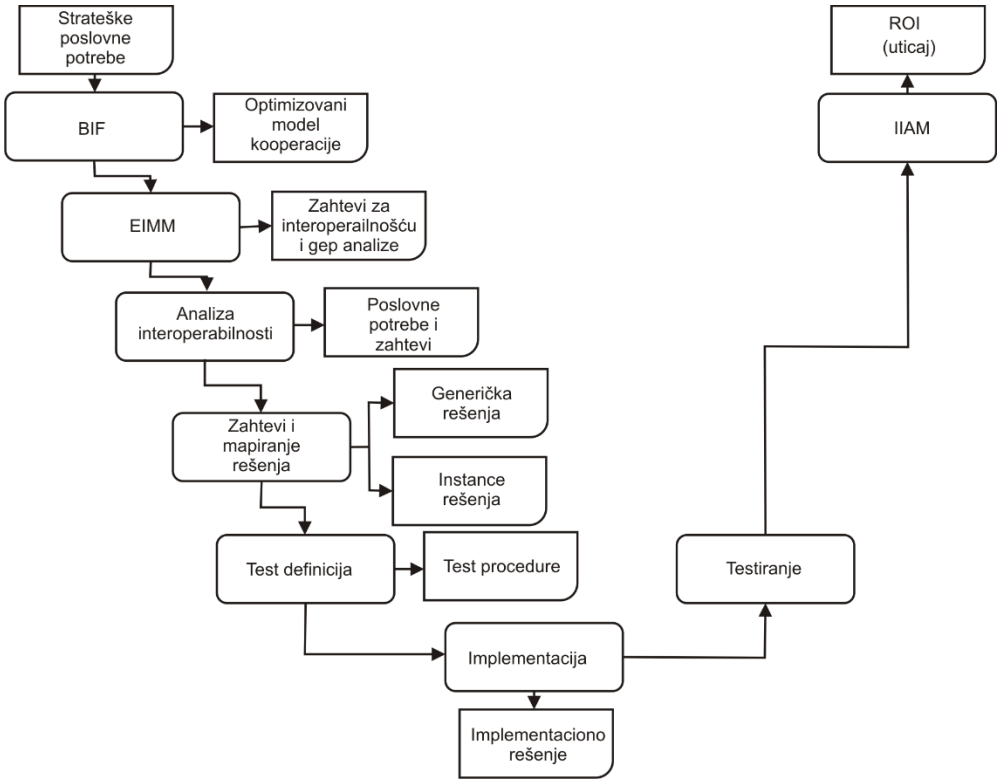
Referencijalni model za aplikativnu integraciju je nastao pod uticajem Enterprise Unified Process (EUP) (Ambler, Nalbone, & Vizdos, 2005; EUP, 2016). EUP proširuje koncepte Unified Software Development Process-a (UP) (Jacobson, Booch, & Rumbaugh, 1999). Za razliku od UP koji opisuje životni ciklus razvoja softverskog sistema, EUP proširuje opseg na opis životnog ciklusa čitavog informaciono-komunikaconog sistema. Prema (A.-J. Berre et al., 2007), disciplina interoperabilnosti predstavlja logički povezanu grupu aktivnosti koja pripada specifičnom polju interoperabilnosti. Osnovne faze i discipline AIF referentnog modela za konceptualnu integraciju su prikazane na slici 4.9.



Slika 4.9 Faze AIF referencijalnog modela za aplikativnu integraciju. Preuzeto i modifikovano prema (ATHENA D.A4.2, 2007)

AIF okvir za konceptualnu integraciju proširuje EUP uvođenjem novih disciplina interoperabilnosti. Faze životnog ciklusa projekta interoperabilnosti su prikazane u vidu kolona, dok su discipline interoperabilnosti predstavljene redovima na slici 4.9.

Bitna karakteristika referencijalnog modela za aplikativnu integraciju je **ATHENA Interoperability Methodology** (AIM). AIM integriše različite metode koje su razvijene u okviru ATHENA projekta. Ona se smatra osnovom koju treba konfigurisati ili proširiti u skladu sa specifičnim zahtevima konkretnog projekta. V model AIM metodologije je prikazan na slici 4.10.



Slika 4.10 AIM metodologija (A.-J. Berre et al., 2007)

Aktivnosti AIM metodologije podrazumevaju upotrebu sledećih metoda (A.-J. Berre et al., 2007):

- **Business Interoperability Framework (BIF)** predstavlja okvir za identifikaciju poslovnih zahteva u skladu sa strateškim poslovnim potrebama. Pomoću BIF okvira može da se odredi željeni nivo interoperabilnosti koji treba postići između partnera koji učestvuju u kolaboraciji.

- **Enterprise Interoperability Maturity Model (EIMM)** metod predstavlja skup oblasti koje su od interesa za kolaboraciju i nivoa „zrelosti“ (eng. maturity concerns) na osnovu kojih je moguće proceniti trenutnu spsobnost partnera da učestvuju u međusobnoj kolaboraciji. EIMM metod daje niz preporuka za poboljšanje trenutnog nivoa interoperabilnosti.
- **Analiza interoperabilnosti** predstavlja metod koji se fokusira na postizanje zajedničkog razumevanja bitnih elemenata kolaborativnog poslovnog procesa koji su ključni za postizanje interoperabilnosti na različitim nivoima. Na primer, razumevanje zajedničkih kolaborativnih poslovnih modela ili elemenata poruka koje se razmenjuju u kolaboraciji.
- **Mapiranje zahteva i rešenja** ima za cilj da pomogne različitim vrstama korisnika da pronađu potencijalna rešenja u skladu sa svojim specifičnim zahtevima.
- **ATHENA okvir za testiranje** uključuje aktivnosti definisanja testova i testiranja ostvarene interoperabilnosti i nivoa podobnosti (eng. conformance).
- **Implementacija** se vrši u skladu sa predloženim rešenjem u koraku četiri (mapiranje zahteva i rešenja). Ona podrazumeva implementaciju indvidualnih komponenti koje su definisane u skladu sa AIF okvirom za konceptualnu integraciju.
- **Interoperabilty Impact Analysis Method (IIAM)** omogućava procenu povratka investicija i uticaj mera interoperabilnosti.

Referencijalni model za tehničku integraciju

Referencijalni model za tehničku integraciju opisuje integrisanu arhitekturu za realizaciju interoperabilnosti kolaborativnih poslovnih sistema. Za podršku interoperabilnosti na **poslovnom nivou** je razvijena Modeling Platform for Collaborative Enterprises (MPCE) platforma, koja je bazirana na POP* meta-modelu. POP* meta-model definiše skup osnovnih koncepata za modelovanje poslovnih sistema. Može da se posmatra kao fleksibilan, posredni jezik koji ima za cilj da omogući razmenu između raličitih alata za modelovanje poslovnih sistema (ATHENA DA1.3.1, 2005). Skup alata za modelovanje kolaborativnh poslovnih

procesa čine: Maestro, Gabriel i Nehemiah, koji imaju mogućnost kreiranja privatnih kolaborativnih procesa i odgovarajućih pogleda nad njima. Za tehničku podršku na nivou servisa je implementiran ATHENA SOA okvir koji se čine tri komponente: (1) alati za modelovanje servisa i mapiranje između PIM4SOA i PSM modela, (2) Jonson alat za realizaciju servisa i (3) agenti za implementaciju potrebnih veb servisa u skladu sa specifičnim potrebama konkretnog projekta (ATHENA WD.B5.7.1, 2007).

4.3.2. Service-Oriented Analysis and Design (SOAD)

SOAD predstavlja iterdisciplinaran pristup koji je razvijen od strane (Zimmermann, Krogdahl, & Gee, 2004). Njihov pristup se bazira na ideji da objektno-orijentisani dizajn, poslovne arhitekture i modelovanje poslovnog sistema treba da budu integrisani u konherentan pristup za modelovanje servisa. Na primer, za modelovanje poslovnog sistema mogu da se koriste UML modeli a kao poslovna arhitektura može da bude izabran Zachman okvir. Glavni nedostatak predloženog pristupa se ogleda u činjenici da i pored integracije poslovnog nivoa, naglasak ostaje i dalje na tehničkom modelovanju. Nije posvećena dovoljna pažnja modelovanju poslovnih zahteva i ne postoji podrška za modelovanje kolaborativnih pogleda.

4.3.3. Zaključna razmatranja

U literaturi se kao osnovni nedostak servisno orijentisanih pristupa navodi nepostojanje adekvatne veze sa modelovanjem poslovnih zahteva. Prema (Ziemann, 2010), ne postoji adekvatan servisno-orijentisani pristup koji se zasniva na modelovanju poslovnog sistema. Problem je što modeli poslovnih sistema i specifikacija poslovnih zahteva još uvek imaju pretežno dokumentacioni karakter (Bastida, Berre, Elvesaeter, Hahn, & Johnses, 2009). Analizom servisno-orijentisanih okvira u ovom poglavlju su navedene pretpostavke opravdane. Iako se pridaje značaja vezi sa poslovnim nivoom, osnovni fokus su i dalje tehnički aspekti realizacije i implementacija poslovnih servisa.

4.4. Pregled relevantnih radova

Pristup koji je predložen u (Edward Barkmeyer & Denno, 2007), je relevantan za predmet istraživanja disertacije, s obzirom na to da se bazira na upotrebi zajedničke

referentne ontologije za specifikaciju potrebnih informacionih zahteva u kolaborativnom poslovnom procesu. Autori koriste termin *zajednički* (eng. joint) poslovni proces da označe pogled na aktivnosti poslovnog procesa u čiju je realizaciju uključeno više poslovnih partnera. Metodologija koja se predlaže u radu (Edward Barkmeyer & Denno, 2007) se sastoji iz tri glavne komponente: (1) globalne referentne ontologije, (2) formalne specifikacije zajedničkih poslovnih procesa primenom BPMN standardne notacije i (3) veze između aktivnosti BPMN poslovnog procesa i referentne ontologije. Autori u radu predlažu interesantno konceptualno rešenje koje je delom inspirisalo kreiranje UML View Profila i odgovarajuću ekstenziju BPMN meta-modela. Oni predlažu da se struktura poruke prikaže na BPMN dijagramu. Za potrebe grafičke reprezentacije poruke na dijagramu, autori definišu internu notaciju. U njihovom radu nije predložena formalna notacija za specifikaciju informacionih zahteva. Slično pristupu koji se predlaže u tezi, autori sugerišu da je potrebno kreirati pogled nad globalnom referentnom ontologijom. Oni definišu pogled kao podskup relevantnih svojstava entiteta referentne ontologije, ne uzimajući u obzir kompleksna pravila koja je potrebno specificirati za automatizaciju poslovnog procesa. Za razliku od njihovog pristupa, u tezi se predlaže definisanje OCL ograničenja.

Barnickel i saradnici su u radu (Barnickel, Böttcher, & Paschke, 2010), demonstrirali metodološki pristup za modelovanje poslovnih sistema koji korisiti semantičke mostove (eng. semantic bridges) za dizajn informacionih tokova. Oni obezbeđuju kompletno *end-to-end* rešenje od specifikacije do implementacije. Njihov pristup je baziran na semantičkim mostovima koji su primenjeni na entitete domenske ontologije, sa ciljem da se reši problem semantičke heterogenosti. Kako bi se obezbedilo bolje razumevanje i vizuelizacija predloženog rešenja, oni su izvršili ekstenziju BPMN Data Object kategorije koristeći semantičke podgrafove. U radu je pokazano da se za ontološku reprezentaciju informacionih tokova može koristiti Resource Description Framework (RDF) ili Web Description Language (OWL). Ipak, u radu se ne definiše mehanizam za implementaciju informacionih zahteva, kao delova entiteta referentne ontologije, već se koriste kompletni i nepromenjeni entiteti. Kao direktna posledica, javlja se potreba za semantičkom rekonsilijacijom

(eng. semantic reconciliation) različitih referentnih onotologija koje koriste različiti partneri. Pristup koji se prezentuje u radu se razlikuje od pristupa za specifikaciju pogleda referentne ontologije koji se predlaže u tezi. Naime, u disertaciji se predlaže upotreba zajedničke referentne ontologije, uz pretpostavku da partneri koji učestvuju u kolaboraciji koriste lokalne formalizovane ontološke modele koji se mapiraju na elemente globalne referentne ontologije.

Gao i saradnici predlažu ekstenziju BPMN 2.0 meta-modela primenom semantičkih ontologija (Gao, Derguech, & Zaremba, 2011). Oni iznose čvrst stav da BPMN 2.0 meta-model po ugledu na poglede ARIS arhitekture, treba da sadrži više detalja i mogućnosti za reprezentaciju funkcionalnog pogleda, pogleda podataka, organizaconog i kontrolnog pogleda. Za problem istraživanja koji se analizira u tezi je relevantan pogled podataka (eng. data view), za koji se predlaže ekstenizija BPMN 2.0 meta-modela koristeći *princip linkovanja podataka* (eng. link data principle). Autori predlažu ekstenziju *ItemAwareElements* kategorije elemenata, koja omogućava njihovu anotaciju konceptima domenski specifičnih ontologija koje su specificirane koristeći Resource Description Framework Schema (RDFS) ili OWL. Za razliku on njihovog pristupa, u tezi je odabrana UML grafička reprezentacija referentne ontologije. Kao glavni argument se ističe njena pogodnost za specifikaciju na konceptualnom nivou. Rešenje koje se predlaže u pristupu koji je opisan u (Gao et al., 2011) je prilagođeno specifičnim karakteristikama ARIS pogleda, dok je BPMN 2.0 ekstenzija koja se predlaže u tezi opštijeg karaktera.

5. SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI U IZABRANIM METODOLOŠKIM PRISTUPIMA

U ovom poglavlju su opisane faze jednog novog specifičnog pristupa za specifikaciju aspekata interoperabilnosti u metodološkim pristupima za razvoj informacionih sistema. Predloženi pristup je zasnovan na principima sistemsko-teorijskog modela životnog ciklusa softvera koji čine tri osnovne faze: identifikacija, realizacija i implementacija. Za svaku od faza su precizno definisani potrebni koraci i dat je skup preporuka za njihovu primenu. Na kraju poglavlja, su sumirani zahtevi za interoperabilnošću kolaborativnih poslovnih sistema, na osnovu kojih su definisani kriterijumi za validaciju predloženog pristupa.

5.1. Faze i koraci predloženog pristupa

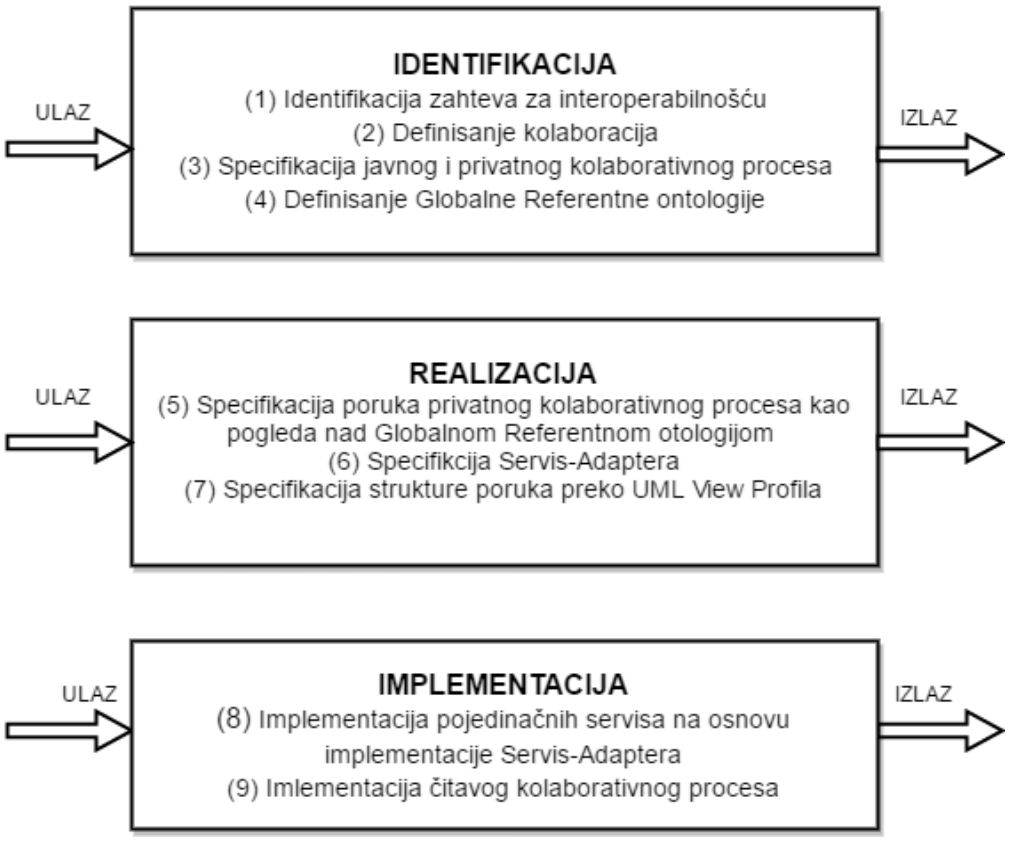
Za fazu identifikacije, realizacije i implementacije sistemsko-teorijskog pristupa su definisani potrebni koraci za specifikaciju aspekata interoperabilnosti, koji su prikazani na slici 5.2. Na osnovu detaljne analize aspekata interoperabilnosti koja je prikazana u drugom poglavlju (2.3 aspekti interoperabilnosti), a u skladu sa preporukama ATHENA referentnog modela za konceptualnu integraciju (ATHENA D.A4.2, 2007) za specifikaciju su odabrani aspekti: *procesa*, *servisa* i *podataka*.

U literaturi postoji veliki broj definicija *poslovnih procesa* i sve se u osnovi slažu da poslovni proces može da se definiše kao sekvenca poslovnih funkcija koja ima za cilj kreiranje izlaza za interne ili eksterne korisnike (Ziemann, Matheis, & Freiheit, 2007). U poglavlju (2.3.1) je ukazano na osnovne razlike između *intra* i *inter-organizacionih* poslovnih procesa. Dok se prva kategorija poslovnih procesa odnosi na aktivnosti koje se izvršavaju unutar jedne organizacije, *inter-organizacioni* poslovni procesi podrazumevaju kolaboraciju različitih organizacija na realizaciji zajedničkog cilja. S obzirom na to da je centralna tema disertacije specifikacija aspekata interoperabilnosti, od interesa su inter-organizacioni poslovni procesi koji se odvijaju između više poslovnih partnera. Ovi procesi se nazivaju još i *kros-*

organizacioni (eng. cros-organizational) poslovni procesi. Kros-organizacioni poslovni procesi predstavljaju specijalizaciju intra-organizacionih poslovnih procesa i poseduju niz specifičnih karakteristika kao što su na primer: potreba za skrivanjem informacija ili mogućnost prilagođavanja poslovnog procesa potrebama različitih poslovnih partnera (Lippe, Greiner, & Barros, 2005).

Aspekt *servisa* se u tezi posmatra kao prelazni koncept između koncepta poslovnog nivoa i novoa izvršenja. U disertaciji je usvojena definicija koja definiše servis kao softversku komponentu koja implementira poslovnu funkciju, i opisana je pomoću razumljivog, mašinski čitljivog interfejsa koji može da se pronađe i pozove na mreži (Ziemann, 2010). Za prevazilaženje problema interoperabilnosti sa *aspekta informacija* u tezi se predlaže upotreba zajedničke referentne ontologije.

Na slici 5.1 su prikazani koraci predloženog metodološkog pristupa u odnosu na opšte faze sistemsko-teorijskog životnog ciklusa razvoja IS.



Slika 5.1 Koraci predložene metodologije za specifikaciju aspekata interoperabilnosti u skladu sa fazama sistemsko-teorijskog životnog ciklusa

U fazi *identifikacije* se predlažu sledeći koraci:

1) Identifikacija zahteva za interoperabilnošću, odnosno identifikacija poslovnih partnera i polaznog skupa kolaborativnih poslovnih funkcija.
2) Definisanje kolaboracija između identifikovanih poslovnih partnera. Za definisanje kolaboracija na najvišem nivou apstrakcije se predlaže kreiranje BPMN dijagrama konverzacije i kolaboracije.
3) Specifikacija privatnog i javnog kolaborativnog procesa, kao mehanizma za zaštitu privatnosti.
4) Definisanje globalne referentne ontologije, kao mehanizma za nedvosmislenu interpretaciju značenja poruka koje se razmenjuju između različitih poslovnih partnera.

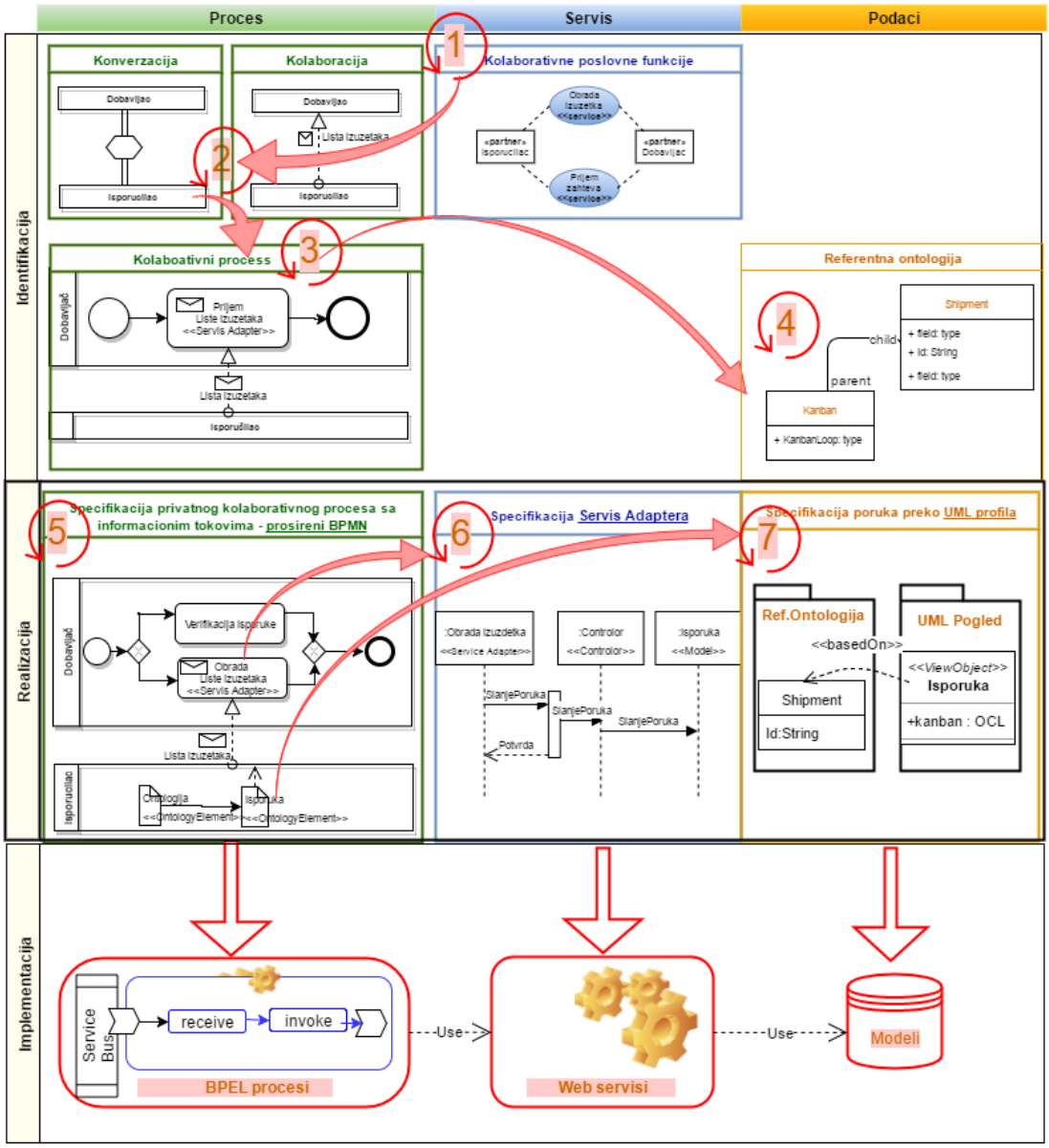
Nakon identifikacije kolaborativnih poslovnih procesa i odgovarajuće referentne ontologije, u fazi *realizacije* je potrebno obezbediti mehanizam za njihovu međusobnu vezu. Za detaljnu specifikaciju kolaborativnih poslovnih procesa koristeći koncepte referentne ontologije su definisani sledeći koraci:

5) Specifikacija poruka privatnog kolaborativnog procesa kao pogleda nad referentnom ontologijom.
6) Specifikacija servisa kolaborativnog poslovnog procesa primenom Servis Adapter paterna.
7) Specifikacija strukture poruka preko UML View Profila.

Za fazu implementacije su karakteristična dva koraka:

8) Implementacija pojedinačnih servisa kolaborativnog poslovnog procesa koji su specificirani preko Servis-Adapter paterna.
9) Implementacija orkestracije kolaborativnog poslovnog procesa (kompozicija pojedinačnih poslovnih servisa).

Osnovni koraci predloženog pristupa i tehnike koje se koriste za kreiranje odgovarajućih modela su prikazane na slici 5.2. Na primer, na slici 5.2 je moguće videti da drugi korak pripada fazi identifikacije, i da se za specifikaciju aspekta procesa koriste BPMN modeli konverzacije i kolaboracije.



Slika 5.2 Faze predloženog pristupa za specifikaciju interoperabilnosti

5.2. Identifikacija aspekata interoperabilnosti

Osnovni cilj faze identifikacije standardnih metodoloških pristupa je da se identifikuju i precizno opišu *esencijalni* ili *suštinski* procesi realnog sistema. Svaki suštinski proces se sastoji od skupa akcija kojim sistem reaguje na jedan i samo jedan događaj (SSA, 2000). Drugim rečima, skup akcija grupisanih u jednom suštinskom procesu mora biti takav da sistem, u idealnoj tehnologiji, po njihovom izvršenju prelazi u stanje mirovanja dok ne nastupi neki drugi događaj. Ukoliko se

koristi metoda Strukturne sistemske analize, esencijalni poslovni procesi predstavljaju poslovne procese koji se dalje ne dekomponuju i nazivaju se *primitivni poslovni procesi* ili *primitivne funkcije* (Lazarević et al., 2006; SSA, 2000).

Rezultat faze identifikacije je **logički model** sistema, kojim se definiše **„šta“** budući sistem treba da ponudi. Za definisanje modela funkcija poslovnog sistema može da se koristi bilo koja od konvencionalnih ili objektnih metoda za funkcionalnu specifikaciju (Lazarević & Nešković, 1998). U konvencionalnom pristupu razvoja IS identifikacija sistema se vrši primenom metode Strukturne sistemske analize (Lazarević et al., 2006), dok se u savremenom objektno-orijentisanom pristupu najčešće koristi Dijagram slučajeva korišćenja (Larman, 2004). Bez obzira na metodu za funkcionalnu specifikaciju sistema koja se koristi u fazi identifikacije, za primenu metodološkog pristupa koji se predlaže u tezi je bitno da se izvrši identifikacija polaznog skupa suštinskih procesa realnog sistema. Skup suštinskih poslovnih procesa predstavlja osnovu za analizu aspekata interoperabilnosti i identifikaciju zahteva za interoperabilnošću. Za uspešnu specifikaciju aspekata interoperabilnosti je ključno identifikovati kolaborativne, odnosno inter-organizacione poslovne procese na kojima radi više različitih partnera. Iz tog razloga se predlaže da se u fazi identifikacije, pored modela funkcija specificira i model procesa. Nakon identifikacije suštinskih procesa realnog sistema, za specifikaciju aspekata interoperabilnosti se predlažu sledeći koraci:

- **Korak 1**: Identifikacija zahteva za interoperabilnošću;
- **Korak 2**: Definisanje kolaboracija;
- **Korak 3**: Specifikacija privatnog i javnog kolaborativnog procesa;
- **Korak 4**: Definisanje globalne referentne ontologije.

5.2.1. Korak 1: Identifikacija zahteva za interoperabilnošću

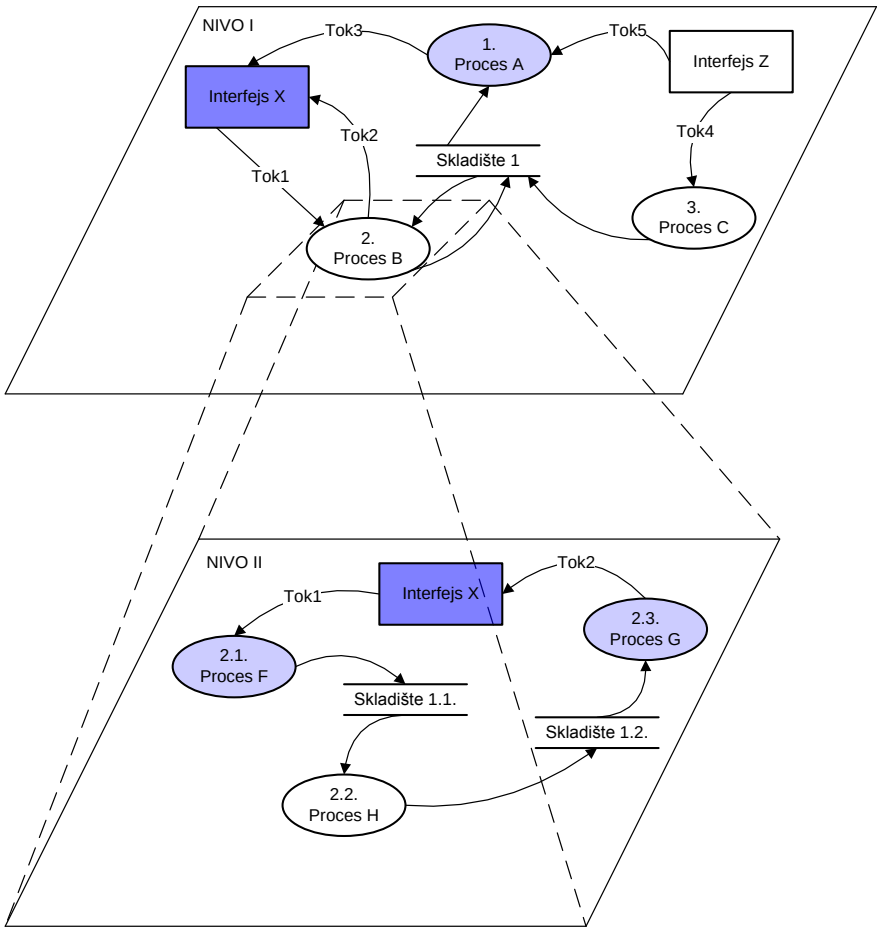
U prvom koraku faze *Identifikacije* sistema je potrebno identifikovati dve vrste zahteva za interoperabilnošću: (1) interoperabilne esencijalne poslovne funkcije i (2) partnere koji učestvuju u kolaboraciji. Na osnovu analize polaznog skupa esencijalnih poslovnih funkcija treba identifikovati one funkcije koje zahtevaju kolaboraciju između različitih partnera. Drugim rečima, treba doći do jedinstvenog

skupa esencijalnih poslovnih funkcija koje zahtevaju interoperabilnost, nevezano za specifične kolaborativne poslove kojima pripadaju. Nakon toga je potrebno identifikovati poslovne partnere koji učestvuju u kolaboraciji.

Zahtevi za interoperabilnošću mogu da se analiziraju na različitom nivou detalja. U prvom koraku je potrebno da se identifikuje koji esencijalni poslovni procesi zahtevaju interoperabilnost, bez ulaženja u detalje kako postići interoperabilnost. Kasnije u fazi realizacije se razmatraju moguće opcije i predlaže detaljan način specifikacije za realizaciju njihove interoperabilnosti.

Suštinski poslovni procesi koji zahtevaju interoperabilnost mogu da se identifikuju na više načina (SSA, 2000). Jedan pristup je direktnim modelovanjem, na osnovu znanja koje analitičar poseduje o realnom sistemu koji se modeluje. Drugi pristup se sastoji u primeni neke od konvencionalnih ili objektnih metoda za funkcionalnu specifikaciju sistema gde se u prvom koraku definiše kompletan logički model funkcija posmatranog sistema, a zatim iz dobijenog skupa esencijalnih poslovnih procesa identifikuju oni koji treba da budu interoperabilni. Različite tehnike za funkcionalnu specifikaciju sistema imaju različite pogodnosti za identifikaciju zahteva za interoperabilnošću.

Na primer, ukoliko se u fazi *identifikacije* sistema koristi Strukturna sistemska analiza, identifikacija zahteva za interoperabilnošću je u velikoj meri olakšana. Kandidate za suštinske poslovne procese koji treba da budu interoperabilni je moguće identifikovati na osnovu primitivnih poslovnih procesa koji komuniciraju sa spoljnim interfejsima sistema. Bitno je napomenuti da je komunikacija sa spoljnim interfejsom potreban, ali ne i dovoljan uslov da se izvede zaključak da za primitivan poslovni proces treba obezbediti realizaciju interoperabilnosti. Naime, neki od interfejsa sistema mogu da budu predstavljeni na dijagramu SSA samo radi sagledavanja šire slike i razumevanja kompleksnog poslovnog scenarija. Prema tome, za lakšu identifikaciju interoperabilnih primitivnih poslovnih procesa se predlaže da se prvo identifikuju partneri koji učestvuju u kolaboraciji. Nakon identifikacije partnera, moguće je selektovati one primitivne procese koji sa njima komuniciraju, i za njih zaključiti da treba da podrže aspekte interoperabilnosti.



Slika 5.3 Identifikacija zahteva za interoperabilnošću primenom metode SSA

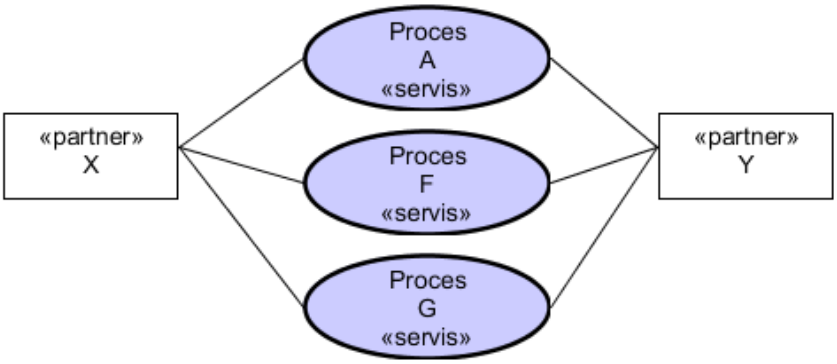
Na slici 5.3 je na prvom nivou dekompozicije prikazan poslovni proces koji se sastoji od dva primitivna procesa A i C, i jednog složenog procesa B. Složeni proces B je na drugom nivou dekompozicije dekomponovan na primitivne poslovne procese F, G i H. Pretpostavimo da je u prvom koraku na osnovu analize interfejsa, utvrđeno da je za kolaboraciju bitan *Interfejs X*. Nakon identifikacije partnera, u drugom koraku je utvrđeno da skup interoperabilnih zahteva čine primitivni procesi A, F i G.

Za specifikaciju zahteva za interoperabilnošću u fazi identifikacije se predlaže minimalna moguća modifikacija postojećih modela, koja se sastoji u označavanju partnera koji učestvuju u kolaboraciji i interoperabilih poslovnih procesa drugom bojom. U skladu sa datom preporukom, na dijagramu prikazanom na slici 5.3 *Interfejs X* i primitivni poslovni procesi A, F i G su označeni plavom bojom.

Pored uvođenja oznake na samom dijagramu, predlaže se i kreiranje dijagrama slučajeva korišćenja za reprezentaciju interoperabilnih primitivnih poslovnih

procesa. Na ovom dijagramu treba prikazati: (1) poslovne funkcije za koje je analizom utvrđeno da treba da budu interoperabilne i (2) identifikovane partnere koji učestvuju u kolaboraciji.

Za prikaz partnera je uveden poseban stereotip <<partner>>, dok se za označavanje primitivnih poslovnih procesa koji treba da budu interoperabilni koristi stereotip <<servis>>.



Slika 5.4 Dijagram interoperabilnih slučajeva korišćenja

Na slici 5.4 je prikazan dijagram interoperabilnih slučajeva korišćenja koji identifikuje tri interoperabilne poslovne funkcije A, F i G između poslovnih partnera X i Y.

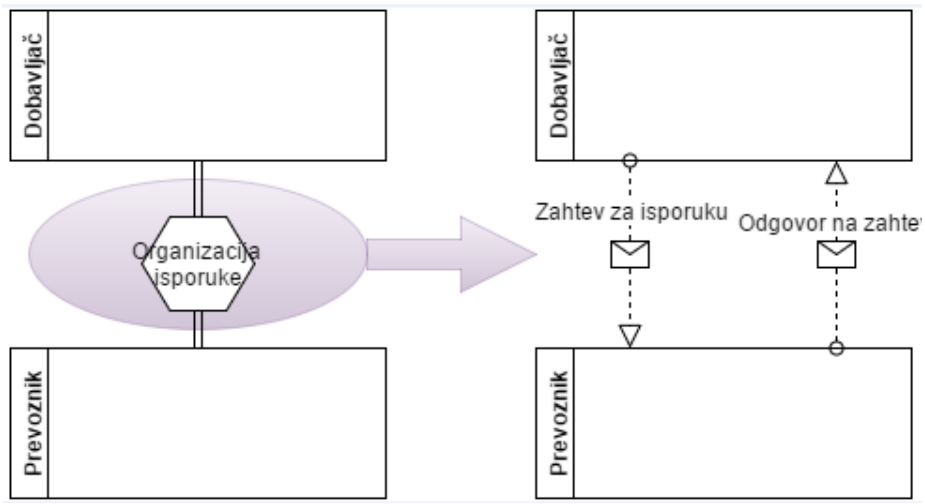
5.2.2. Korak 2: Definisanje kolaboracija

Nakon identifikacije zahteva za interoperabilnošću, potrebno je precizno definisati kolaboracije između poslovnih partnera koje se realizuju razmenom elektronskih poruka. S obzirom na to da su poslovni partneri identifikovani u prethodnom koraku, osnovni cilj ove faze je identifikacija poslovnih poruka koje se razmenjuju između partnera.

Kao pogodna tehnika za reprezentaciju kolaboracija odabrana je BPMN 2.0 notacija, koja poseduje tri različita dijagrama za opis interakcija (OMG BPMN 2.0, 2011): 1) *dijagrame konverzacije* koji pružaju mogućnost veoma apstraktne specifikacije, budući da se na njima identifikuju samo poslovni partneri i logički povezane grupe poruka koje se razmenjuju, 2) *dijagrame kolaboracije* koji su na srednjem nivou

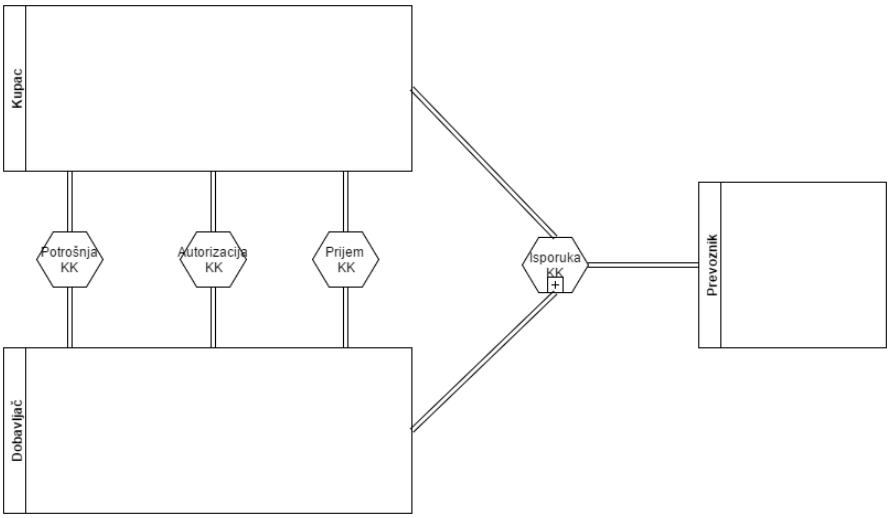
apstrakcije, jer se prikazuju pojedinačne poruke koje se razmenjuju između partnera i 3) *dijagrame koreografije* koji definišu detaljnu specifikaciju redosleda i uslova razmene poruka.

Za početnu identifikaciju kolaborativnih procesa se predlaže **dijagram konverzacije**. Na dijagramu konverzacije se predstavljaju partneri i definišu se logički povezane grupe poruka. U ovom koraku se predlaže da se partneri predstave po principu „crne kutije“ (eng. black box), kako bi se sva pažnja usmerila na identifikaciju poruka. Na slici 5.5 je prikazana konverzacija *Organizacija isporuke* između dva poslovna partnera: *Dobavljača* i *Prevoznika*. Sintaksa za označavanje konverzacije koja reprezentuje grupu poruka je šestougaonik sa duplim punim linijama ka partnerima, kao što je prikazano na slici 5.5.



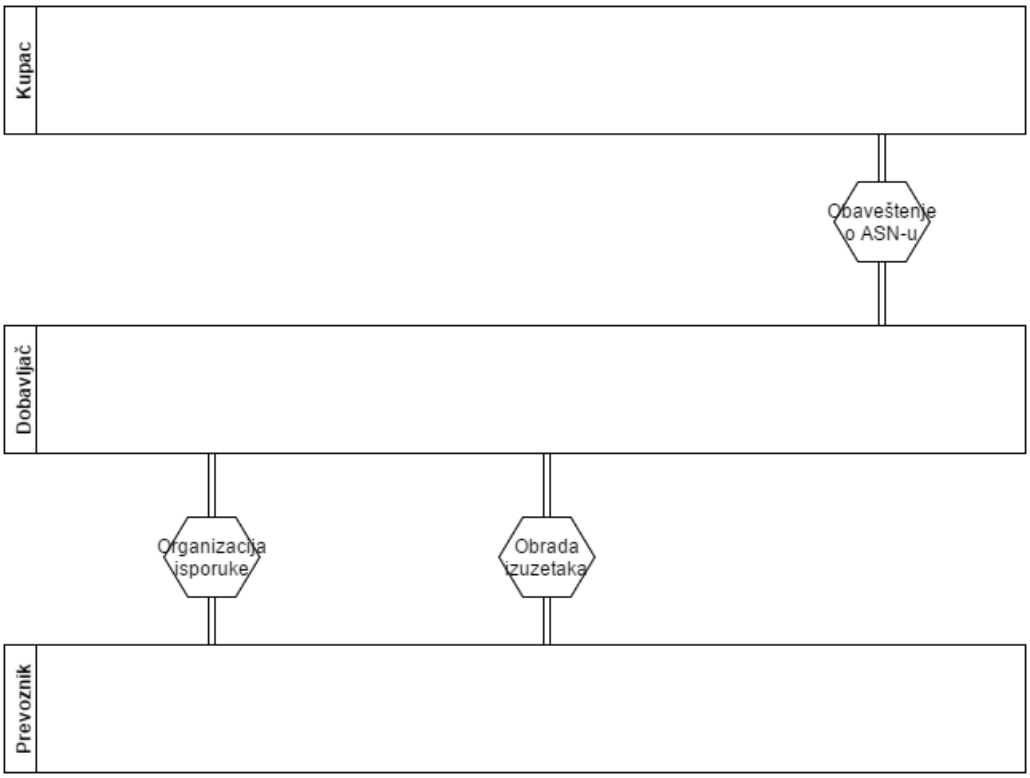
Slika 5.5 BPMN dijagram kolaboracije i konverzacije

Dijagrami konverzacije su pogodni za upravljanje komplikovanim interakcijama, budući da pored kompaktnog prikaza kolaboracije podržavaju i princip apstrakcije. Konverzacija na višem nivou može da se prikaže apstraktnije, i da se po potrebi dekomponuje (OMG BPMN 2.0, 2011). Na slici 5.6 je prikazan dijagram konverzacije koji pored tri standardne konverzacije (*Potrošnja Kanban Kontejnera*, *Autorizacija Kanban Kontejnera* i *Prijem Kanban Kontejnera*), sadrži i složenu konverzaciju *Isporuka Kanban Kontejnera*. Složena konverzacija *Isporuka Kanban Kontejnera* označava razmenu poruka između tri učesnika kolaborativnog procesa: *Kupca*, *Dobavljača* i *Prevoznika*.



Slika 5.6 Primer dijagrama konverzacije sa složenom konverzacijom Isporuka Kanban kontejnera

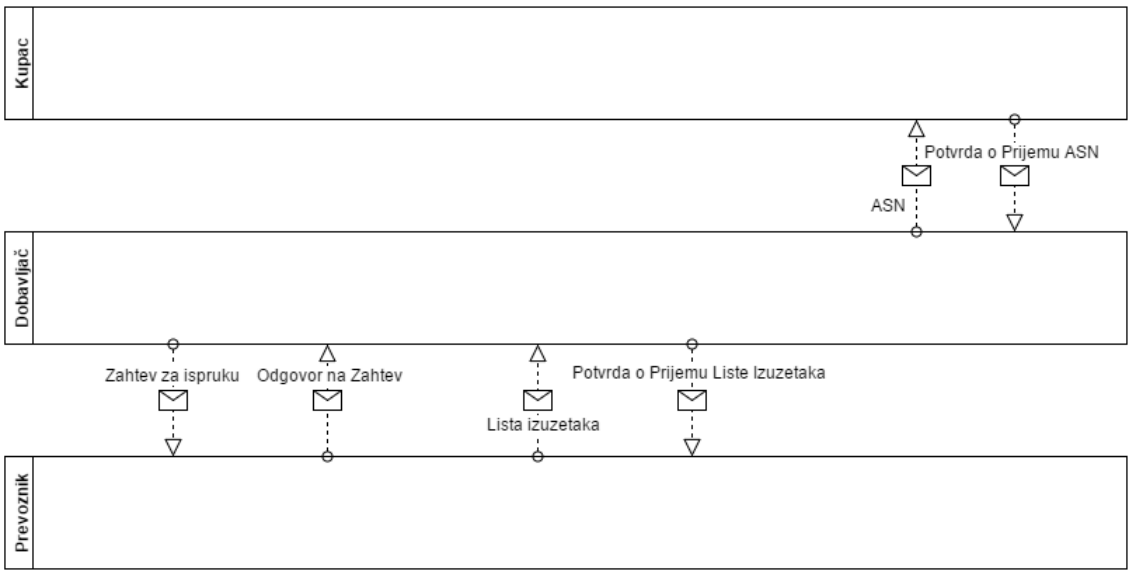
Na slici 5.7 je prikazana dekompozicija složene konverzacije *Isporuka Kanban Kontejnera* na tri standardne konverzacije: *Organizacija Isporuke*, *Obrada Izuzetaka* i *Obaveštenje o ASN*-u.



Slika 5.7 Dekompozicija složene konverzacije Isporuka Kanban kontejnera

Sledeći korak analize kolaborativnog poslovnog procesa podrazumeva identifikaciju pojedinačnih poruka koje se razmenjuju između poslovnih partnera, a koje su na

dijagramu konverzacije prikazane zbirno. Za modelovanje pojedinačnih poruka koje se razmenjuju između poslovnih partnera se predlaže upotreba dijagrama konverzacije. Na slici 5.8 je prikazan dijagram kolaboracije, koji je nastao kao rezultat razrade diJagrama konverzacije za *Isporuku Kanban Kontejnera* koji je prikazan na slici 5.7.



Slika 5.8 Dijagram konverzacije za kolaborativni proces Isporuka Kanban kontejnera

Iako se na dijagramu kolaboracije poruke pojavljuju u određenom redosledu kao što je pokazano na slici 5.8, BPMN notacija ne tumači ovaj redosled kao sekvencu. Na primer, bez obzira na redosled poruka na dijagramu 5.8, ne možemo da zaključimo da proces počinje tako što *Dobavljač* prvo pošalje *Prevozniku Zahtev za Isporuku*. Za specifikaciju sekvence izvršenja poruka je potrebno uključiti odgovarajuće aktivnosti. Naime, preporuka je da se sekvenca toka poruka između učesnika kolaboracije specificira pomoću odgovarajuće sekvence aktivnosti za slanje (eng. send) i prijem (eng. receive) poruka. O preporukama za definisanje sekvence poruka, odnosno preporukama za specifikaciju javnog i privatnog procesa biće više reči u narednom koraku.

5.2.3. Korak 3: Specifikacija privatnog i javnog procesa

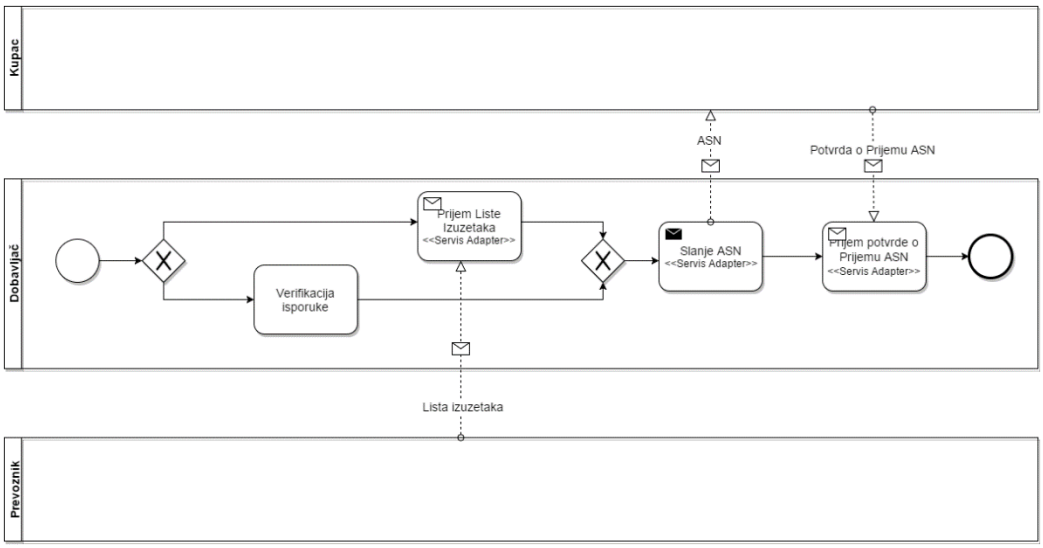
Ovaj korak podrazumeva detaljnije modelovanje, odnosno razradu dijagrama kolaboracije koji su definisani u drugom koraku. Nakon identifikacije partnera i poruka koje se razmenjuju pri kolaboraciji, u ovom koraku je fokus na modelovanju

poslovnih aktivnosti. Kolaborativni poslovni proces može da se posmatra kao sekvenca aktivnosti kojima se ostvaruje neki zajednički cilj partnera koji učestvuju u kolaboraciji. Aktivnosti kolaborativnog poslovnog procesa mogu biti složene ili osnovne, odnosno atomske. Sa tačke gledišta poslovnog procesa kao celine, osnovne aktivnosti su nedeljive. Aktivnosti po pravilu predstavljaju primitivne poslovne funkcije iz prethodno definisanog modela poslovnih funkcija (Lazarević & Nešković, 1998). U poslovnom procesu mogu da se jave i dodatne aktivnosti koje nisu deo funkcionalnog logičkog modela. Ove aktivnosti su posledica specifičnog organizacionog i tehnološkog okruženja u kojem se poslovni proces odvija.

Opisivanje dinamike kolaborativnog procesa se svodi na definisanje sekvence atomskih aktivnosti koje su prikazane na *Dijagramu interoperabilnih slučajeva korišćenja* i po potrebi uvođenje dodatnih aktivnosti. Jedan od osnovnih zahteva za modelovanje kolaborativnih poslovnih procesa se odnosi na zaštitu privatnosti (Lippe et al., 2005; Ziemann et al., 2007). Partneri koji učestvuju u kolaboraciji moraju da imaju mogućnost da prikažu interne informacije o privatnim podacima i procesima na različitom nivou apstrakcije. Generalno pravilo koje se preporučuje za uspešnu kolaboraciju između poslovnih partnera, je da model kolaborativnog poslovnog procesa treba da obezbedi minimum internih informacija koje su neophodne za zajedničko izvršenje procesa (ATHENA D.A2.1, 2005; Lippe et al., 2005; Ziemann, 2010; Ziemann et al., 2007). Kao mehanizam za zaštitu privatnosti u ovom koraku se predlaže specifikacija privatne i javne reprezentacije kolaborativnog poslovnog procesa.

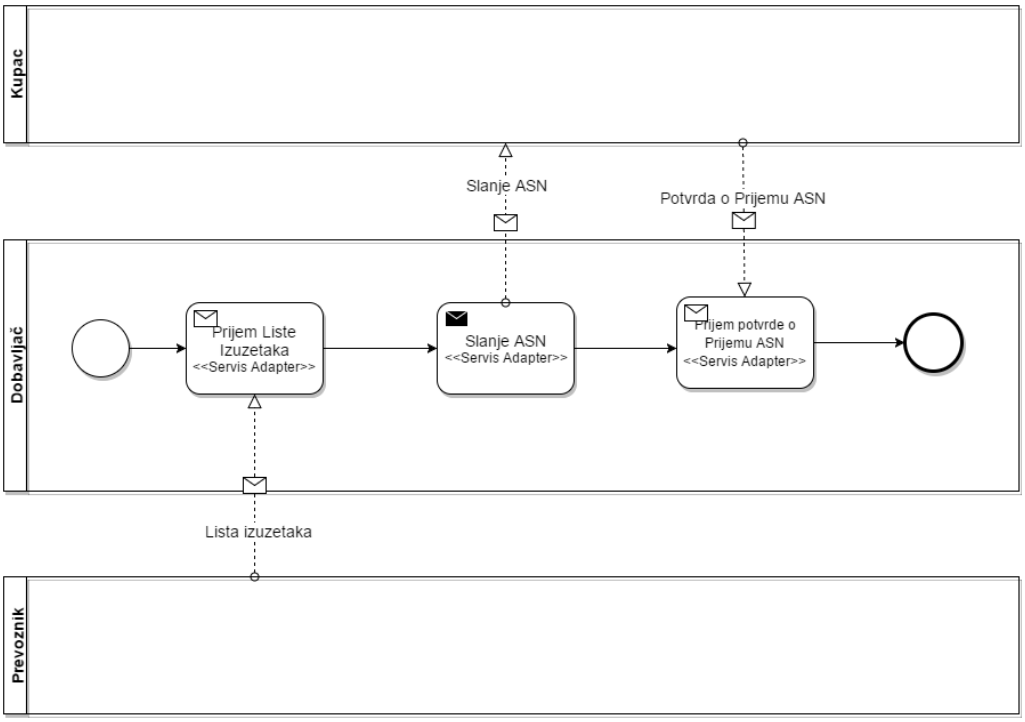
Privatni poslovni proces se kreira za internu upotrebu i predstavlja detaljan opis poslovne prakse, pa se iz tog razloga smatra intelektualnim vlasništvom. Sve aktivnosti privatnog procesa se izvršavaju interno. Bitno je naglasiti da se kolaboracija sa poslovnim partnerima obavlja isključivo razmenom poruka, i da privatni proces treba da poseduje interne aktivnosti koje služe samo za slanje ili prijem poruka. **Javni proces** može da se izvede iz privatnog procesa, apstrakcijom svih privatnih elemenata, kao što je objašnjeno u poglavlju (2.3.1) (Ziemann, 2010).

Na slici 5.9 je prikazan primer jednog privatnog kolaborativnog poslovnog procesa sa aspekta *Dobavljača*. Ostali partneri koji učestvuju u kolaboraciji su prikazani koristeći koncept „crne kutije“, budući da je analiza njihovog funkcionisanja van granica posmatranog sistema. Aktivnost *Verifikacija isporuke*, koja ukazuje na to da je prikazani poslovni proces privatnog karaktera, je tipičan primer aktivnosti koja nema uticaja na eksterno vidljivo ponašanje poslovnog procesa.



Slika 5.9 Primer privatnog kolaborativnog procesa

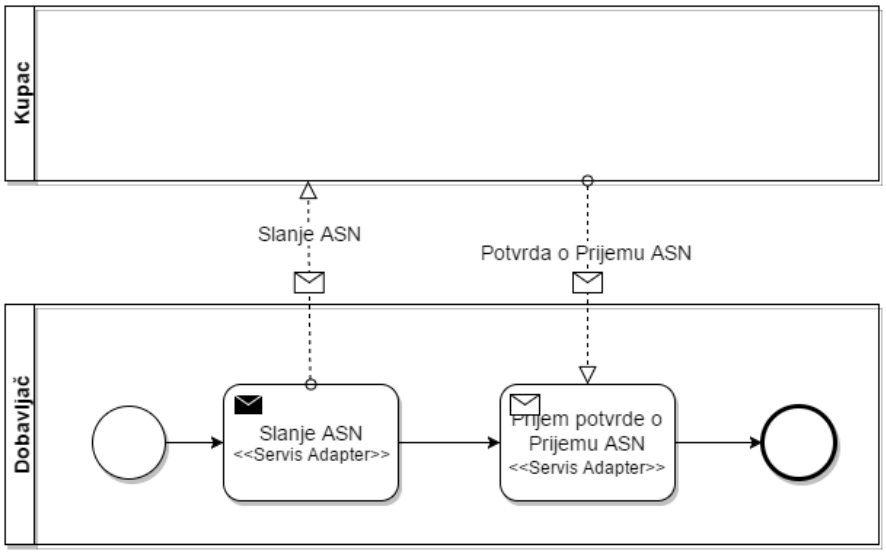
Na osnovu privatne definicije poslovnog procesa je potrebno kreirati njegovu javnu reprezentaciju. U tezi se predlaže da se za definisanje javnog kolaborativnog procesa primeni metoda apstrakcije**,** koja je detaljno objašnjena poglavlju (2.3.1). Metoda apstrakcije se sastoji u apstrahovanju svih poverljivih elemenata, tako da se javni proces sastoji uglavnom od aktivnosti slanja i primanja poruka koje se razmenjuju u interakciji. Neki od autora striktno ograničavaju definiciju javnog procesa na komunikacione aktivnosti privatnog procesa (Weske, 2012a). Ova striktna definicija javnog procesa je prihvatljiva za definisanje kolaborativnog poslovnog procesa na izvršnom nivou, ali sa aspekta poslovnog nivoa mogu da budu od koristi dodatne informacije. Na primer, neke od internih aktivnosti mogu da budu korisne za razumevanje kolaborativnog procesa, pogotovo ako je kolaboracija složena i učestvuje više partnera. Na slici 5.10 je prikazana moguća javna reprezentacija privatnog procesa sa slike 5.9.



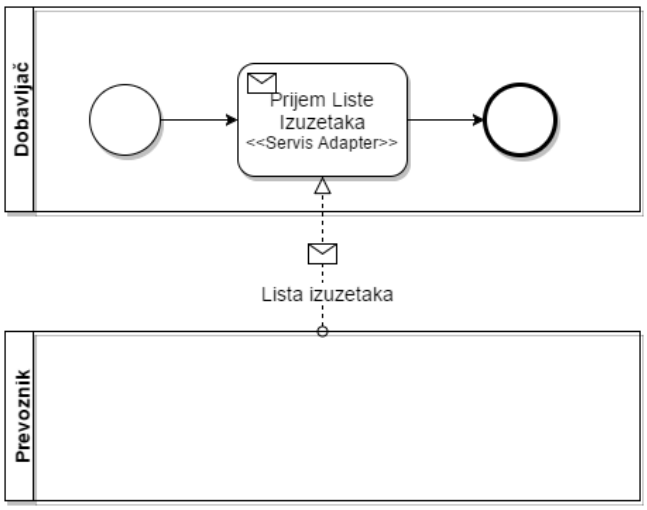
Slika 5.10 Primer javnog kolaborativnog procesa

Kao što se može videti na dijagramu kolaboracije koji je prikazan na slici 5.10, javni proces sadrži samo aktivnosti koje su potrebne za slanje i primanje poruka između partnera. Aktivnosti javnog poslovnog procesa su označene pomoću stereotipa <<Servis Adapter>>, koji označava da se za specifikaciju interoperabilnosti servisa predlaže Servis Adapter patern koji je detaljno objašnjen u šestom koraku.

Po pravilu, jedan privatni proces može da podrži više različitih reprezentacija javnih procesa. Veza kojom se iskazuje da privatni proces podržava (eng. *supports*) javni proces u BPMN-u nema posebnu notaciju. Različiti javni procesi mogu da prikazu različite aspekte privatnog procesa u zavisnosti od potreba kolaboracije sa specifičnim partnerom. Na primer, na osnovu definicije javnog procesa koji je prikazan na slici 5.10, mogu da se izvedu dva dodatna, različita javna procesa. Javna reprezentacija procesa koji je relevantan za komunikaciju sa *Kupcem* odnosno *Prevoznikom*, je prikazana na slikama 5.11 i 5.12 respektivno.



Slika 5.11 Kolaborativni javni proces sa Kupcem



Slika 5.12 Kolaborativni javni proces sa Prevoznikom

Na kraju ćemo sumirati preporuke za specifikaciju privatnog i javnog kolaborativnog poslovnog procesa. U ovom koraku se preporučuje da se prvo specificira privatni kolaborativni poslovni proces koristeći *Dijagram interoperabilnih slučajeva korišćenja*, pa da se zatim primenom metode apstrakcije kreira odgovarajuća javna reprezentacija. Kreiranjem javnog poslovnog procesa se postiže poštovanje principa skrivanja informacija, tako da kompleksnost realizacije privatnog procesa ne doprinosi povećanju kompleksnosti čitavog kolaborativnog poslovnog procesa. Sa poslovnog aspekta, na taj način se postiže zaštita privatnih procesa koji spadaju u bitan intelektualni kapital i potencijal za ostvarenje konkurentske prednosti.

5.2.4. Korak 4: Definisanje globalne Referentne Ontologije

U ovom poglavlju ćemo opisati upotrebu ontologija u kontekstu metodologije za specifikaciju aspekata interoperabilnosti koja se predlaže u tezi.

5.2.4.1. Određenje pojama Referentne Ontologije

Pojam *ontologija* dolazi iz oblasti filozofije, gde se definiše kao grana meta-fizike koja se bavi suštinom bića, postojanja i realnosti (Giaretta & Guarino, 1995; Guarino, 1997). Jedna od popularnih definicija, koja je često citirana u literaturi, definiše pojam ontologije kao "formalnu, eksplicitnu specifikaciju deljene konceptualizacije" (Gruber, 1995). Konceptualizacija predstavlja apstraktni, pojednostavljeni model domena koji želimo da predstavimo. Ona reprezentuje objekte (instance koncepata), koncepte, ostale entitete i relacije koje postoje između njih. Kao sredstvo za specifikaciju konceptualizacije Gruber predlaže ontologiju. Ontologije omogućavaju eksplicitnu specifikaciju konceptualizacije, budući da se za koncepte i njihove veze zadaju imena i definicije u apstraktnom modelu. Formalno predstavljanje konceptualizacije koristeći neki od ontoloških jezika ukazuje na mogućnost mašinskog procesiranja ontologija.

Guarino i Giaretta sugerišu da je Gruberova definicija ontologije suviše striktna i ukazuju na to da ontologija predstavlja samo "parcijalni deo konceptualizacije" (Giaretta & Guarino, 1995). Hepp opsežno diskutuje da li je ontologija konceptualni sistem ili njegova specifikacija (Hepp, 2008). Chandrasekaran i saradnici definišu ulogu ontologija u informacionim sistemima u radu (Chandrasekaran et al., 1999). Oni definišu ontologiju kao reprezentacioni vokabular, koji je specijalizovan za određeni domen ili predmet od interesa.

Ontologije se često klasifikuju prema svojoj opštosti (Gašević, Djurić, & Devedžić, 2006). Ontologije koje opisuju veoma generalne koncepte su poznate kao gornje ontologije (eng. upper ontology), top-level ontologije ili temeljne ontologije (eng. foundation ontology). Primeri top-level ontologija su dati u (Russell & Norvig, 2010; Sowa, 2000). Ontologije koje pored opštih koncepta, uključuju i one koncepte koji su specifični za određeni domen se nazivaju referentne ili domenske ontologije.

U disertaciji se koristi eKanban domenska ontologija koja reprezentuje specifične koncepte za realizaciju elektronskog Kanban kolaborativnog poslovnog procesa (E. Barkmeyer & Kulvatunyou, 2007). Naime, polazi se od pretpostavke da partneri pre početka bilo kog vida kolaboracije moraju da definišu zajedničku referentnu ontologiju, koja se koristi za formalnu specifikaciju osnovnih koncepata poslovnog domena. Značaj zajedničkog razumevanja važnih koncepata poslovnog domena i njegov uticaj na kolaboraciju je opširno diskutovan u (Kalfoglou, 2001). Bitno je naglasiti da zajednička referentna ontologija predstavlja znanje koje je prihvaćeno i usaglašeno od strane poslovnih partnera koji učestvuju u kolaboraciji. U primeru koji se koristi u radu, referentna ontologija je bitna jer obezbeđuje osnovu za nedvosmislenu interpretaciju značenja poruka koje se razmenjuju između partnera u kolaboraciji.

Kao što je diskutovano u (Giaretta & Guarino, 1995; Gómez-Pérez, 2004; Hepp, 2008) za konstrukciju referentne ontologije mogu da se koriste različiti jezici. U tezi se za grafičku reprezentaciju referentne ontologije koristi UML dijagram klasa. Bitni razlozi koji su uticali na odluku da se UML dijagram klasa izabere kao pogodna tehnika za grafičku reprezentaciju referentne ontologije su:

- UML dijagram klasa podržava vizuelnu reprezentaciju bitnih elemenata ontologije. Na primer, moguće je predstaviti hijerarhiju klasa i podklasa, veze između klasa i bitne atribute;
- UML je otvoren standard i poseduje standardan mehanizam za definisanje proširenja, na primer mehanizam za kreiranje Profila;
- OCL je moćan mehanizam za definisanje dodatnih ograničenja, na primer vrednosti atributa i moguće instance veza;
- UML standard je opšte prihvaćen u industriji i ima široku mrežu korisnika. Na primer, UML se najčešće koristi za vizuelnu reprezentaciju integracionih standarda koji su bazirani na XML Schema standardu (OAGIS, RosettaNet, Universal Business Language (UBL)).
- Osnovni UML koncepti se mapiraju u odgovarajće OWL koncepte, kao što je definisano u (OMG, 2014).

Upotreba UML dijagrama klasa za reprezentaciju ontologija, bilo direktno, bilo za potrebe grafičke vizuelizacije onih onoloških jezika koji nemaju ovu mogućnost, je česta u praksi (Baclawski et al., 2002; Kogut et al., 2002). Baclawski i saradnici su u svom radu prezentovali alat za razvoj ontologija baziranih na UML-u (Baclawski et al., 2002). Autori (Baclawski et al., 2002; Kogut et al., 2002) ističu da UML nije pogodan samo za vizuelizciju kompleksnih ontologija, već i za upravljanje njihovim razvojem. Cranfield i Purvis su detaljno istražili mogućnost primene UML dijagrama klasa za reprezentaciju ontologija i rezultate sistematično prikazali u (Cranefield & Purvis, 1999; Kogut et al., 2002).

5.2.4.2. Razvoj Referentne Ontologije

Bitan preduslov uspešne kolaboracije između različitih partnera je dogovor oko izbora postojeće ili definisanja nove globalne referentne ontologije. Definisanje referentne ontologije podrazumeva preciziranje svih potrebnih informacija o konceptima poslovnog domena, kao i njihovim dozvoljenim relacijama. U tezi nije kreirana nova referentna ontologija, već je upotrebljena postojeća eKanban referentna ontologija koja je kreirana za potrebe ATHENA projekta (E. Barkmeyer & Kulvatunyou, 2007). Preporuke za kreiranje nove referentne ontologije, neće biti diskutovane u tezi budući da je literatura o ovoj temi obimna (Giaretta & Guarino, 1995; Gómez-Pérez, 2004; Gruber, 1995; Hepp, 2008).

5.3. Realizacija aspekata interoperabilnosti

Za specifikaciju aspekata interoperabilnosti procesa, servisa i informacija u fazi identifikacije je pokazano da je moguće primeniti postojeće modele i tehnike bez potrebe za definisanjem značajnijih proširenja. Na primer, pokazano je da je odabrana BPMN tehnika pogodna za reprezentaciju kolaborativnih poslovnih procesa na različitim nivoima apstrakcije: počev od opštih dijagrama konverzacije do mogućnosti definisanja privatnog i javnog kolaborativnog procesa. Do problema dolazi u fazi *realizacije*, koja treba da obezbedi detaljnu specifikaciju identifikovanih aspekata interoperabilnosti. Imajući u vidu ograničenja BPMN informacione perspektive, jasno je da je neophodno definisati proširenje BPMN notacije za

adekvatnu reprezentaciju veze između kolaborativnih poslovnih procesa i referentne ontologije.

Predložena metodologija za specifikaciju aspekata interoperabilnosti u fazi realizacije definiše sledeće korake:

- **Korak 5:** Specifikacija poruka privatnog kolaborativnog procesa kao pogleda nad referentnom ontologijom;
- **Korak 6:** Specifikacija servisa kolaborativnog procesa primenom Servis Adapter paterna;
- **Korak 7:** Specifikacija strukture poruka preko UML View Profila.

5.3.1. Korak 5: Specifikacija informacionih tokova kolaborativnog procesa

U ovom poglavlju su u prvom delu date opšte teorijske osnove koje su bitne za razumevanje predloženog pristupa za ekstenziju BPMN meta-modela. Nakon toga su obrazloženi razlozi zbog kojih se smatra da je ekstenzija neophodna i objašnjen je način njene realizacije. Način primene predložene formalizacije za specifikaciju informacionih tokova kolaborativnog procesa je ilustrovan na primeru.

5.3.1.1. Teorijske osnove pristupa

5.3.1.1.1. BPMN informaciona perspektiva

BPMN je usvojen kao jedan od industrijskih standarda za modelovanje kolaborativnih poslovnih procesa (OMG BPMN 2.0, 2011). Kao značajna prednost BPMN notacije u odnosu na ostale jezike za modelovanje poslovnih procesa ističe se sposobnost reprezentacije četiri bitne perspektive modelovanja procesa: (1) funkcionalne (koje aktivnosti se izvršavaju); (2) bihevioralne (kada i kako se izvršavaju aktivnosti); (3) organizacione (gde i ko izvršava aktivnosti) i informacione (informacioni entiteti/podaci koji se koriste ili su rezultat izvršenja procesa) (Curtis, Kellner, & Over, 1992; OMG BPMN 2.0, 2011).

Sa aspekta modelovanja kolaborativnih poslovnih procesa, BPMN 2.0 ima adekvatnu podršku za modelovanje funkcionalne, bihevioralne i organizacione perspektive

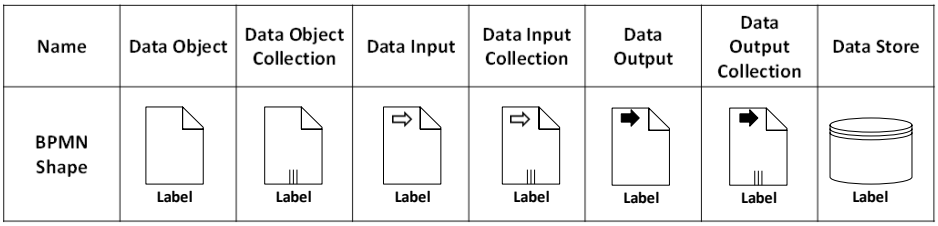
(Jankovic, Ljubicic, Anicic, & Marjanovic, 2015). Na primer, BPMN 2.0 notacija ima mogućnost apstrakcije procesa, kao i detaljne specifikacije sekvence izvršenja međusobno povezanih aktivnosti. Osnovni problem se ogleda u tome što ne postoji adekvatna podrška za reprezentaciju informacionog toka. Za uspešno modelovanje kolaboracije je pored detaljne specifikacije sekvence izvršenja poslovih procesa, potrebno definisati i pridružiti aktivnostima detaljne informacione zahteve.

U BPMN 1.x verzijama standarda nije bilo moguće definisati semantiku za informacione elemente kao što su podaci ili tokovi podataka. Ovi elementi su klasifikovani kao artifakti, na primer jednostavne anotacije dijagrama. U BPMN 2.0 verziji standarda, podaci su unapređeni u promenljive procesa (eng. process variable). Iako je podacima na ovaj način pridružena semantička dimenzija, nedostatak se i dalje ogleda u činjenici da je samo mali deo informacija koje mogu da budu specificirane semantičkim modelom vidljiv na dijagramu, na primer: labele, ikona za reprezentaciju podataka ili skladišta podataka. Umesto adekvatne grafičke reprezentacije, BPMN 2.0 standard označava XML Schema-u kao pogodno sredstvo za definisanje strukture podataka.

U radu (Jankovic et al., 2015) se ukazuje na to da bi vizuelni opis strukture i relacija između elemenata informacionog modela, trebalo da bude integralni deo informacione perspektive modelovanja poslovnih procesa. BPMN 2.0 notacija ne pruža ovu mogućnost, odnosno nije predviđena za modelovanje podataka i informacija (OMG BPMN 2.0, 2011). Kako bi se omogućila veća fleksibilnost pri modelovanju, tok informacionih entiteta (na primer podaci, artifakti, proizvodi) između elemenata procesa je razdvojen od toka sekvence aktivnosti (eng. sequence flow) (Jakob Freund, Bernd Rucker, 2012; Miers, 2008).

Primarna konstrukcija za modelovanje svih vrsta informacionih entiteta bez obzira na njihovu fizičku prirodu (papirni ili elektronski dokumenti) u BPMN-u je *Data Object* (OMG BPMN 2.0, 2011). U BPMN 2.0 verziji standarda za modelovanje podataka je uvedena nova kategorija za podatke (eng. *Data Category*), koja reprezentuje podatke kao semantičke elemente. Uvođenje posebne kategorije podataka predstavlja veliki pomak, s obzirom na to da su u verziji BPMN 1.2 podaci

predstavljeni kao artifakti, odnosno jednostavne anotacije dijagrama bez dodatne semantike. Bruice Silver u svojoj knjizi (Silver, 2011) posmatra podatke kao programske konstrukcije, koje su privremeno smeštene u instanci procesa. *Data Object* elementi su vizuelno reprezentovani na dijagramu pomoću simbola koji su prikazani na slici 5.13.



Slika 5.13 Elementi kategorije podataka. Preuzeto i prilagođeno prema (OMG BPMN 2.0, 2011)

Elementi kategorije podataka mogu da budu referencirani pomoću *DataObjectReference* koja specificira različita stanja istog *DataObject*-a, na primer *<DataObject Name>[DataObjectReferenceState]*. Struktura *DataObject*-a nije vidljiva na dijagramu, ali može da se definiše pomoću pridruženog *itemDefinition* elementa koji specificira *XML Schema-u*.

Za reprezentaciju perzistentnih podataka BPMN 2.0 uvodi novi koncept – skladište podataka (eng. data store). Dodatni elementi kategorije podataka su: *Data Inputs*, *Data Outputs* i *Properties*. Kolekcije *Data Object*-a, *Data Input*-a, *Data Output*-a su reprezentovane pomoću *Data Object Collection*, *Data Input Collection* i *Data Output Collection* simbola prikazanih na slici 5.13 respektivno. *Property* elementi nemaju mogućnost vizuelne reprezentacije na dijagramu, i relevantni su za izvršenje procesa.

BPMN 2.0 standard uvodi posebnu vrstu asocijacije–*Data Association*. Obična ne-direkciona asocijacija se koristi za povezivanje tekstualnih anotacija sa relevantnim elementima poslovnog procesa. Veza između elemenata kategorije podataka i ostalih elemenata modela (na primer aktivnosti ili događaji) se predstavlja pomoću *Data Association* elementa. BPMN 2.0 uvodi novi simbol, kovertu, za reprezentaciju informacionog sadržaja poruka koje se razmenjuju između različitih poslovnih partnera u kolaborativnom procesu.

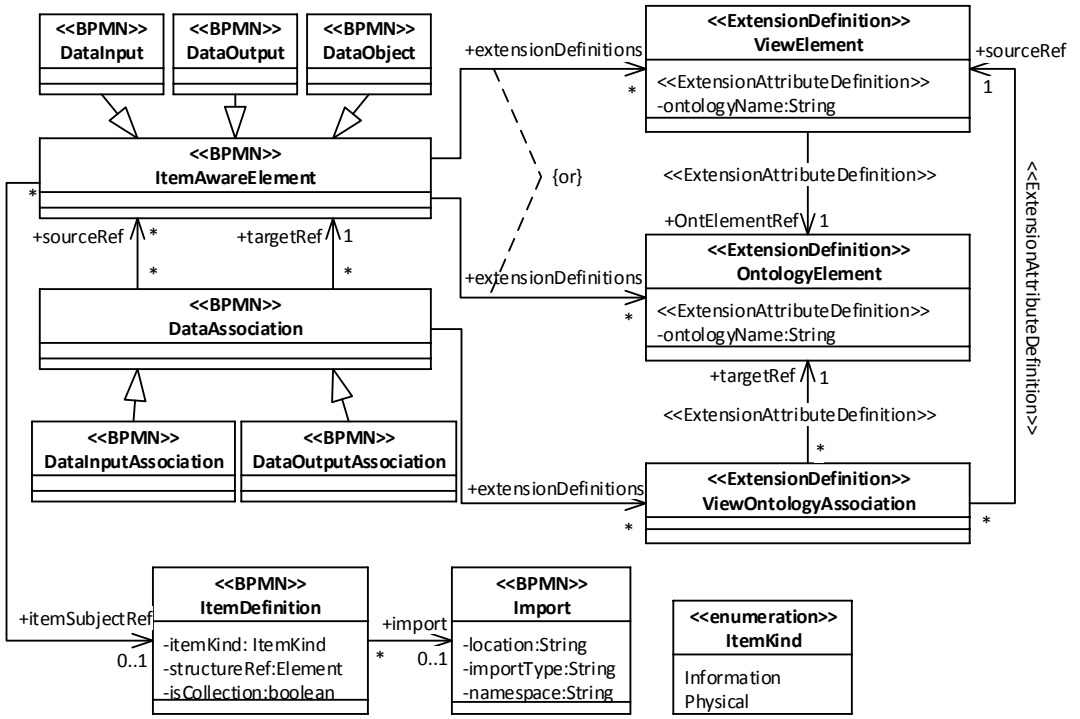
5.3.1.1.2. Mehanizam za ekstenziju BPMN –a

BPMN 2.0 sadrži standardni mehanizam za proširenje, koji omogućava definisanje novih koncepata sa potrebnom semantikom. BPMN meta-model je dizajniran sa ciljem da bude lako proširljiv. BPMN mehanizam za proširenje se sastoji od skupa elemenata za proširenje, pomoću kojih je standardnim BPMN elementima moguće dodati nove elemente i atribute. BPMN mehanizma za proširenje čine četiri osnovna elementa (OMG BPMN 2.0, 2011): *ExtensionDefinition*, *ExtensionAttributeDefinition*, *ExtensionAttributeValue* i *Extension*.

Extension element se koristi za povezivanje BPMN modela sa ekstenzijom čija je struktura definisana koristeći *ExtensionDefinition* element. *ExtensionDefinition* element grupiše dodatne atribute koji se koriste za ekstenziju BPMN modela, i poseduje mehanizam za njihovo povezivanje sa bilo kojim BPMN elementom. Za definiciju atributa, koja podrazumeva njegovo ime i tip, koristi se *ExtensionAttributeDefinition* element. Vrednost atributa koji je dodat ekstenzijom BPMN-a može da se specificira pomoću *ExtensionAttributeValue* elementa.

5.3.1.2. Ekstenzija BPMN-a za formalizaciju informacionih tokova kolaborativnih procesa

Ekstenzija BPMN meta-modela koja je prikazana na slici 5.14, je kreirana koristeći BPMN mehanizam za ekstenziju čiji su koncepti ukratko objašnjeni u prethodnom poglavlju.



Slika 5.14 Ekstenzija BPMN modela (Marija Jankovic, Ljubicic, Anicic, & Marjanovic, 2015)

Predložena ekstenzija BPMN meta-modela omogućava da se u definiciju dijagrama BPMN procesa uključe:

- Definicija dokumenta koji reprezentuje model referentne ontologije
- Definicija pogleda nad referentnom ontologijom.

Za definisanje strukture predložene ekstenzije su korišćeni elementi: *ExtensionDefinition* i *ExtensionAttributeDefinition*. Ova dva elementa su na slici 5.14. prikazana odgovarajućim stereotipima, koji takođe nose naziv *<<ExtensionDefinition>>* i *<<ExtensonAttributeDefinition>>*. Originalni BPMN elementi koji nisu proširenи su označeni stereotipom *<<BPMN>>.*

BPMN meta-model elementi koji su relevantni za povezivanje sa *modelom dokumenta ontologije* ili *modelom pogleda* su: *DataObject*, *DataInput* i *DataOutput*. Oni predstavljaju podklase *ItemAwareElementa,* pa je iz tog razloga ovaj element odabran kao relevantan koncept za proširenje BPMN meta-modela. *ItemAwareElement* može da bude proširen na dva načina: pomoću definicije proširenja za *OntologyElement* ili *ViewElement*. Ova mogućnost proširenja *ItemAwareElementa* na jedan od pomenuta dva načina, je na dijagramu definisana

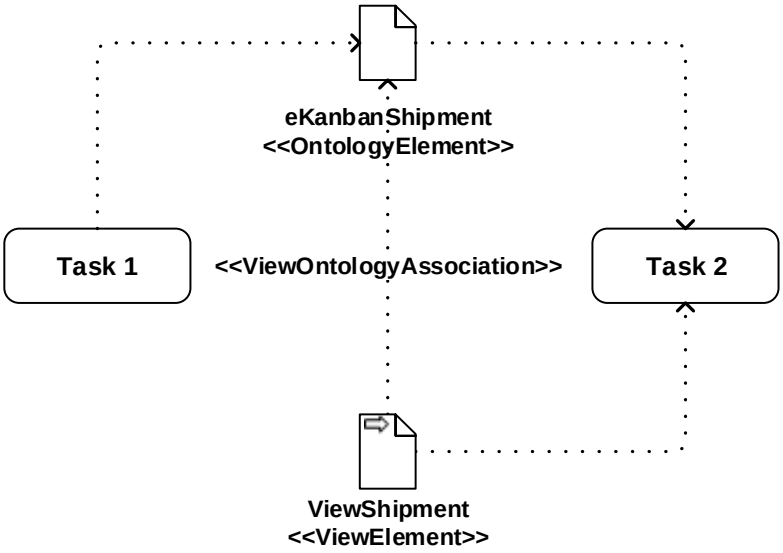
pomoću {or} ograničenja. Jedan *ItemAwareElement*, na primer *DataObject*, može praktično da sadrži ili *Dokument referentne ontologije* ili *Model pogleda* koji je definisan nad odgovarajućom referentnom ontologijom. *Dokument referentne ontologije* se predstavlja pomoću *OntologyElement*-a, dok se *Model pogleda* predstavlja pomoću *ViewElement*-a.

Prema BPMN 2.0 standardu, *Item Definition* i *Import* elementi se koriste za definiciju strukture podataka *ItemAwareElemenata*, na primer za definiciju strukture *DataInputa*. Ovi elementi se takođe koriste i u predlogu našeg proširenja, za importovanje stuktura podataka *Dokumenta referentne ontologije* ili *Modela pogleda*. Ukoliko nije drugačije specificirano, strukture su po pravilu serijalizovane u XML Schema formatu.

ViewElement ima *OntElementRef* referencu na *OntologyElement* od kojeg zavisi. Na ovaj način se definiše veza između *Dokumenta referentne ontologije* i odgovarajućeg *Modela pogleda* koji se iz njega izvodi u vreme izvršenja. Za potrebe vizuelne reprezentacije ove zavisnosti na dijagramu poslovnog procesa, definisan je novi element *ViewOntologyAssociation*. *ViewOntologyAssociation* predstavlja proširenje *DataAssociation* elementa. *DataAssociaton* element je proširen u smislu uvođenja ograničenja da:

- izvor veze (eng. *association source*) treba da bude *ViewElement* (*sourceRef* svojstvo);
- ponor veze (eng. *association target*) treba da bude *OntologyElement* (*targetRef* svojstvo).

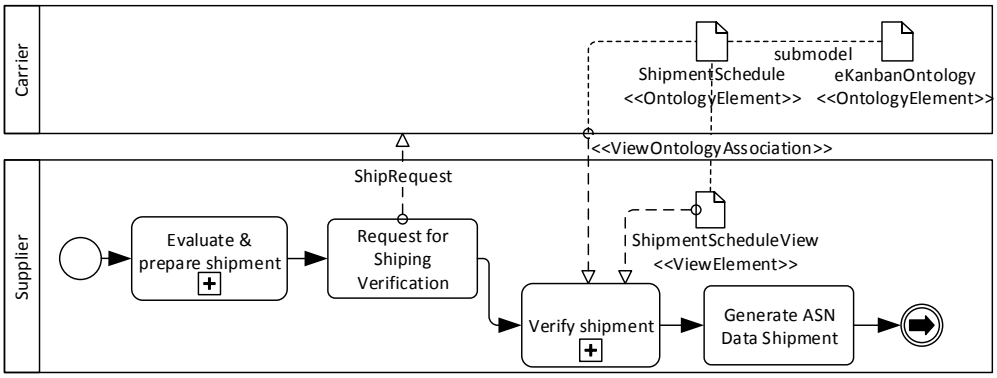
Na slici 5.15 je prikazan primer koji ilustruje primenu koncepata za koje su definisana proširenja BPMN meta-modela. Koncepti koji su proširenu su označeni odgovarajućim stereotipima.



Slika 5.15 BPMN proces sa elementima ekstenzije(Marija Jankovic et al., 2015)

5.3.1.3. Primena predložene formalizacije za specifikaciju informacionih tokova kolaborativnog procesa

Na slici 5.16 je prikazan primer korišćenja proširenih BPMN koncepata za povezivanje aktivnosti *Verify shipment,* sa odgovarajućim elementima za reprezentaciju referentne ontologije, odnosno *Modela pogleda*. Različiti tipovi *Data Object*-a su koji su povezani sa aktivnošću su anotirani koristeći odgovarajuće stereotipe predložene BPMN ekstenzijom. *OntologyElement* stereotip je primenjen na *DataObject* koji reprezentuje *ShipmentSchedule* dokument iz eKanban referentne ontologije. *ViewElement* stereotip koji je primenjen na *DataObject* predstavlja Model pogleda koji je nazvan *ShipmentScheduleView. ShipmentScheduleView* predstavlja pogled nad *ShipmentSchedule* dokumentom referentne ontologije, i bitno je napomenuti da sadrži samo one informacije koje su potrebne za izvršenje aktivnosti *VerifyShipment.* Veza između dokumenta referentne ontologije i relevantnog pogleda je eksplicitno prikazana na dijagramu pomoću stereotipa <<ViewOntologyAssociation>>.



Slika 5.16 Anotiran BPMN model (Marija Jankovic et al., 2015)

5.3.2. Korak 6: Specifikacija servisa kolaborativnog procesa

5.3.2.1. Teorijske osnove pristupa

Kao pogodna tehnika za implementaciju pojedinačnih servisa kolaborativnog poslovnog procesa u tezi se predlaže tehnologija veb servisa. Ova odluka je opravdana imajući u vidu da je jedan od ciljeva koji se žele postići specifikacijom aspekta servisa, upravo realizacija interoperabilnosti između različitih poslovnih partnera (Peltz, 2003; Sheth, 2011).

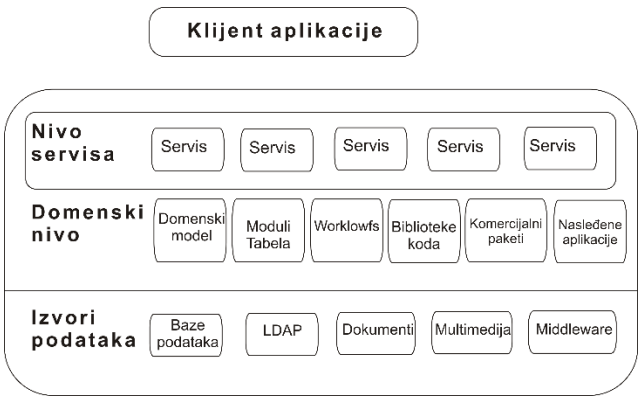
Svi veb servisi moraju da budu dizajnirani imajući u vidu određeni stil interakcije (Graham, 2008). Naime, za specifikaciju implementacije servisa je bitno uzeti u obzir način na koji će servisi da komunicirju sa klijentima. Upotreba veb servisa ukazuje na primenu distribuiranog klijent-server modela. Za klijent-server model je karakteristično da klijent šalje zahtev programu koji se nalazi na serveru. Na primer, klijent može da bude pošiljalac poruke koji šalje zahtev servisu koji je primalac poruke i nalazi se na serveru. Program na strani klijenta može da ima korisnički interfejs ili može da se izvršava bez intervencije korisnika kao pozadinski (eng. background) proces. Serverski programi se uvek izvršavaju kao pozadinski procesi, koji po prijemu zahteva mogu da ga procesiraju ili da ga proslede drugom serveru.

Osnovni i najjednostavniji klijent-server interakcioni patern je *Request/Response*, koji se koristi kada klijent očekuje trenutno izvršenje zahteva od strane servisa (Daigneau, 2012). Nakon uspostavljanja konekcije sa servisom, klijent šalje zahtev i

čeka odgovor. Server procesira zahtev odmah po prijemu i vraća odgovor putem iste konekcije. *Request/Response* patern opisuje sinhronu komunikaciju između klijenta i servera. Prema (Daigneau, 2012), u osnovne klijent-server paterne se pored *Request/Response* paterna ubrajaju: *Request/Acknowledge*, *Media Type Negotiation* i *Linked Service* paterni.

Pri specifikaciji implementacije servisa je bitno imati u vidu opštu logičku arhitekturu softverskog sistema. Osnovna karakteristika arhitekture savremenih višeslojnih sistema je podela na više slojeva koji poseduju logički povezane entitete. Jedna od glavnih prednosti višeslojne arhitekture je potpuno razdvajanje korisničkog interfejsa i prezentacije podataka od logike aplikacije. Prezentacioni sloj sadrži logiku koja prikazuje informacije i prima ulaz od korisnika. Sloj poslovne logike sadrži poslovnu logiku aplikacije i obezbeđuje sve servise neophodne za nesmetano izvršenje aplikacija. Moguća je jednostavna zamena i nadogradnja aplikacije, kao i promena načina čuvanja podataka koja ne utiče na klijenta. Za specifikaciju servisa koja se predlaže u disertaciji, jedan od polaznih uslova je bio da se sačuvaju navedene prednosti višeslojne arhitekture.

U svojoj knjizi „Patterns of Enterprise Application Architecture (POEAA)“ Martin Fowler, predlaže uvođenje posebnog nivoa servisa (*eng. service layer*) koji može da se koristi za kreiranje različitih Application Programming Interface (API) za različite tipove klijenata (Fowler, 2003). Daigneau uvodi nivo servisa koji definiše jasnu granicu između klijentskih aplikacija i logike specifičnog domena (Daigneau, 2012). Na slici 5.17 je prikazano da nivo servisa predstavlja deo domenskog nivoa.

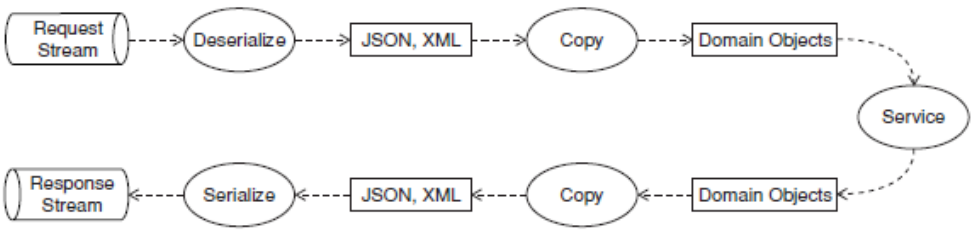


Slika 5.17 Odnos nivoa servisa i domenskog nivoa. Preuzeto i prilagođeno prema (Daigneau, 2012)

Uloga servisa koji pripadaju *nivou servisa*, je da ispunjavaju zahteve klijenata (eng. requests) koordinirajući akcije objekata koji pripadaju domenskom modelu (Daigneau, 2012). Servisi takođe mogu da obezbede pristup do komercijalih paketa, biblioteka koda, različitih *workflow* modela, itd. Za specifikaciju nivoa servisa, odnosno upravljanje zahtevima (eng. requests) i odgovorima (eng. responds) se predlažu četiri paterna (Daigneau, 2012; Fowler, 2003): (1) *Service Controller*, (2) *Data Transfer Object*, (3) *Request Mapper* i (4) *Response Mapper*.

Data Transfer Object (DTO)Patern

Glavni izazov klijent-server interakcionih paterna je da obezbede jednostavnu manipulaciju podacima iz *request* i *response* poruka klijenata. Bitna prednost upotrebe servisa sa aspekta interoperabilnosti, je mogućnost razmene podataka sa klijentima primenom otvorenih standarda kao što su XML i JavaScript Object Notation (JSON). Ipak, kada servis primi zahtev, podaci se često kopiraju iz XML i JSON struktura u domenske objekte čije je značenje smisleno (na primer kupac, dobavljač, itd.). Rad sa domenskim objektima je daleko lakši. Ovaj postupak koji je zahtevan i veoma komplikovan podrazumeva (Daigneau, 2012): (1) deserijalizaciju ili konvertovanje *request stream*-a u zahtevani format podataka (na primer XML ili JSON) i (2) parsiranje i kopiranje dobijenih struktura podataka u ciljne domenske objekte korsteći specifičan API za format podataka koji se čita. Isti postupak je potrebno ponoviti i reverzno u slučaju kada se šalje *response*. Bez obzira na format podataka ili alate za parsiranje koji se koriste, potrebno je napisati dosta programskog koda za ekstrakciju, konvertovanje i kopiranje podataka u ciljne domenske objekte. Opisani postupak je ilustrovan na slici 5.18.



Slika 5.18 Manipulacija request i response stream-om. Preuzeto i modifikovano prema (Daigneau, 2012)

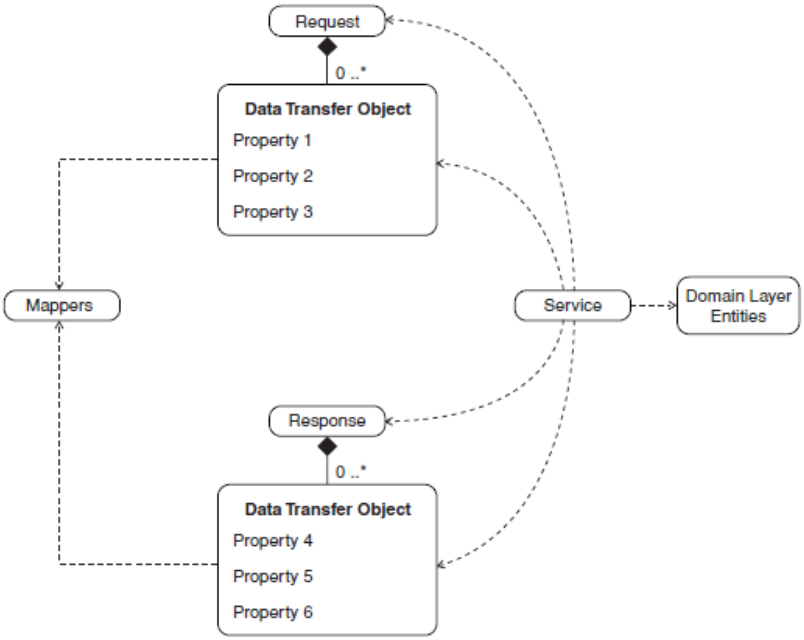
U pokušaju da pojednostave opisani postupak, programeri često modifikuju domenske objekte, kako bi uključili data-binding tehnologije kao što su Java Architecture for XML Binding (JAXB) i .NET *DataContractSerializer*. Naime, oni anotiraju svojstva (eng. property) domenskih objekata kao što su *get* i *set* metode ili atributi, kako bi definisali način na koji podaci treba da se koriste u *request* i *response* pozivima. Drugim rečima, kada servis primi *request* od klijenta, navedene tehnologije automatski izvrše deserijalizaciju *request stream*-a, inicijalizuju potrebne objekte i populiraju ih odgovarajućim podacima iz *request*-a.

Daigneau ukazuje na niz problema do kojih može doći kada se domenski objekti koriste u request i response pozivima (Daigneau, 2012). Prvo, programeri moraju da prevaziđu probleme do kojih može doći pri radu sa cikličnim referencama. Drugi problem se odnosi na usku povezanost sa domenskim modelom podataka. Naime, ako dođe do promene internog domenskog modela, zahtevaju se odgovarajuće promene i na strani klijenta. Slično, ukoliko klijent izvrši izmene u strukturi request-a ili response-a, potrebno je uskladiti domenski model.

DTO patern predstavlja jedno od mogućih rešenja navedenih problema (Daigneau, 2012; Fowler, 2003). DTO objekti se kreiraju kao posebni entiteti čija je uloga isključivo da definišu način na koji se primaju podaci od servisa, i vraćaju nazad ka servisu. Fowler definiše DTO objekte kao ponovno upotrebljive klase koje sadrže podatke, ali ne i poslovnu logiku (Fowler, 2003). Njihova svojstva kao što su *get* i *set* metode, mogu da obavijaju primitivne tipove podataka (integer, string, itd.) ili druge DTO objekte. DTO klase mogu da se definišu na strani servisa, klijenta ili na obe strane komunikacionog kanala.

DTO objekti su prvo opisani sa ciljem da se smanji broj poziva metoda u distribuiranim objektnim sistemima (na primer Common Object Request Broker Architecture (CORBA), Distributed Component Object Model (DCOM)). Daigneau je pokazao da se ovaj isti patern može koristiti od strane servisa da bi se pojednostavilo upravljanje *request* i *response* podacima, i odvojila struktura poruke od nivoa domenskih entiteta. On objašnjava da je upravljanje *request/response* podacima

jednostavnije, s obzirom na to da servisi ne moraju da koriste API za JSON, XML ili druge formate. Na slici 5.19 je prikazan DTO patern.



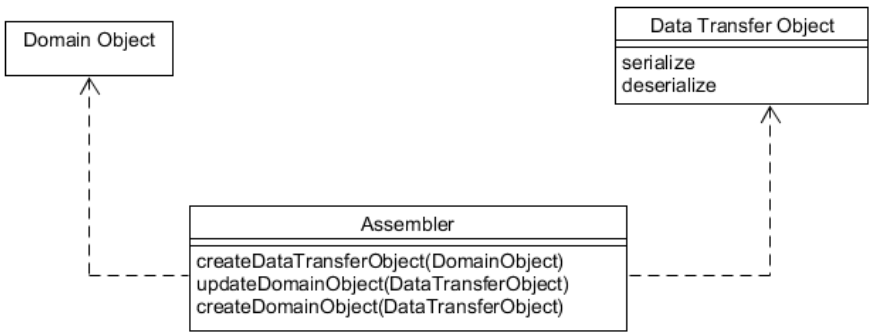
Slika 5.19 Data Transfer Object patern (Daigneau, 2012)

Na slici 5.19 se može videti da su entiteti domenskog nivoa odvojeni od *request* i *response* struktura, jer se DTO objekti kreiraju kao posebni entiteti čija je uloga isključivo da definišu način na koji se primaju podaci od i vraćaju nazad ka servisu.

Da bi mogao da se transportuje preko mreže DTO objekat mora da bude serijalizovan. Obično se na serverskoj strani koristi poseban program za transfer podataka između DTO i domenskih objekata. Fowler za ovaj program koristi termin *maper* (eng. mapper), dok Daigneau koristi naziv *asembler* (eng. assembler) (Daigneau, 2012; Fowler, 2003). DTO objekat može da sadrži podatke iz više povezanih domenskih objekata, na primer podatke o narudžbenici sa stavkama i podatke o kupcu. Ipak se preporučuje da se vodi računa o složenosti DTO objekata, na primer nije poželjno predstaviti čitav domenski model jednim DTO objektom budući da može doći do problema prilikom serijalizacije. Da li je praktičnije kreirati jedan DTO za čitavu interakciju, ili poseban DTO za svaki request zavisi od konkretnog poslovnog scenarija. U disertaciji se predlaže da se za svaku poruku koja se razmenjuje između učesnika kolaboracije definiše poseban DTO objekat.

Pored jednostavnih get i set metoda, DTO objekat može da sadrži metode koje su potrebne za njegovu serijalizaciju. Brojne platforme imaju ugrađene mehanizme za serijalizaciju jednostavnih objekata (Daigneau, 2012). Na primer, JAVA platforma poseduje mehanizam za binarnu serijalizaciju, dok .NET poseduje mehanizam za binarnu i XML serijalizaciju.

Bitna karakteristika DTO objekata je da ne zavise od domenskih objekata. Pri specifikaciji servisa koji koriste DTO objekte je bitno imati u vidu da ova značajna karakteristika mora da se zadrži. Fowler predlaže da se kreira poseban asembler objekat čija je odgovornost da kreira DTO iz domenskog modela i da ažurira domenski objekat na osnovu njega (slika 5.20) (Fowler, 2003). *Asembler objekat* je kreiran po uzoru na *Mapper patern*, jer je njegova uloga da obezbedi mapiranje između DTO i domenskih objekata.



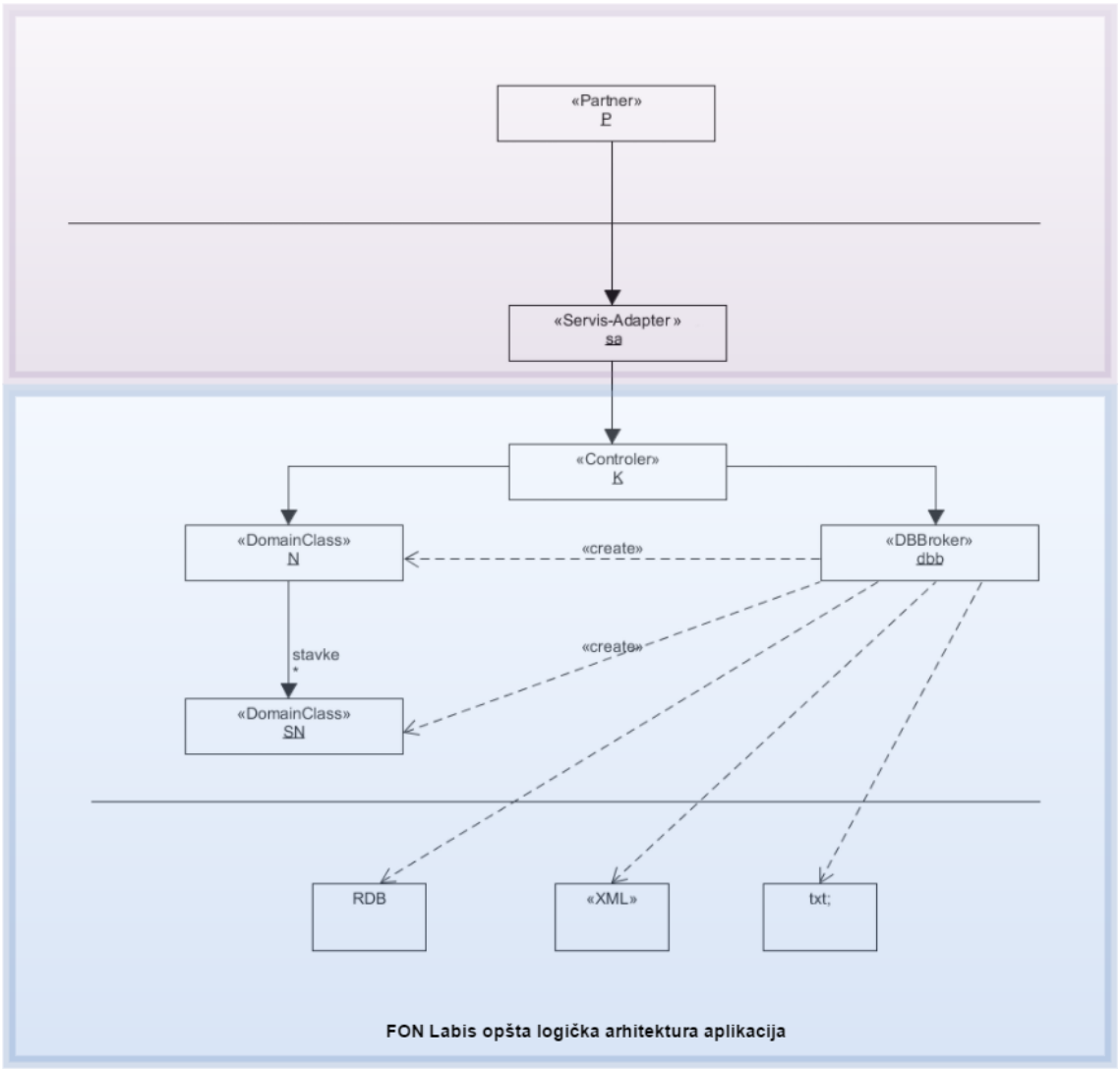
Slika 5.20 Asembler objekat (Fowler, 2003)

Prema klasifikaciji koja je data u (Fowler, 2003), DTO patern spada u grupu distribucionih paterna. Ovoj grupi paterna pripada i *Remote Fascade* patern koji je interesantan jer prodržava rad sa DTO objektima. Ovaj patern formira „fasadu“, odnosno pruža mogućnost klijentu da formira složen *request* objekat koji grupiše potrebne objekte.

5.3.2.2. Specifikacija Servis Adaptera

Za specifikaciju servisa kolaborativnog poslovnog procesa se predlaže nova logička arhitektura koja proširuje FON Labis logičku arhitekturu za implementaciju

aplikacija, primenom posebno dizajniranog *Servis Adapter* paterna koji je prikazan na slici 5.21.

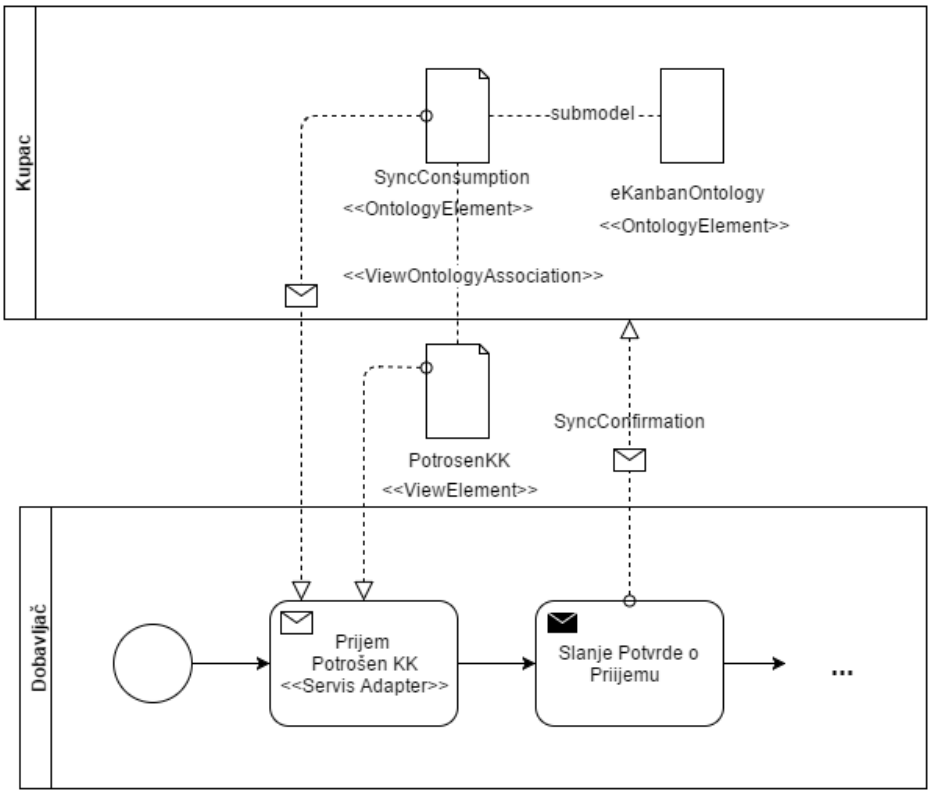


Slika 5.21 Proširenje standardne FON Labis logičke arhitekture aplikacija uvođenjem Servis Adapter Paterna

Servis Adapter patern je dizajniran po uzoru na preporuke za implementaciju *Asembler* objekta i *Remote Facade* paterna (Fowler, 2003). Slično *Remote Facade* paternu, *Servis Adapter* ima mogućnost rada sa DTO objektima. Njegove funkcionalnosti su dizajnirane imajući u vidu polazni zahtev da se u potpunosti zadrže karakteristike FON Labis logičke arhitekture koja je bazirana na UC MVC paternu. Implementacija poslovne logike kojom upravlja *Kontrolor* treba da ostane neizmenjena. Bitno je naglasiti da *Kontrolor* i dalje implementira logiku posmatranog slučaja korišćenja, upravlja transakcijom i komunicira sa bazom

podataka. Osnovna uloga *Servis Adaptera* je da omogući manipulaciju podacima iz *request* objekta koji šalje klijent i transformiše ih u adekvatnu formu za *Kontrolora*, kao što je prikazano na slici 5.21. Naime, sva komunikacija između Klijenta i *Kontrolora* se obavlja isključivo preko *Servis Adaptera*.

Uloga *Servis Adaptera* će biti objašnjena na primeru prijema standardne IV&I eKanban *SyncConsumption* poruke koju Kupac šalje Dobavljaču kada dođe do potrošnje Kanban kontejnera. Logički dijagram sekvenci za prijem *SyncConsumption* poruke je prikazan na slici 5.23. Pre detaljnog objašnjena strukture logičkog dijagrama sekvenci, ukazaćemo na njegovu vezu sa elementima koji su prethodno specificirani na privatnom dijagramu kolaboracije. Na slici 5.22 je prikazana aktivnost *Prijem Potrošen KK* čija je uloga da primi *SyncConsumption* poruku koju šalje Kupac. Na dijagramu je moguće videti da *SyncConsumption* poruka predstavlja podmodel *eKanbanOntology* referentne ontologije. Prema tome, *SyncConsumption* poruka je definisana koristeći koncepte globalne referentne ontologije. Za interne potrebe Dobavljača je potrebno mapirati globalnu *SyncConsumption* poruku u lokalnu *PotrošenKK* poruku. Lokalna *PotrosenKK* poruka predstavlja pogled (<<*ViewElement*>>) nad *SyncConsumption* porukom (<<*OntologyElement*>>), koji se kreira na osnovu definisanog *UML View Profila*. Predloženi način za detaljnu specifikaciju strukture lokalnih poruka, kao pogleda nad globalnom referentnom ontologijom upotrebom *UML View Profila* je objašnjen u poglavlju (5.3.3).

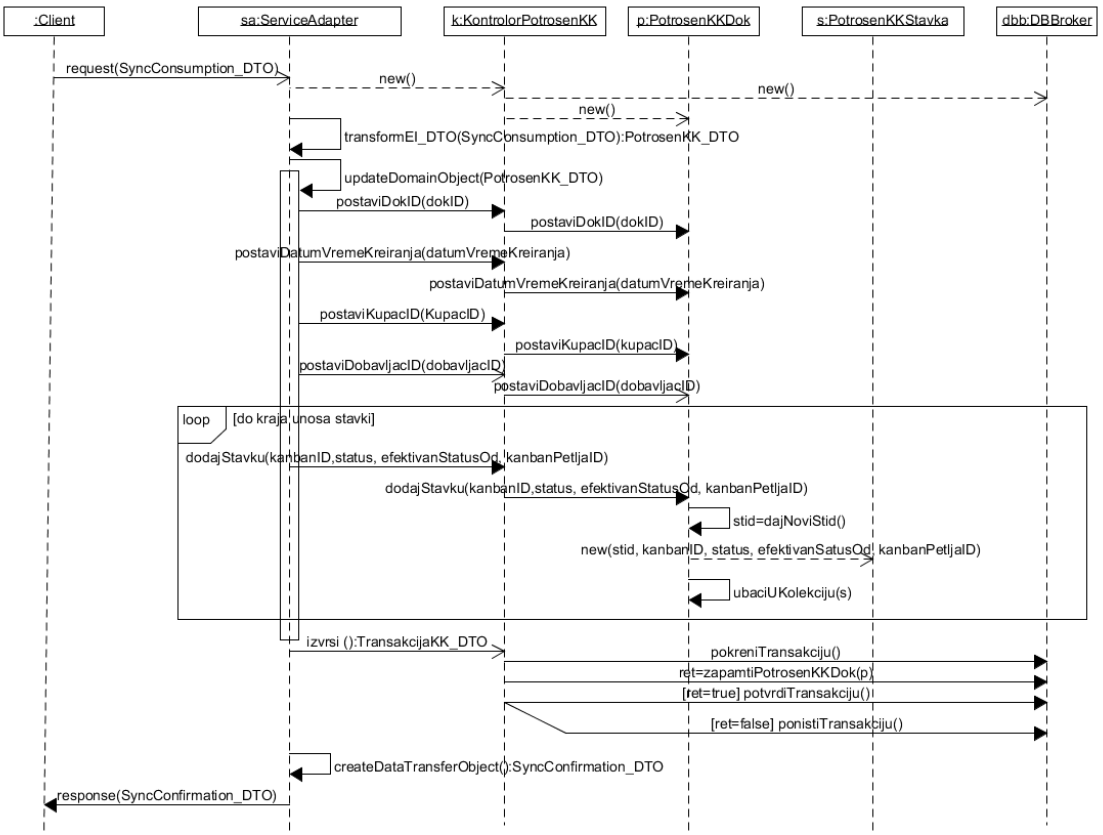


Slika 5.22 Veza procesa Prijem Potrošen KK sa eksternom porukom SyncConsumption i njenom internom reprezentacijom PotrošenKK

Na dijagramu na slici 5.22 je pomoću stereotipa *<<Servis Adapter>>* specificirano da aktivnost *Prijem Potrošen KK* predstavlja posebnu vrstu servisa čija implementacija podrazumeva primenu *Servis Adapter* paterna. Za lakšu manipulaciju podacima iz *request* i *response* poruka, u radu se predlaže primena DTO paterna. U skladu sa datom preporukom, eksterna standardna *SyncConsumption* poruka koja se koristi u globalnoj kolaboraciji, kao i njena interna reprezentacija *PotrosenKK* predstavljaju DTO objekte.

Po prijemu *SyncConsumption* poruke, uloga *Servis Adaptera* je da izvrši njenu transformaciju u odgovarajuću lokalnu reprezentaciju. Metoda *transformEI_DTO Servis Adaptera* kao ulazni parametar prima *SyncConsumption_DTO* poruku i transformiše je u lokalnu *PotrošenKK_DTO* poruku. Za transformaciju poruke *Servis Adapter* koristi OCL pravila koja su defisana *UML View Profilom*. Nakon transformacije, *Servis Adapter* izvršava metodu *updateDomainObject*, koja kao ulazni parametar ima složeni interni objekat *PotrosenKK_DTO. Servis Adapter* vrši ekstrakciju potrebnih podataka iz složenog DTO objekta i u odgovarajućoj formi ih

prosleđuje *Kontroloru*. Ova funkcionalnost je implementirana po uzoru na *Remote Fascade* patern (Fowler, 2003). Na taj način *KontrolorPotrosenKK* zadržava svoju standardnu funkcionalnost, odnosno dalje prosleđuje preuzete parametre do odgovarajućih domenskih objekata. Time se postiže da komunikacija između *Kontrolora*, *Domenskih objekata* (*PotrosenKKDok i PotrosenKKStavka*) i *DBBrokera* ostane neizmenjena. Nakon prosleđivanja potrebnih podataka *Kontroloru*, *Servis Adapter* pokreće transakciju za pamćenje podataka u bazi pomoću metode *izvrsi ()*. U slučaju uspešnosti izvršene transakcije *Kontrolor* vraća *Servis Adapteru true*, u suprotnom *false*. Signal o uspešnosti izvršenja transakcije je potrebno vratiti *Klijentu* u formi odgovarajuće *response* poruke. S obzirom na to da se komunikacija između *Servis Adaptera* i *Klijenta* vrši isključivo preko DTO objekata, *Servis Adapter* kreira odgovarajući eksterni DTO objekat i vraća ga *Klijentu*, što je na dijagramu 5.22 prikazano pomoću metode *createDataTransferObject()*.



Slika 5.23 Proširenje FON Labis dijagrama sekvenci primenom Servis Adapter paterna

Za komunikaciju između *Klijenta* i *Servis Adaptera* na slici 5.23 je primenjen osnovni *Request/Response* patern. U zavisnosti od specifičnog poslovnog scenarija, moguća je i primena ostalih paterna. Naime, UN/CEFACT's Modeling Methodology (UMM) definiše dva tipa jednosmernih i četiri tipova dvosmernih poslovnih tansakcija (UMM, 2011).

Na kraju ćemo sumirati osnovne funcionalnosti *Servis Adapter* paterna:

- Komunikacija sa klijentom koja podrazumeva prijem *request* i slanje odgovarajućih *response* poruka.
- Transformacija eksterne poruke u strukturu odgovarajućeg internog DTO objekta, koristeći OCL pravila koja su definisana UML View Profilom. Odnosno, mapiranje eksterne poruke koja je definisana koristeći koncepte globalne referentne ontologije, na internu reprezentaciju koristeći koncepte lokalne ontologije.
- Komunikacija sa standardnim Kontrolorom FON Labis metodologije (prosleđivanje parametara i prijem povratnih vrednosti funkcija).
- Kreiranje eksternih DTO objekata na osnovu povratnih vrednosti funkcija Kontrolora.

Pravila za definisanje internih DTO objekata, koji predstavljaju poglede nad odgovarajućom referentnom ontologijom su definisana UML View Profilom koji je objašenjen u narednom poglavlju.

5.3.3. Korak 7: Specifikacija strukture poruka kolaborativnog procesa

5.3.3.1. Teorijske osnove pristupa

UML je standardni grafički jezik za projektovanje, specifikaciju, vizuelizaciju i dokumentovanje svih tipova programskih sistema u različitim fazama životnog ciklusa (OMG, 2010a; OMG, 2010b; Rumbaugh, Jacobson, & Booch, 2004; Booch, Rumbaugh, & Jacobson, 2005). Bitna karakterisitika UML-a je mogućnost definisanja proširenja za reprezentaciju koncepata specifičnog domena poslovanja ili koncepata specifične poslovne platforme (OMG, 1999b). UML definiše nekoliko standardnih mehanizama proširenja (ekstenzija) u koje spadaju stereotipi (eng. stereotypes),

označene vrednosti (eng. tagged values), ograničenja (eng. constraints) i ikone za grafičku reprezentaciju (eng. icons). Skup navedenih mehanizama za proširenje standardne UML notacije se naziva UML Profil. Kombinacijom ovih mehanizama je moguće kreirati nove tipove gradivnih elemenata koji su potrebni za modelovanje nekog specifičnog sistema (OMG, 1999a). Bitno je naglasiti da UML specijalizuje postojeće koncepte i definiše konvencije za njihovu primenu. Drugim rečima, UML Profil ne uvodi nove osnovne koncepte.

Stereotip kreira virtuelnu UML metaklasu koja se bazira na postojećoj UML metaklasi, i na taj način obezbeđuje klasifikaciju instaci bazne metaklase. Stereotip predstavlja proširenje rečnika UML-a i dozvoljava da se pridoda novo semantičko značenje nekom elementu modela. UML ne definiše ograničenja za primenu stereotipa, tako da ovaj mehanizam može da se koristi za proširenje bilo kog elementa modela. Stereotip može da se predstavi na dva načina: kao tekst između para uglastih zagrada << >> ili pomoću ikonice koja predstavlja specifičnu grafičku reprezentaciju.

Označene vrednosti imaju ulogu atributa UML metaklase, pomoću kojih se mogu definisati nove osobine klase. Najjednostavniji oblik za reprezentaciju označenih vrednosti je pomoću niza znakova u vitičastim zagradama koje se navode ispod elementa na koji se odnose. Niz znakova kojim se definiše označena vrednost se sastoji od imena, separatora (simbol =) i vrednosti (naznake). Skup označenih vrednosti može da bude primenjen na elemente modela stereotipa kojem je priključen.

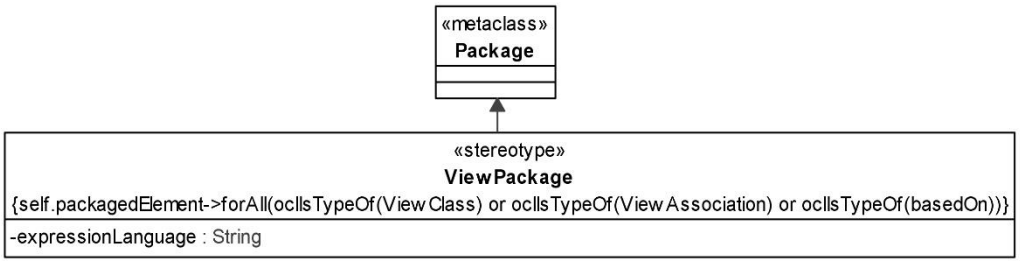
Ograničenja omogućavaju tekstualnu specifikaciju semantike elemenata modela. Ograničenja se označavaju nizom znakova u vitičastim zagradama i nalaze se ispred imena elementa na koji se odnose. Za opis ograničenja se koristi određeni jezik, na primer fomalni jezik za opis ograničenja u UML metamodelima je OCL (OMG, 2006). Ograničenja mogu da budu specificirana i neformalno, koristeći konstrukcije govornog jezika. Na osnovu ograničenja može da se izvrši validacija modela, odnosno da se proveri da li je model dobro formiran.

5.3.3.2. UML View Profil kao mehaizam za formalizaciju strukture poruka kolaborativnog procesa

U ovom poglavlju je predstavljen UML Profil, koji se predlaže kao formalni mehanizam za identifikaciju informacionih zahteva na nivou aktivnosti poslovnog procesa. Primenom predloženog UML Profila, informacioni zahtevi mogu da se definišu kao podgrupa svojstava odgovarajućih poslovnih entiteta globalne referentne ontologije. Za potrebe kreiranja UML Profila definisani su sledeći stereotipi: *ViewPackage*, *ViewClass*, *ViewProperty*, *ViewAssociation*, *basedOn* i *Key*. U tabeli 5.1 su prikazane karakteristike stereotipa *ViewPackage*, dok su odgovarajuće klase prikazane na slici 5.24.

Tabela 5.1 Stereotip ViewPackage

***Stereotip*: ViewPackage**	
Osnovna klasa	Package
Opis	Predstavlja paket koji sadrži definiciju modela pogleda.
Ograničenja	Članice paketa moraju da budu iz sledećeg skupa stereotipa: ViewClass, ViewAssociation ili based On.
Tagged Values	expressionLanguage – je jezik koji se koristi za definisanje izraza i izvođenje pravila unutar članica paketa.



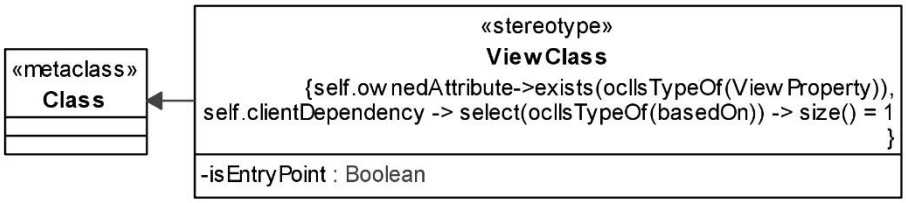
Slika 5.24 Stereotip ViewPackage

U tabeli 5.2 su prikazane karakteristike stereotipa ViewClass, dok su odgovarajuće klase prikazane na slici 5.25.

Tabela 5.2 Stereotip ViewClass

Stereotip*: View*Class	
Osnovna klasa	Class
Opis	Predstavlja klasu koja služi za definiciju pogleda, koji je baziran na odgovarajućoj klasi referentne ontologije. Sadrži ViewProperty svojstva za definisanje podgrupe svojstava odgovarajuće referentne ontologije.
Ograničenja	Mora da sadrži makar jedno svojstvo koje ima stereotip ViewProperty. Mora da bude bazirana na nekoj od klasa

	referentne ontologije (predstavljeno pomoću veze zavisnosti sa sterotipom basedOn).
Tagged Values	isEntryPoint – označava da li je ViewClass ulazna tačka pogleda, na primer inicijalna tačka za izvršenje transformacije.

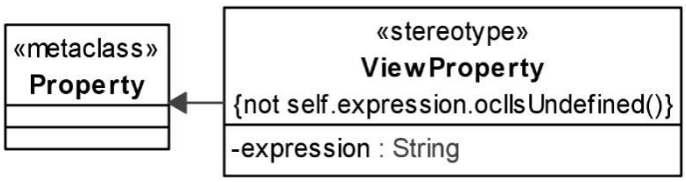


Slika 5.25 Stereotip ViewClass

U tabeli 5.3 su prikazane karakteristike stereotipa ViewProperty, dok su odgovarajuće klase prikazane na slici 5.26.

Tabela 5.3 Stereotip ViewProperty

Stereotip*: View*Property	
Osnovna klasa	Property
Opis	Predstavlja svojstvo definisano u okviru ViewClass. Vrednost svojstva je određena izrazom koji se definiše nad svojstvima relevantne klase referentne ontologije.
Ograničenja	Mora da ima definisanu vrednost za *tagged value* izraz
Tagged Values	expression – označava pravilo izvođenja, odnosno definiše pravila mapiranja za jedan ViewProperty prema jednom ili više svojstava odgovarajuće klase referentne ontologije.

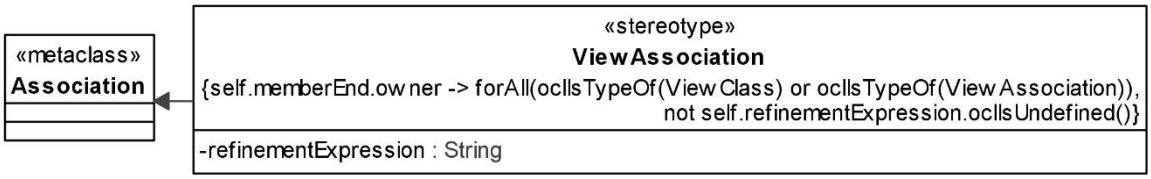


Slika 5.26 Stereotip ViewProperty

U tabeli 5.4 su prikazane karakteristike stereotipa ViewAssociation, dok su odgovarajuće klase prikazane na slici 5.27.

Tabela 5.4 Stereotip ViewAssociation

Stereotip*: View*Association	
Osnovna klasa	Association
Opis	Reprezentuje asocijaciju koja povezuje dve ViewClass-e.
Ograničenja	Vlasnik kraja asocijacije (*eng. association end*) mora da bude ViewClass ili ViewAssociation. Mora da ima definisanu vrednost za *refinementExpression*.
Tagged Values	refinementExpression – je izraz koji definiše uslov za dodatno filtriranje skupa ViewClass objekata na ViewAssociation kraju.

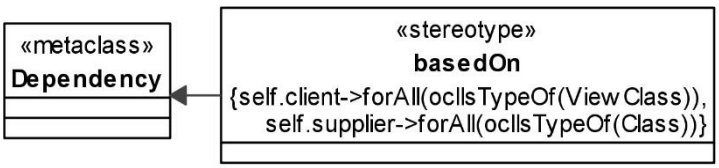


Slika 5.27 Stereotip ViewAssociation

U tabeli 5.5 su prikazane karakteristike stereotipa basedOn, dok su odgovarajuće klase prikazane na slici 5.28.

Tabela 5.5 *Stereotip BasedOn*

Stereotip*: *basedOn	
Osnovna klasa	Dependency
Opis	Definiše zavisnost ViewClass od odgovarajuće klase referentne ontologije. Drugim rečima, definiše onu klasu referentne ontologije na osnovu čijih svojstava je formiran podskup ViewProperty u okviru ViewClass.
Ograničenja	Izvor basedOn zavisnosti mora da bude ViewClass, dok odredište mora da bude Class.

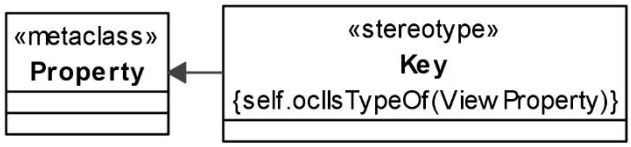


Slika 5.28 Stereotip basedOn

U tabeli 5.6 su prikazane karakteristike stereotipa Key, dok su odgovarajuće klase prikazane na slici 5.29.

Tabela 5.6 *Stereotip Key*

Stereotip*: *Key	
Osnovna klasa	Property
Opis	Reprezentuje identifikator ViewClass-a.
Ograničenja	Mora da se primeni na ViewProperty.



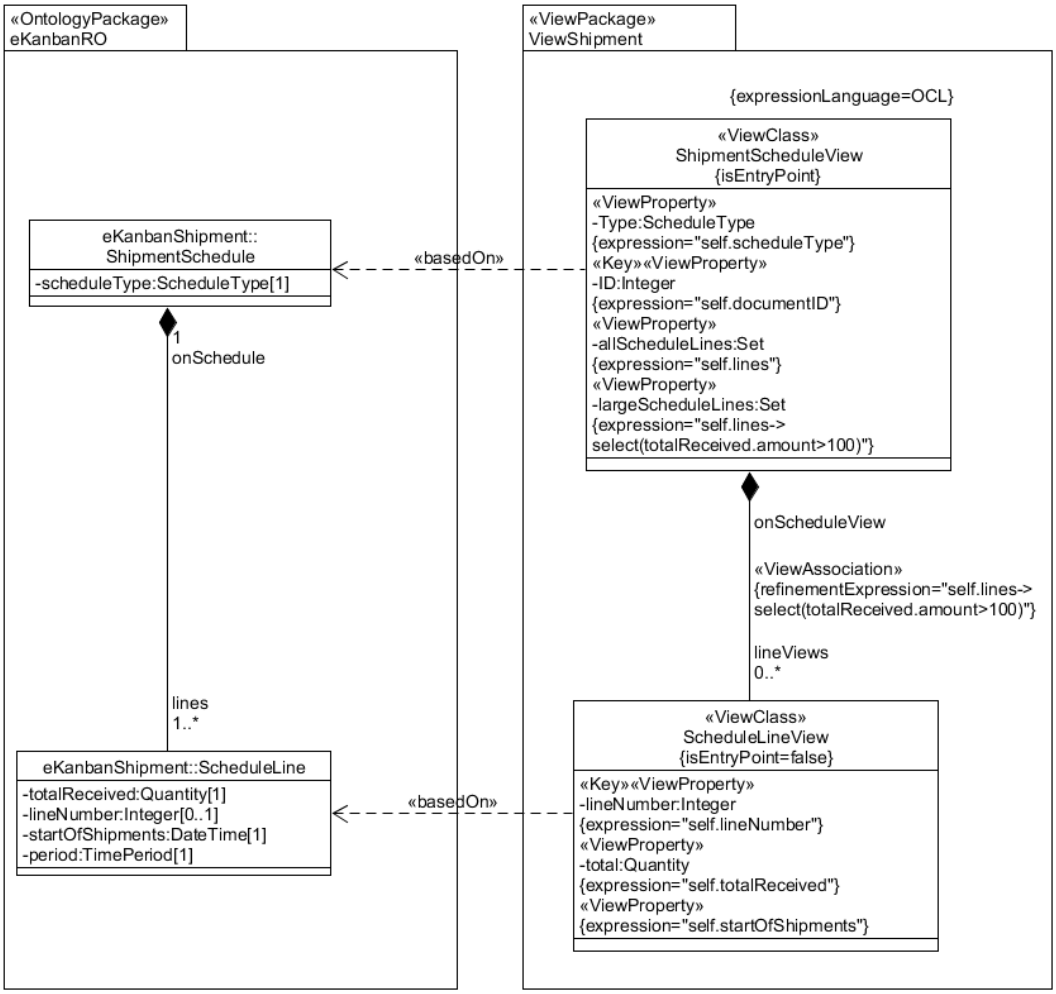
Slika 5.29 Stereotip Key

5.3.3.3. Primena predložene formalizacije za specifikaciju strukture poruka kolaborativnog procesa

U ovom poglavlju ćemo ilustrovati primenu predloženog UML View Profila na primeru eKanban referentne ontologije. Model pogleda je definisan nad *ShipmentSchedule* dokumentom eKanban referentne ontologije. Zbog preglednosti, unutar *eKanbanRO* paketa na slici 5.30 su prikazani samo neophodni elementi za kreiranje *ShipmentSchedule* pogleda. Model pogleda je definisan unutar *ViewShipment* paketa, sa *ShipmentScheduleView ViewClass*-om kao polaznom tačkom za transformaciju[1].

Na slici 5.30 je prikazana definicija *ShipmentScheduleView* pogleda nad *ShipmentSchedule* elementom eKanban referentne ontologije. *ShipmentScheduleView* se mapira na *ShipmentSchedule* klasu eKanban referentne ontologije, što je definisano pomoću *basedOn* zavisnosti. Stereotipi *View Properties* klase *ShipmentScheduleView* su mapirani na svojstva *SchipmentSchedule* klase. Mapiranje između klase pogleda i odgovarajuće klase referentne ontologije je definisano pomoću OCL izraza, koji su prikazani u okviru *tagged values* za svaki *ViewProperty* pojedinačno. OCL izrazi koji definišu potrebna mapiranja, su u formi koja je pogodna za izvršenje automatske transformacije. Ključna reč *self* unutar OCL izraza, služi da označi klasu referentne ontologije na koju se vrši mapiranje klase pogleda. Na primer, *ViewProperty Type* svojstvo je definisano pomoću OCL izraza "*self.schedule.Type*" koji se izvršava nad *ShipmentSchedule* instancom, i kao rezultat izvršenja ima vrednost njenog *sheduleType* svojstva.

[1] Ograničenje {isEntryPoint} ima vrednost *true*



Slika 5.30 Definicija ShipmentScheduleView pogleda (Marija Jankovic et al., 2015)

Specifikacija *ViewProperty*-ija proizvoljne *ViewClass*-e može da se definiše ne samo na osnovu svojstava polazne klase referentne ontologije na koju se mapira, već i na osnovu svojstava svih ostalih klasa koje su u relaciji sa njom. U skladu sa tim, *SchipmentSheduleView* sadrži *allScheduleLines ViewProperty* svojstvo koje reprezentuje skup svih *ScheduleLine* objekta klase *ShipmentSchedule*. Slično, sadrži *largeSheduleLines ViewProperty* svojstvo koje reprezentuje skup selektovanih *ScheduleLine* objekata u skladu sa kriterijumom da je ukupna primljena količina veća od 100. U oba slučaja, rezultujući skup sadrži kompletne *SheduleLine* objekte, tj. objekti poseduju sva svojstva *ScheduleLine* klase. Ukoliko bi bilo potrebno da se koristi samo odgovarajući podskup svojstava klase *ScheduleLine*, morao bi da se definiše novi pogled, odnosno novi *ViewClass*. Na primer, na slici 5.30 je kreiran pogled *SheduleLineView* koji je baziran na *SheduleLine* klasi referentne ontologije, i

specificirana su željena svojstva. Uz to je dodatno potrebno definisati, novu vezu *ViewAssociation* sa odgovarajućim izrazom za filtriranje. Na slici 5.30 je prikazana veza *ViewAssociation* između *ShipmentScheduleView* i *SheduleLineView* klasa.

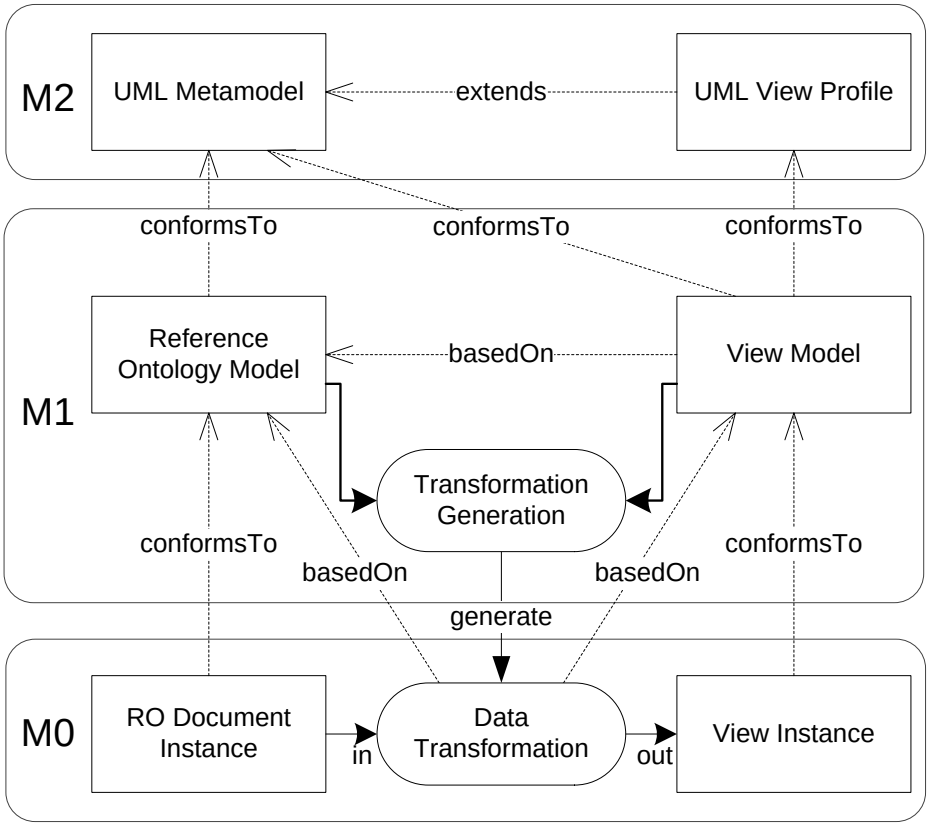
Bitno je naglasiti da sada *ShipmentScheduleView* ima svojstvo *largeScheduleLines* i *ViewAssociation,* definisane koristeći identičan izraz: *self.lines->select (totalReceived.amount).* Ovaj primer je odabran da bi se ilustrovalo da isti izraz može da kao rezultat izvršenja ima dva skupa različitih objekata. Naime, *ViewProperty largeScheduleLines* će sadržati skup *ScheduleLine* objekata, dok će skup koji je dobijen kao rezultat izvršenja izraza preko *ViewAssociation* sadržati *ScheduleLineView* objekte.

5.4. Implementacija aspekata interoperabilnosti

Predloženi pristup za specifikaciju aspekata interoperabilnosti je dizajniran imajući u vidu mogućnost automatizacije implementacije. U tezi je predstavljen način formalizacije svih faza predloženog pristupa u skladu sa Model Driven Architecture (MDA) principima. U fazi implementacije su identifikovana dva osnovna koraka: (1) implementacija servisa kolaborativnog procesa i (2) njihova orkestracija, odnosno implementacija čitavog kolaborativnog poslovnog procesa.

5.4.1. Implementacija servisa kolaborativnog procesa

Prvi korak se odnosi na implementaciju servisa kolaborativnog poslovnog procesa, čija nezavisna specifikacija omogućava različite načine implementacije. Predlaže se da generalizacija transformacije poruka bude zasnovana na MDA pristupu. Strukture poruka koje se razmenjuju su opisane pomoću *UML View Profila*, pa se može definisati opšti algoritam koji omogućava transformaciju formata jedne poruke u drugi.

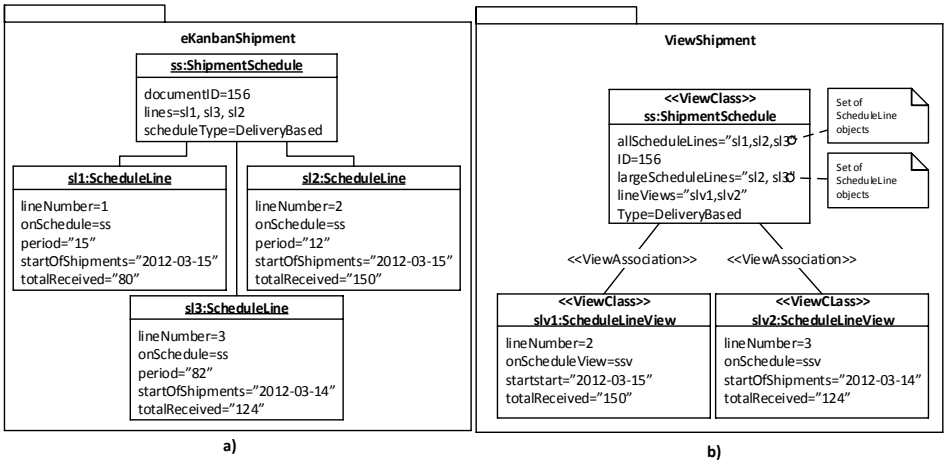


Slika 5.31 Transformacija modela podataka (Marija Jankovic et al., 2015)

UML View profil se koristi za specifikaciju semantičkih pravila mapiranja između modela pogleda i modela referentne ontologije, što je prikazano na M1 meta-nivou na slici 5.31. Definisana pravila su sadržana u definiciji modela pogleda koristeći OCL jezik. Na osnovu definisanih OCL pravila mogu da se generišu transformaciona pravila za izvođenje instance modela pogleda na osnovu odgovarajuće instance modela referentne ontologije. Odgovarajuća instanca dokumenta referentne ontologije i instanca pogleda su prikazane na nivou M0 na slici 5.31. Postoji više različitih načina na koji može da se definiše transformacija modela. Object Mangement Group (OMG) kao standardan način za transformaciju modela predlaže Query/View/Transformation (QVT) specifikaciju. Drugi pristup za transformaciju modela čine XML transformacioni jezici kao što su XQUERY i Extensible Stylesheet Language Transformations (XSLT). Pristup za transformaciju modela koji se predlaže u tezi koristi QVT, na osnovu kojeg se po potrebi mogu generisati transformacije bazirane na XML-u.

Transformaciona pravila za dobijanje instance modela referentne ontologije mogu da se generišu automatski na bazi pravila koja su definisana modelom pogleda. Ovaj automatizam je moguć, s obzirom na činjenicu da i sama definicija transformacije može da se predstavi kao model. Konkretno, rezultat QVT transformacije može da bude sama QVT transformacija. Da bi rezultat QVT transformacije mogao da se izvrši kao nova QVT transformacija, QVT specifikacija definiše *asTransformation* operaciju. Ova operacija se koristi da se pozovu *on-the-fly* transformacije definicija koje su kreirane dinamički. Ova QVT osobina se koristi u pristupu predloženom u tezi, kao što je na slici 5.31 reprezentovano pomoću *Transformation Generation* čvora. U ovom koraku, definicije QVT transformacije mogu da se generišu dinamički na bazi pravila koja su definisana modelom pogleda. Algoritam za generisanje transformacije se bazira na činjenici da izvorni (eng. source) i ciljni (eng. target) modeli predstavljaju instance istog meta-modela. Polazna tačka za izvršenje transformacije je definisana pomoću označene vrednosti (eng. tagged value) *entryPoint* stereotipa <<ViewClass>>. Pravila za generisanje QVT transformacije na osnovu definisanih OCL izraza se primenjuju prvo na <<ViewProperty>> svojstva i <<ViewAssociation>> relacije, one <<ViewClass>> klase koja je označena kao polazna tačka (EntryPoint=true). Nakon toga, transformaciona pravila se primenjuju na ostale <<ViewClass>> klase, njihova <<ViewProperty>> svojstva i <<ViewAssociation>> relacije. U sledećem koraku se izvršavaju QVT transformacije na nivou M0, koje su označene čvorom *Data transformation*. Transformacija podataka za validaciju predloženog pristupa je izvršena koristeći Eclipse Modeling Framework (EMF) uz primenu QVT implementacije.

Na slici 5.31 je prikazan primer instanci *ShipmentSchedule* i *ShipmentScheduleView* dokumenata, koji su dobijeni kao rezultat transformacionog procesa.



Slika 5.32 Primer instance dokumena Referentne Ontologije i Instance Pogleda (Marija Jankovic et al., 2015)

5.4.2. Orkestracija kolaborativnog poslovnog procesa

Predloženi metodološki pristup za specifikaciju aspekata interoperabilnosti predlaže BPMN2.0 notaciju za detaljnu specifikaciju kolaborativnog poslovnog procesa. Preporuke za implementaciju pojedinačnih servisa kolaborativnog poslovnog procesa su opisane u prethodnom poglavlju (5.4.1). U ovom poglavlju se daje kratka diskusija mogućih pristupa za implementaciju čitavog kolaborativnog poslovnog procesa. Specifikacija i implementacija kolaborativnog poslovnog procesa, koristeći pristup za specifikaciju pojedinačnih servisa koji se predlaže u tezi, je interesantna tema i jedan od mogućih daljih pravaca istraživanja.

Dva osnovna pristupa za kompoziciju servisa koji se predlažu u literaturi su orkestracija (eng. orchestration) i koreografija (eng. choreography) (Miers, 2008; OMG BPMN 2.0, 2011; Silver, 2011; Weske, 2012b). Za orkestaciju je karakteristično da postoji centralni element koji upravlja svim ostalim aspektima poslovnog procesa (Weske, 2012b). Za razliku od orkestracije, koreografija podrazumeva autonomnost svih elemenata poslovnog procesa.

Orkestracija predstavlja opšti pristup koji može da se koristi za modelovanje poslovnih procesa, ali i za kompoziciju servisa (Weske, 2007). SOA obezbeđuje teoretski okvir za kompoziciju servisa, kao fundamentalnih jedinica za konstruisanje poslovnih procesa (Erl, 2004, 2007; Graham, 2008). Orkestracija

podrazumeva definisanje sekvence aktivnosti kolaborativnog poslovnog porcesa. Nakon definisanja sekvence aktivnosti je potrebno kreirati centralnog kontrolora za njenu implementaciju. U skladu sa SOA, individualni koraci sekvence izvršenja se implementiraju pomoću odgovarajućih operacija servisa. Za implementaciju sekvence servisa mogu da se koriste različite tehnike. Preporuka je da se orkestracija relativno jednostavne kompozicije servisa implementira na nivou kompozitnih servisa, na primer dodavanjem specifičnog Java koda. Za kreiranje kompleksnijih orkestracija se predlaže da se prvo kreira vizuelna reprezentacija sekvence poslovnog procesa, pa da se nakon toga generiše odgovarajući kod za izvršenje sekvence u nekom od BPM alata.

Poznati standardi za orkestraciju poslovnih procesa su BPMN za definisanje vizuelne reprezentacije sekvence i BPEL za njeno izvršenje (OMG BPMN 2.0, 2011). BPEL je izvršni jezik za procese koji je baziran na automatskoj orkestraciji veb servisa. On predstavlja XML jezik za programiranje koji može da se direktno izvrši u BPEL proces engine-u kao što je IBM WebSphere Process Server ili Oracle BPEL Proces Manager. Pored standardnog skupa grafičkih simbola koji su jednostavni i pogodni za vizuelizaciju i dokumentaciju poslovnog procesa, BPMN 2.0 verzija uvodi potpuno novu dimenziju. U pitanju je standardizovan meta-model i XML-bazirana serijalizacija formata za BPMN, koja pruža mogućnost izvršenja BPMN procesa. Definisana su dva formata za serijalizaciju: XML Metadata Interchange (XMI) i XML Schema Definition (XSD) (OMG BPMN 2.0, 2011). Prethodne verzije BPMN-a nisu bile izvršne. Iako na tržištu postoje BPM alati koji mogu da izvrše procese koji su modelovani u BPMN1.x verziji, oni koriste samo specifikaciju modela, dok za implementaciju koriste svoje specifične lokalne implementacione jezike ili BPEL (Silver, 2011). BPEL je podržan od strane vodećih *middleware* vendora i nastaviće da ima bitnu ulogu u SOA bez obzira na BPMN2.0.

5.5. Kriterijumi za validaciju specifikacije aspekata interoperabilnosti

U skladu sa preporukama za proces prikupljanja zahteva prvo je sprovedeno prikupljanje zahteva za interoperabilnošću, a zatim i njihova analiza (Abran, Moore, Bourque, & Dupuis, 2004).

5.5.1. Prikupljanje i analiza zahteva za interoperabilnošću

Bazu za identifikaciju *zahteva za interoperabilnošću* u ovoj tezi čine dve osnovne grupe zahteva: (1) zahtevi koji su identifikovani na osnovu analize postojeće naučne literature iz oblasti interoperabilnosti informacionih sistema i (2) zahtevi koji su dokumentovani u okviru vodećih evropskih istraživačkih projekata na temu interoperabilnosti. Učešće na ATHENA projektu i različitim konferencijama na temu interoperabilnosti je omogućilo razmenu znanja sa istraživačima i industrijskim stručnjacima iz oblasti i bilo je od velike koristi za sagledavanje aktuelnih rešenja i zahteva za interoperabilnošću.

Nekoliko izvora dokumentovanih interoperabilnih zahteva je uzeto u obzir: UEML projekat (IST-2001-34229), IDEAS projekat (IST-2001-37368) i ATHENA projekat (Project No 507849). U okviru **UEML projekta** identifikovano je 240 zahteva koji opisuju potrebe za interoperabilnošću na nivou poslovnog sistema kao celine. Od ukupnog broja, 55 zahteva je relevantno da realizaciju interoperabilnosti između poslovnih partnera (ATHENA A1.2.1., 2005). **IDEAS projekat** je definisao 57 opštih interoperabilih zahteva koji su u prvoj fazi klasifikovani i apstrahovani u skladu za aspektima interoperabilnosti. Polazan skup zahteva je filtriran i odabran je 21 zahtev koji je sačuvan u DRDS repozitorijumu zahteva (ATHENA A1.2.1., 2005). Zahtevi koji su identifikovani na osnovu industrijskih scenarija u **ATHENA projektu** su nazvani *specifični* zahtevi. U prvom koraku je izvršena selekcija samo onih zaheva koji se tiču interoperabilnosti. U drugom koraku se vrši selekcija *opštih* zahteva koji su prikupljeni iz eksternih izvora, uključujući i UEML projekat. Specifični i opšti zahtevi su elaborirani i kreiran je skup ATHENA interoperabilnih zahteva koji su sačuvani u Dynamic Requirements Definition System (DRDS) (ATHENA A1.2.1., 2005).

Problemi identifikacije i klasifikacije zahteva za interoperaabilnošću su analizirani u sledećim radovima koji su bazirani na istraživanju u okviru INTEROP projekta (A. Berre & Doumeingts, 2004; Petit, Krogstie, & Sindre, 2004). Identifikovani problemi i zahtevi za interoperabilnošću su opisani u formi paterna (Alexander, 1999; Coplien & Schmidt, 1995) u (Ottosson, 2005). Zahtevi za interoperabilnošću kolaborativnih poslovnih procesa su detaljno analizirani u (Lippe et al., 2005; Ziemann et al., 2007).

Kao rezultat detaljne analize zahteva za interoperabilnošću iz prethodno navedenih izvora, u skladu sa predmetom istraživanja disertacije, definisane su četiri klase zahteva (ATHENA D.A2.1, 2005; Lippe et al., 2005; Ziemann et al., 2007):

- Klasa A: Podrška za apstrakciju CBP procesa
- Klasa B: Podrška za dizajn CBP procesa
- Klasa C: Podrška za efikasnu kompoziciju CBP procesa
- Klasa D: Podrška globalne poslovne informacione šeme

Sledi opis glavnih karakteristike svake od četiri osnovne klase zahteva, kao i identifikacija relevantnih podzahteva.

5.5.1.1. Klasa A: Podrška za apstrakciju CBP procesa

Pristup modelovanju CBP procesa treba da obezbedi koncept koji će omogućiti apstrakciju internih poslovnih procesa i kreiranje potrebnih interfejsa ka spoljnom svetu. Apstrakcija privatnih procesa se odnosi na mogućnost selektivnog prikazivanja, odnosno skrivanja detalja privatnih procesa. Ovaj zahtev se mapira na sledeći skup detaljnijih zahteva:

- **A-1: Mehanizam za zaštitu privatnosti.** Korisnici moraju da imaju mogućnost da prikažu interne informacije o privatnim podacima i procesima na različitom nivou apstrakcije[2]. Bitno je da model kolaborativnog poslovnog procesa obezbedi partnerima samo one informacije koje su neophodne za

[2] Videti (ATHENA D.A2.1, 2005), strana 69, GR5

zajedničko izvršenje procesa, odnosno treba otkriti minimum internih informacija[3].

- **A-2: Skalabilno prikazivanje privatnih informacija.** Partneri treba da imaju mogućnost selektivnog prikazivanja, odnosno skrivanja privatnih informacija pri kolaboraciji sa različitim partnerima u zavisnosti od vrste ugovora ili nivoa poverenja[4].

- **A-3. Mehanizam za repezentaciju interne/privatne i eksterne/javne apstrakcije procesa[5].**

- **A-4: Mogućnost adaptacije bez izmena internog procesa[6].** Partneri moraju da imaju mogućnost da fleksibilno učestvuju u interakcijama sa različitim učesnicima kolaborativnog procesa, bez potrebe za izmenom internog procesa.

5.5.1.2. Klasa B: Podrška za dizajn CBP procesa

Dizajn CBP procesa mora da bude konceptualan, nezavistan od operativnog poslovnog procesa i platformski-specifičnog implementacionog nivoa. Za nivo CBP dizajna su karakteristični sledeći zahtevi:

- **B-1: Dizajn CBP procesa treba da bude konceptualan** i nezavistan od platforme i konkretnog izvšnog okruženja. U kontekstu MDA treba da bude u skladu sa definicijom PIM modela.[7]

[3] Videti (Ziemann, 2010), strana 39

[4] Videti (ATHENA D.A2.1, 2005), strana 69, GR6 i (Lippe, Greiner, & Barros, 2005), strana ?

[5] (Lippe et al., 2005)

[6] Videti (ATHENA D.A2.1, 2005), strana 69, GR7 i (Ziemann, 2010), strana 39

[7] (ATHENA D.A2.1, 2005), strana 73, GR 38

- **B-2:** Specifikacija CBP procesa treba da: (1) bude **pogodna za lako razumevanje i korišćenje od strane poslovnih korisnika**, (2) ima dovoljnu ekspresivnu snagu, i (3) jasnu, formalnu semantiku[8].

- **B-3: Mogućnost grafičke/vizuelne specifikacije.** Tehnike za modelovanje CBP procesa treba da podrže upotrebu visoko efikasnih grafičkih/vizuelnih korisničkih interfejsa. Ukoliko kompletna specifikacija CBP procesa, nije moguća primenom odgovarajuće grafičke tehnike za modelovanje, potrebno je uključiti druge tehnike koje nisu grafičke prirode kao što je na primer tekstualni opis. Potrebno je voditi računa o verzijama i ažurnosti čitave specifikacije[9].

- **B-4: Rad sa različitim formatima dokumenata.** Sistem treba da omogući reprezentaciju i upravljanje različitim formatima dokumenata. Sistem treba da omogući razmenu dokumenata sa različitim partnerima, bez obzira na format dokumenata koji koriste. Na primer, za scenario koji uključuje slanje poruke koja predstavlja dokument narudžbenica, jedna aplikacija može da koristi EDIFACT 850 Order standard, druga UBL PurchaseOrder specifikaciju, a treća OAGIS PurchaseOrder specifikaciju. Ove tri standardne specifikacije na različite načine definišu dokument narudžbenica, koristeći različitu terminologiju, strukturnu organizaciju podataka, reprezentaciju podataka i sintaksu[10].

- **B-5: Mogućnost reprezentacije različitih organizacionih jedinica i uloga partnera.** Imajući u vidu činjenicu da u kolaborativnom procesu učestvuju različiti partneri, bitna je mogućnost reperezentacije organizacionih jedinica

[8] (ATHENA D.A2.1, 2005), strana 73, GR 39

[9] (ATHENA D.A2.1, 2005), strana 72, GR 31

[10] (ATHENA D.A2.1, 2005), strana 71, GR 24

i njihovih veza sa CBP procesom. Termin „uloga“ se koristi da označi tip organizacione jedinice sa jasno definisanim kvalifikcijama i veštinama. [11]

- **B-6: Validacija CBP dizajna**. Poslovni korisnici bi trebalo da budu u mogućnosti da validiraju dizajn CBP procesa pomoću modela koji imaju dovoljnu ekspresivni snagu i koji su izvršni, donosno postoji mogućnost njihove simulacije.[12]

- **B-7: Podrška za analizu CBP procesa.** Tehnika za modelovanje CBP procesa treba da bude pogodna za automatsku verifikaciju sintaksnih i semantičkih grešaka tokom modelovanja.[13]

5.5.1.3. Klasa C: Podrška za efikasnu kompoziciju CBP procesa

Kompozicija CBP procesa treba da bude efikasna i laka. Ovaj zahtev se mapira na sledeći skup detaljnijih zahteva:

- **C-1: Mehanizam za kompoziciju CBP procesa.** Potreban je mehanizam koji će omogućiti efikasno i lako komponovanje CBP procesa koristeći komponente privatnih i javnih procesa partnera koji učestvuju u kolaboraciji[14].

- **C-2: Adekvatna reprezentacija informacionog toka podataka**. Potrebno je da postoji mogućnost reprezentacije informacionog toka izmedju partnera u CBP procesu, tako da budu jasno i precizno definisani potrebni ulazi i izlazi za svaku od pojedinačnih aktivnosti. Posebno je bitna mogućnost opisa parcijalnog ulaza koji je potreban svakom od partnera kako bi bio u mogućstvu da obavi svoj deo posla[15].

---

[11] (Jörg Ziemann, Thomas Matheis, & Jörn Freiheit, 2007),strana 24

[12] (ATHENA D.A2.1, 2005), strana 73, GR40

[13] (Jörg Ziemann et al., 2007), strana 24, (ATHENA D.A2.1, 2005), strana 73, GR 41

[14] (ATHENA D.A2.1, 2005), strana 74, GR 45

[15] (Lippe et al., 2005) i (Jörg Ziemann et al., 2007)

- **C-3: Opis CBP interfejsa.** Jezik za modelovanje treba da pruži mogućnost opisa CBP interfejsa. Bitno je da interfejsi budu opisani precizno i formalno, kako bi mogli da podrže automatsku verifikaciju interakcione sekvence , što je naročito bitno za veće scenarije.[16]

5.5.1.4. Klasa D: Podrška globalne poslovne informacione šeme

Treba da bude podržana globalna poslovna informaciona šema koja predstavlja zajedničku referencu za specifikaciju poruka koje se razmenjuju u CBP procesu. Sledi lista detaljnijih zahteva koji se odnose na definiciju i korišćenje globalne informacione šeme:

- **D-1: Definicija opšte strukture i formata podataka.** Globalna poslovna informaciona šema treba da podrži jedinstven način specifikacije strukture i formata podataka poruka koje se razmenjuju između partnera u CBP procesu.

  Sve poruke i poslovni dokumenti koji se koriste u CBP procesu moraju da budu definisani u skladu sa globalnom informacionom šemom[17].

- **D-2: Definicija globalnih poslovnih objektnih tipova, relacija i ograničenja.** Globalna poslovna informaciona šema treba da definiše potrebne globalne objektne tipove, veze i organičenja u skladu sa konkretnim CBP procesom.

  Na taj način će partneri koji učestvuju u CBP procesu biti u stanju da stuktuiraju poslovne poruke na konceptualnom nivou koji je nezavistan od implementacije. Drugim rečima, globalna šema obezbeđuje nezavisnu bazu za mapiranje prema specifičnim lokalnim reprezenatcijama podataka svakog od partnera[18].

---

[16] (Jörg Ziemann et al., 2007), (Lippe et al., 2005)

[17] (Lippe et al., 2005), (ATHENA D.A2.1, 2005),strana 74, GR 49, (Jörg Ziemann et al., 2007)

[18] (Lippe et al., 2005), (ATHENA D.A2.1, 2005), strana 74, GR 50, (Jörg Ziemann et al., 2007)

- **D-3: Definicija specifičnih poslovnih objekata svakog od partnera.** Neophodno je da globalna poslovna informaciona šema podrži definiciju specifičnih poslovnih objekata, u slučaju da partner odluči da ih podeli sa ostalim učesnicima CBP procesa.

  Svaki partner ima pravo da definiše dostupnost i nivo vidljivosti svojih podataka[19].

- **D-4: Reflektovanje internih organizacionih ograničenja.** Globalna poslovna šema treba da podrži specifikaciju eksterno izloženih organizacionih struktura (uloga).

  Globalna poslovna informaciona šema treba da ima mogućnost da reflektuje interna organizaciona ograničenja eksterno. Bitno je da CBP specifikacija sadrži opis zahteva koji se tiču organizacionih uloga koje su odgovorne za izvršenje CBP aktvnosti, što omogućava bolje razumevanje procesa od strane partnera[20].

- **D-5: Mogućnost struktuiranja poruka u skadu sa poslovnim domenom.** Poslovne poruke koje se razmenjuju između partnera u CBP procesu, treba da sadrže samo relevantne podatke za partnere u skladu sa domenom CBP procesa[21].

Identifikovani zahtevi su zbirno prikazani i klasifikovani prema aspektima interoperabilnosti za koje se u disertaciji predlaže način specifikacije (tabela 5.7).

[19] (ATHENA D.A2.1, 2005), strana 74, GR 51, (Lippe et al., 2005), (Jörg Ziemann et al., 2007)

[20] (Lippe et al., 2005)

[21] (ATHENA D.A2.1, 2005), strana 74, GR 48

Tabela 5.7 . Klasifikacija zahteva za interoperabilnošću u skladu sa aspektima interoperabilnosti

		Proces	Servis	Informacija
Podrška za apstrakciju CBP procesa	A-1: Mehanizam za zaštitu privatnosti	●		●
	A-2: Skalabilno prikazivanje privatnih informacija			●
	A-3: Mehanizam za reprezentaciju interne/privatne i eksterne/javne apstrakcije procesa	●		
	A-4: Mogućnost adaptacije bez izmena internog procesa	●		
Podrška za dizajn CBP procesa	B-1: Dizajn treba da bude konceptualan i nezavistan od platforme	●		●
	B-2: Pogodan za lako korišćenje i razumevanje od strane poslovnih korisnika	●		●
	B-3: Mogućnost grafičke/vizuelne specifikacije	●		●
	B-4: Rad sa različitim formatima dokumenata			●
	B-5: Mogućnost reprezentacije različitih organizacionih jedinica i uloga partnera	●		
	B-6: Validacija CBP dizajna	●		●
	B-7: Podrška za analizu CBP procesa	●		●
Podrška za kompoziciju CBP	C-1: Mehanizam za kompoziciju CBP procesa	●		
	C-2: Adekvatna reprezentacija informacionog toka podataka			●
	C-3: Opis CBP interfejsa		●	
Podrška globalne inf. šeme	D-1: Defnicija opšte stukture i formata podataka			●
	D-2: Definicija globalnih poslovnih objektnih tipova, relacija i ograničenja			●
	D-3: Definicija specifičnih poslovnih objekata svakog od partnera			●
	D-4: Reflektovanje internih organizacionih ograničenja			●
	D-5: Mogućnost stuktuiranja poruka u skladu sa poslovnim domenom			●

●/*prazno: Zahtevi su potpuno/nisu podržani*

Klasifikacija zahteva za interoperabilnošću u skladu sa predloženim fazama pristupa za specifikaciju aspekata interoperabilnosti je prikazana u tabeli 5.8.

Tabela 5.8 . Klasifikacija zahteva za interoperabilnošću u skladu sa fazama sistemsko-teorijskog životnog ciklusa

		Identifikacija	Realizacija	Implementacija
Podrška za apstrakciju CBP procesa	A-1: Mehanizam za zaštitu privatnosti	●	●	
	A-2: Skalabilno prikazivanje privatnih informacija		●	
	A-3: Mehanizam za reprezentaciju interne/privatne i eksterne/javne apstrakcije procesa	●		
	A-4: Mogućnost adaptacije bez izmena internog procesa	●		
Podrška za dizajn CBP procesa	B-1: Dizajn treba da bude konceptualan i nezavistan od platforme	●	●	
	B-2: Pogodan za lako korišćenje i razumevanje od strane poslovnih korisnika	●	●	
	B-3: Mogućnost grafičke/vizuelne specifikacije	●	●	
	B-4: Rad sa različitim formatima dokumenata		●	
	B-5: Mogućnost reprezentacije različitih organizacionih jedinica i uloga partnera	●		
	B-6: Validacija CBP dizajna			●
	B-7: Podrška za analizu CBP procesa			●
Podrška za kompoziciju CBP	C-1: Mehanizam za kompoziciju CBP procesa		●	
	C-2: Adekvatna reprezentacija informacionog toka podataka		●	
	C-3: Opis CBP interfejsa		●	
Podrška globalne inf. šeme	D-1: Defnicija opšte stukture i formata podataka		●	
	D-2: Definicija globalnih poslovnih objektnih tipova, relacija i ograničenja		●	
	D-3: Definicija specifičnih poslovnih objekata svakog od partnera		●	
	D-4: Reflektovanje internih organizacionih ograničenja		●	
	D-5: Mogućnost stuktuiranja poruka u skladu sa poslovnim domenom		●	

●/*prazno: Zahtevi su potpuno/nisu podržani*

5.5.2. Kriterijumi za validaciju faze identifikacije aspekata interoperabilnosti

Na osnovu analize zahteva koji su u prethodnom poglavlju klasifikovani prema aspektima (tabela 5.7) i fazama pristupa koji se predlaže za njihovu specifikaciju (tabela 5.8) izvedeni su sledeći zaključci:

- Predloženi pristup za specifikaciju *aspekata procesa* u potpunosti podržava zahteve: A-1, A-3, A-4, B-1, B-2, B-3 i B-5.
- Predloženi pristup za specifikaciju *aspekata informacija* u potpunosti podržava zahteve: A-1, B-1, B-2 i B-3.

U fazi identifikacije je za specifikaciju poslovnih procesa odabrana BPMN 2.0 notacija. Kao mehanizam za proveru ispunjenosti zahteva je korišćen dovoljno kompleksan realan eKanban poslovni scenario. Na osnovu praktičnog iskustva tokom modelovanja eKanban poslovnog scenarija je pokazano da BPMN notacija poseduje potrebne modele i mehanizme za adekvatno zadovoljenje navedenih zahteva. Nije bilo potrebe za uvođenjem dodatnih modela, niti za definisanjem proširenja.

Zahtevi za podršku apstrakcije kolaborativnog poslovnog procesa (A-1, A-3 i A-4) su su uspešno modelovani primenom pravila za definisanje privatnog poslovnog procesa, i njegove javne reprezentacije. Jedna od osobina BPMN standarda zahvaljujući kojoj je postigao popularnost u praksi je činjenica da su modeli laki za razumevanje i korišćenje od strane poslovnih korisnika (B-1). Pored vizuelne specifikacije (B-3), BPMN 2.0 verzija standarda poseduje i XML-baziranu serijalizaciju izvršnog formata koji je nezavistan od platforme (B-1). Za reprezentaciju različitih organizacionih jednica, BPMN poseduje poseban koncept *pool*-a (B-5).

U fazi identifikacije se ne vrši detaljna specifikacija strukture poruka kolaborativnog poslovnog procesa, ali se predlaže definisanje globalne referentne ontologije. Budući da se za vizuelizaciju globalne referentne ontologije predlaže UML dijagram klasa omogućeno je lako razumevanje koncepata i relacija globalne referentne

ontologije (B-1, B-2). UML dijagram klasa omogućava konceptualni dizajn koji nije zavistan od platforme (B-1).

5.5.3. Kriterijumi za validaciju faze realizacije aspekata interoperabilnosti

Na osnovu analize relevantnih zaheva za fazu *realizacije* koji su u prethodnom poglavlju klasifikovani prema aspektima interoperabilnosti u tabeli 5.7, izvedeni su sledeći zaključci:

- Predloženi pristup za specifikaciju *aspekata procesa* u potpunosti podržava zahteve: A-1, B-1, B-2, B-3, C-1.
- Predloženi pristup za specifikaciju *aspekata informacija* u potpunosti podržava zahteve: A-1, A-2, B-1, B-2, B-3, B-4, C-2, D-1, D-2, D-3, D-4, D-5.
- Predloženi pristup za specifikaciju *aspekata servisa* u potpunosti podržava zahtev C-3.

U fazi *realizacije* predloženog pristupa za specifikaciju aspekata interoperabilnosti se vrši dalja razrada i detaljna specifikacija modela koji su specificirani u fazi *identifikacije*. Pri analizi ispunjenja zahteva, u razmatranje su uzete one karakteristike modela koje su specifične za fazu *realizacije*.

U fazi *realizacije* je omogućena veza privatnog kolaborativnog poslovnog procesa i referentne ontologije. Dodatna zaštita privatnosti kolaborativnog poslovnog procesa je omogućena uvođenjem koncepta pogleda nad strukturom standarnde poruke koja se razmenjuje u globalnoj kolaboraciji između poslovnih partnera (A-1). Dizajn poslovnog kolaborativnog procesa, koji je obogaćen konceptima referentne ontologije je i dalje nezavistan od konkretnog izvršnog okruženja (B-1, B-2). Izvršena je ekstenzija BPMN 2.0 meta-modela koja je omogućila da se obogati vizuelna reprezentacija kolaborativnog poslovnog procesa uvođenjem novih elemenata za reprezentaciju ontologije i relevantnih pogleda (B-3). Uvedena je dodatna semantika za označavanje aktivnosti kolaborativnog poslovnog procesa koje zahtevaju interoperabilnost i čija specifikacija podrazumeva primenu Servis Adapter paterna (C1).

Zahtevi koji su relevantni za aspekt informacija obuhvataju sve četiri definisane kategorije: A, B, C i D (tabela 5.7). Definisani UML View Profil poseduje mehanizam za filtriranje željenih podataka, tako da je moguće prilagoditi strukturu bilo koje poruke u skladu sa potrebama specifične kolaboracije (A-1, A-2). Definisani UML View Profil ima odgovarajuću grafičku reprezentaciju baziranu na dijagramu klasa, što olakšava komunikaciju i u njegovu definiciju mogu da se uključe i poslovni korisnici (B-2, B-3). Podržan je rad sa različitim formatima podataka, budući da UML View Profil omogućava vezu sa globalnom referentnom ontologijom (B-4). Adekvatna reprezentacija elemenata informacionog toka se odnosi na mogućnost specifikacije njihove strukture definisanjem odgovarajućih OCL pravila (C-2). UML View Profil je dizajniran imajući u vidu zahteve (D-1-D-5).

6. PRIMER PRIMENE PREDLOŽENOG PRISTUPA SPECIFIKACIJE ASPEKATA INTEROPERABILOSTI

U ovom poglavlju je prikazano korišćenje predloženog pristupa za specifikaciju aspekata interoperabilnosti na realnom primeru IV&I eKanban poslovnog sistema primenom FON Labis metodologije za projektovanje informacionih sistema i Larmanove metode. Izvršeno je poređenje i ukazano je na prednosti, odnosno nedostatke jednog pristupa u odnosu na drugi.

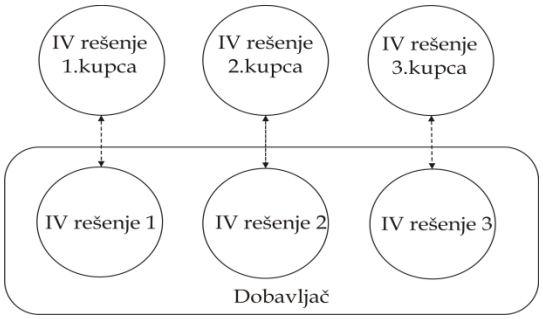
6.1. eKanban eksperimentalni B2B scenario

Kanban je poslovni proces koji koristi sistem signalizacije pomoću fizičkih kartica kako bi se regulisala potreba za materijalom u skladišnom poslovanju, a u skladu sa potrebama proizvodnje ili prodaje. EKanban proces predstavlja nadogradnju klasičnog kanbana, u kojem se razmenom elektronskih B2B poruka upravlja isporukom materijala.

U tezi je analiziran Automotive Industry Action Group (AIAG) eKanban (Electronic Kanban) poslovni proces. U prvom koraku je na osnovu analize IV&I eKanban standarda, prateće dokumentacije i niza intervjua sa AIAG timom eksperata izvršena funkcionalna specifikacija sistema primenom metode strukturne sistemske analize. Sledi verbalni opis eKanban procesa, koji je analiziran sa aspekta dobavljača materijala. Analiza je sprovedena sa ciljem da se omogući specifikacija i implementacija eKanban informacionog sistema dobavljača imajući u vidu bitne aspekte interoperabilnosti.

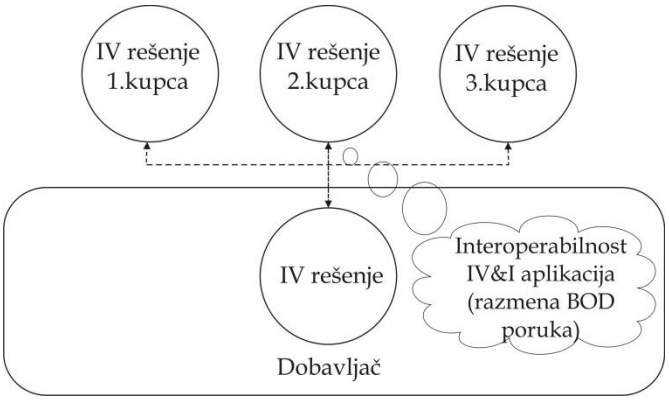
IV&I eKanban poslovni proces podrazumeva da se komunikacija između kupca i dobavljača vrši razmenom standardizovanih elektronskih poruka. Kupac i dobavljač imaju slobodu izbora odgovarajućeg IV&I alata, što dovodi do pojave problema interoperabilnosti. Dobavljač treba da bude u stanju da brzo i efikasno komunicira sa više različitih kupaca. Kupci koji koriste različite IV&I aplikacije sa specifičnim

interfejsima, u većini slučajeva koriste različite specifikacije poruka. Na primer, jedan kupac može da koristi EDIFACT standard, drugi UBL ili OAGIS specifikaciju poruka. Na slici 6.1 je ilustrovano da ukoliko različiti kupci imaju različite interfejse IV&I aplikacija, dobavljač mora da implementira različita rešenja za svakog od njih.



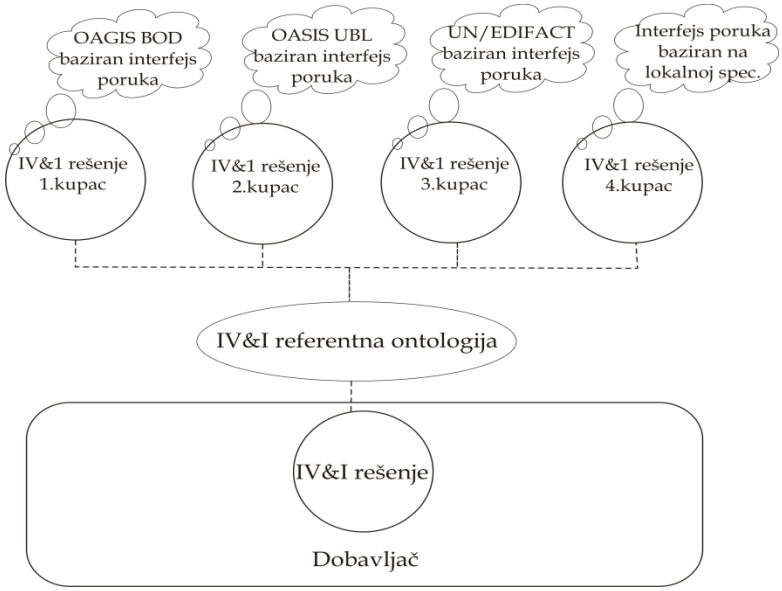
Slika 6.1 eKanban sistem koji ne podržava interoperabilnost

Interoperabilno rešenje, koje je prikazano na slici 6.2, podrazumeva da dobavljač nema potrebu da implementira različite interfejse i može efikasno da komunicira sa svakim od kupaca. Ovo rešenje predstavlja tradicionalan pristup rešavanja problema interoperabilnosti, s obzirom na činjenicu da svi učesnici koriste Business Object Document (BOD) format poruke.



Slika 6.2 Interoperabilno eKanban rešenje (tradicionalni pristup)

Na slici 6.3 je prikazan popularan pristup rešavanju problema interoperabilnosti, upotrebom zajedničke globalne referentne ontologije.



Slika 6.3 Kompleksnost B2B interoperabilnosti i upotreba zajedničke referentne ontologije

EKanban elektronski proces se bazira na specifičnim konceptima, koji se razlikuju u odnosu na osnovne koncepte materijalnog kanbana. Pre nego što nastavimo sa opisom eKanban procesa, objasnićemo dva osnovna koncepta: Kanban kontejner i Kanban petlja (eng. kanban loop). **Kanban kontejner** predstavlja referentnu jedinicu za isporuku ili potrošnju artikala. Za eKanban je karakteristično da se artikli grupišu u Kanban kontejner koji ima standardno pakovanje, što u velikoj meri olakšava upravljanje procesom. Standardno pakovanje ima definisanju vrstu i količinu artikala. Drugim rečima, Kanban kontejner je paket koji ima standardno pakovanje. Svaki Kanban kontejner ima svoj jedinstveni identifikator i status. Status Kanban kontejnera može da bude: "prazan", "pun", "autorizovan" ili "isporučen". Kada kupac potroši Kanban kontejner, on je u obavezi da o tome obavesti dobavljača koji će status Kanban kontejnera u svom sistemu ažurirati na "prazan". Kada kupac autorizuje Kanban kontejner za nabavku, njegov status prelazi u "autorizovan". Nakon isporuke Kanban kontejnera dobavljač postavlja status na "isporučen", sve dok ne primi potvrdnu informaciju od kupca da je Kanban kontejner primljen i da treba postaviti status na "pun".

Drugi bitan pojam je **Kanban petlja**, koja može da se shvati kao osnovna jedinica upravljanja u eKanban poslovnom procesu. Za isporuku svakog pojedinačnog artikla, kupac i dobavljač moraju da unapred dogovore način i uslove isporuke koji

se u eKanban procesu specificiraju pomoću parametara Kanban petlje. Za svaki artikal se definiše posebna Kanban petlja koja može da ima jedan ili više Kanban kontejnera. Kanban petlja može da bude identifikovana eksplicitno, ili implicitno pomoću identifikatora Artikla i identifikatora učesnika eKanban procesa. Eksplicitni identifikator može da se koristi kao prefiks ispred identifikatora Kanban kontejnera na labeli fizičkog Kanban paketa. Ostali bitni parametri Kanban petlje su: broj Kanban kontejnera, i količina artikala u Kanban kontejneru. Napomenućemo da se jedan isti artikal može isporučiti istom kupcu na različitu lokaciju pod različitim uslovima. Stoga se Kanban petlja definiše za kupca i za svaku lokaciju pojedinačno. U praksi se isporuka Kanban petlje vrši na osnovu *Rasporeda Isporuke* (*eng. Shippment Schedule*), gde svaka *Stavka rasporeda (eng. Schedule Line)* odgovara jednoj Kanban petlji.

6.1.1. Verbalni opis problema

Pre početka bilo kog vida saradnje kupac i dobavljač su dužni da se unapred dogovore o uslovima saradnje i da definišu pravila poslovanja. U prvom koraku se potpisivanjem *Ugovora* između kupca, dobavljača i prevoznika preciziraju finansijski uslovi i ostali bitni elementi *Ugovora*. Nakon ugovaranja saradnje, pristupa se planiranju potražnje. Osnova za predviđanje potražnje je dokument *Predviđanje Potražnje (eng. Material Release)* koji se razmenjuje između kupca i dobavljača. Na osnovu uslova saradnje koji su određeni *Ugovorom* i predviđene potražnje, definišu se ostala bitna poslovna pravila saradnje.

Dokument *Polisa Naručivanja Kanban Kontejnera* (eng. *Kanban Ordering Policy)* prvo definiše karakteristike Kanban petlje, zatim se definiše događaj koji ukazuje na to da je došlo do potrošnje Kanban kontejnera. Prilikom preciziranja pravila saradnje je bitno da se definiše *Metodologija za Numerisanje Kanban kontejnera*, koja bi trebalo da jedinstveno identifikuje Kanban kartu u okviru Kanban petlje. Identifikator Kanban kontejnera (Kanban ID), je nezavistan od šifre artikla koja se evidentira posebno. Pored toga je bitno da se učesnici eKanban procesa dogovore o poljima koja su obavezna, odnosno opciona za svaku pojedinačnu elektronsku Kanban transakciju.

Nakon definisanja i prihvatanja uslova i pravila od strane svih učesnika eKanban procesa, može da se otpočne sa praćenjem i realizacijom eKanban procesa. Proces praktično počinje potrošnjom Kanban kontejnera od strane kupca. Naime, kada potroši Kanban kontejner, kupac je dužan da generiše signal, na primer skeniranjem labele Kanban kontejnera. Aplikacija na stani kupca, koja može da bude proizvoljno implementirana, treba da isporuči elektronski signal odgovarajućoj aplikaciji na strani dobavljača. Nakon prijema signala da je Kanban potrošen, u sistemu je potrebno njegov status postaviti na "prazan". Bitno je obratiti pažnju i identifikovati moguće izuzetke, kao što je na primer notifikacija hitne isporuke Kanban kontejnera.

U narednom koraku, dobavljač čeka na autorizaciju praznog Kanban kontejnera od strane kupca. Naime, on nema pravo da isporuči Kanban kontejner, sve dok ga kupac za to eksplicitno ne autorizuje. Nakon prijema signala za autorizaciju, status Kanban kontejnera se ažurira na "autorizovan", i dobavljač počinje sa pripremama za isporuku. Kada će Kupac nakon potrošnje Kanban kontejnera da autorizuje njegovu isporuku zavisi od uslova koji se definisani pre početka saradnje. Na primer, može da bude dogovoreno da je kupac u obavezi da pošalje signal za autorizaciju odmah nakon potrošnje Kanban kontejnera, ili je predviđeno automatsko generisnje signala periodično u toku dana za sve Kanban kontejnere koji su u međuvremenu potrošeni. Prilikom autorizacije je bitno precizirati i detalje isporuke. Moguća su dva scenarija: (1) ukoliko kupac diktira uslove ispruke on je dužan da zajedno sa signalom pošalje i *Raspored Isporuke* sa željenim datumom isporuke i ostalim bitnim instrukcijama; (2) ako je organizacija tansporta u nadležnosti dobavljača kupac šalje samo signal za autorizaciju. Nakon prijema signala za autorizaciju, kupac može da krene sa pripremama za isporuku. Nivo priprema koje je dobavljač potrebno da obavi, zavisi od konkretnog slučaja. Na primer, ukoliko su artikli već izrađeni dobavljač treba da organizuje njihov transport iz skladišta. Komplikovaniji slučaj, gde dobavljač treba da organizuje *just-in-time* proizvodnju u skladu sa rokovima koje postavlja kupac, neće biti razmatran u okviru eksperimentalnog scenarija. Ukoliko dobavljač diktira uslove isporuke, on određuje i datum slanja na osnovu pravila koja su prethodno dogovorena između učesnika eKanban procesa. U našem primeru kupac diktira uslove isporuke i zajedno sa signalom za autorizciju šalje *Raspored Isporuke*.

Posebno treba voditi računa o mogućim izuzecima. Na primer, ukoliko dođe do pada obima proizvodnje, potrebno je povući odgovarajuće Kanban signale iz sistema. Jedna opcija je da se Kanban signali zadrže, ali da se uvede novi status "zadržan" (eng. hold), što pruža mogućnost da se Kanban kontejner samo privremeno ukloni iz Kanban petlje i ponovo aktivira kada se obim proizvodnje vrati na normalu.

Nakon autorizacije Kanban kontejnera od strane kupca, dobavljač pristupa organizaciji isporuke. Organizacija isporuke zavisi od toga da li kupac ili dobavljač diktira uslove isporuke. U našem slučaju, budući da kupac diktira uslove isporuke, dobavljač je već upoznat sa datumom i ostalim zahtevima koji se tiču isporuke. Dobavljač će početi sa organizacijom isporuke u dogledno vreme kako bi ispoštovao ugovorene rokove. U slučaju da je dobavljač zadužen da vodi računa o rokovima isporuke, njegova je obaveza da za svaki Kanban signal čiji je status "autorizovan" proveri da li ispunjava uslove za isporuku. Pravila za isporuku Kanban kontejnera su dogovorena pre početka saradnje i mogu da budu bazirana na različitim principima, na primer: (1) fiksni vremenski interval, (2) veličina pošiljke, (3) "or" uslov ili (4) "and" uslov. Nakon identifikacije Kanban kontejnera koje treba isporučiti, dobavljač izvršava dva paralelna zadatka: (1) prikuplja Kanban kontejnere koje treba isporučiti, i popunjava prateću dokumentaciju koja je predviđena ugovorom i (2) priprema instrukcije za isporuku i prateće liste materijala. Ukoliko dobavljač upravlja isporukom, prevozniku šalje *Zahtev za Isporukom (eng. Request for Conveyance Signal)* kako bi ga informisao o obimu, težini, rokovima isporuke, itd. Kada prazan kamion stigne do dobavljača, počinje utovar. Kanban kontejneri koji su utovareni treba da budu verifikovani u skladu sa listom materijala.

Za fizički transport robe je zadužen prevoznik. U slučaju kada dobavljač diktira uslove isporuke, on će prevozniku poslati *Zahtev za Prevoz Robe (eng. Request for Conveyance Signal)*. U suprotnom, kupac šalje raspored isporuke istovremeno i dobavljaču i isporučiocu. Za verifikaciju isporuke može da bude zadužen dobavljač ili prevoznik. Prilikom verifikacije je bitno identifikovati i evidentirati sve izuzetke. Nakon utovara svih Kanban kontejnera, prevoznik kreće ka kupcu a dobavljač

isovremeno generiše i šalje *Advanced Shippment Notice (ASN)*. Nakon slanja ASN-a kupcu, status Kanban kontejnera se postavlja na "isporučen".

Po prijemu Kanban kontejnera, kupac kontroliše pošiljku i proverava stvarno stanje u skladu sa ASN-om. Kupac evidentira sve izuzetke o kojima treba da obavesti dobavljača. Neki od čestih izuzetaka su: Kanban kontejner koji je poslat nije evidentiran na ASN-u, primljeni artikli su zadržani na inspekciji ili ne zadovoljavaju propisane uslove kvaliteta, količina koja je isporučena ne odgovara zahtevu kupca, itd. Nakon evidencije svih izuzetaka, kupac šalje poruku dobavljaču da je pošiljka primljena, i status Kanban kontejnera se u oba sistema postavlja na "pun".

6.2. FON Labis metodologija za projektovanje informacionih sistema

U ovom poglavlju prikazana je primena predloženog metodološkog pristupa za specifikaciju aspekata interoperabilnosti koristeći FON Labis metodologiju za projektovanje informacionih sistema, na primeru IV&I eKanban poslovnog procesa.

6.2.1. Identifikacija aspekata interoperabilnosti

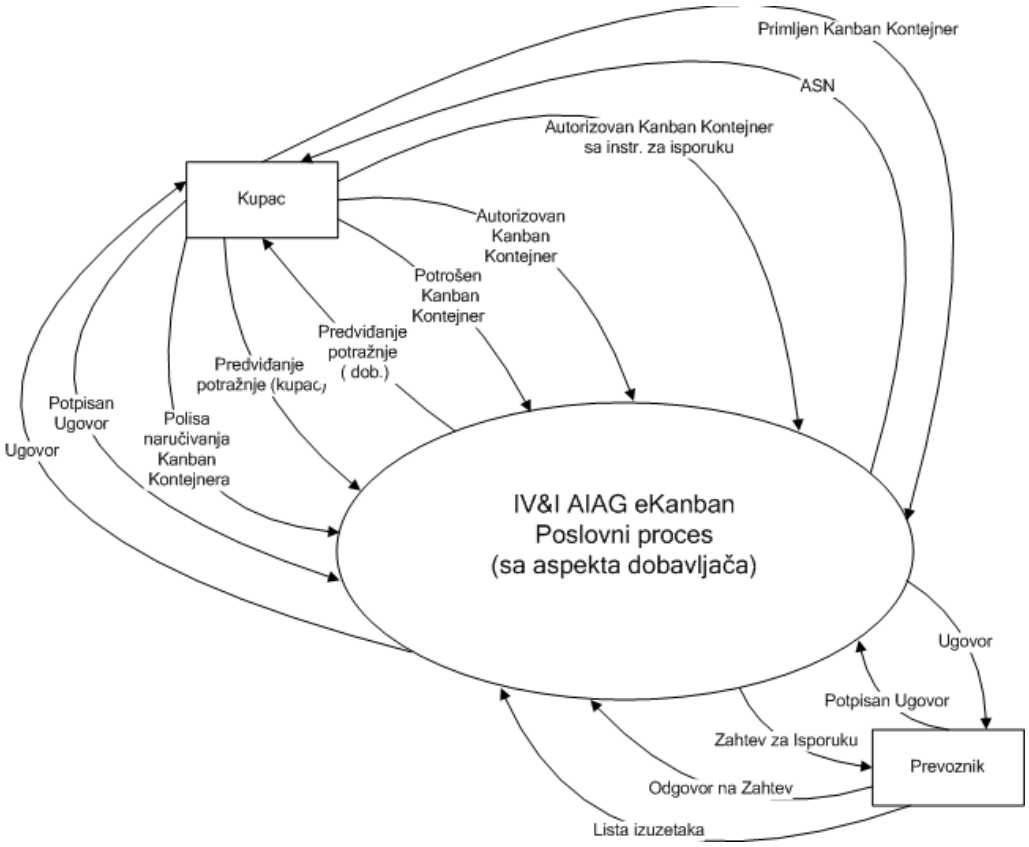
6.2.1.1. Identifikacija zahteva za interoperabilnošću

U prvom koraku predložene metodologije za specifikaciju aspekata interoperabilnosti je potrebno izvršiti identifikaciju zahteva za interoperabilnošću. Dva osnovna zahteva za interoperabilnošću se odnose na identifikaciju esencijalnih poslovnih procesa koji treba da budu interoperabilni i partnera koji učestvuju u kolaboraciji.

Preduslov za identifikaciju zahteva za interoperabilnošću je definicija logičkog modela esencijalih poslovnih funkcija. Za identifikaciju esencijalnih poslovnih procesa FON Labis metodologija predlaže primenu Strukturne sistemske analize.

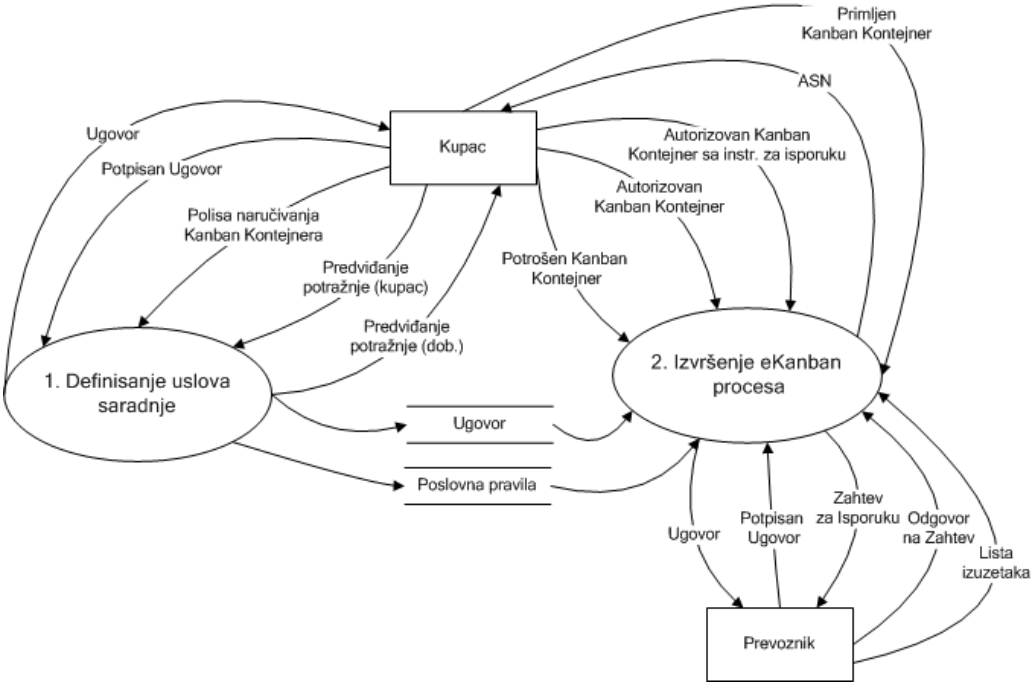
U ovom koraku je primenom SSA metode izvršena funkcionalna specifikacija IV&I eKanban poslovnog scenarija. Kolaboracija između poslovnih partnera IV&I eKanban poslovnog scenarija je analizirana iz ugla dobavljača. Odgovarajući dijagram konteksta IV&I eKanban poslovnog scenarija je prikazan na slici 6.4. Na dijagramu konteksta su definisana dva interfejsa, objekta van sistema sa kojima on

komunicira, Kupac i Prevoznik. Prikazani su i tokovi podataka, preko kojih se obavlja komunikacija između Dobavljača, Kupca i Prevoznika.



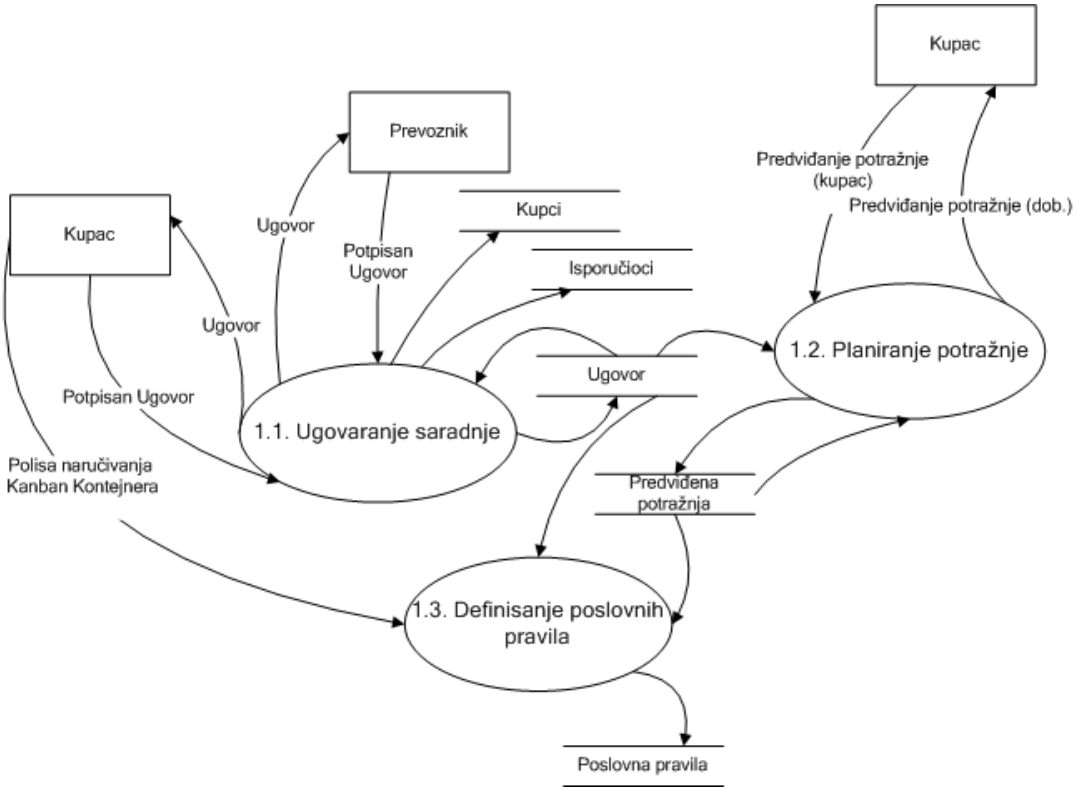
Slika 6.4 Dijagram konteksta IV&I AIAG eKanban poslovnog procesa

Nakon kreiranja dijagrama konteksta, primenom principa hijerarhijske dekompozicije sistema, na prvom nivou dekompozicije su identifikovane dve funkcije (slika 6.5): *Definisanje uslova saradnje* i *Izvršenje eKanban procesa*. Ove funkcije se zatim dalje dekomponuju, kao što je prikazano na slikama 6.6 i 6.7.



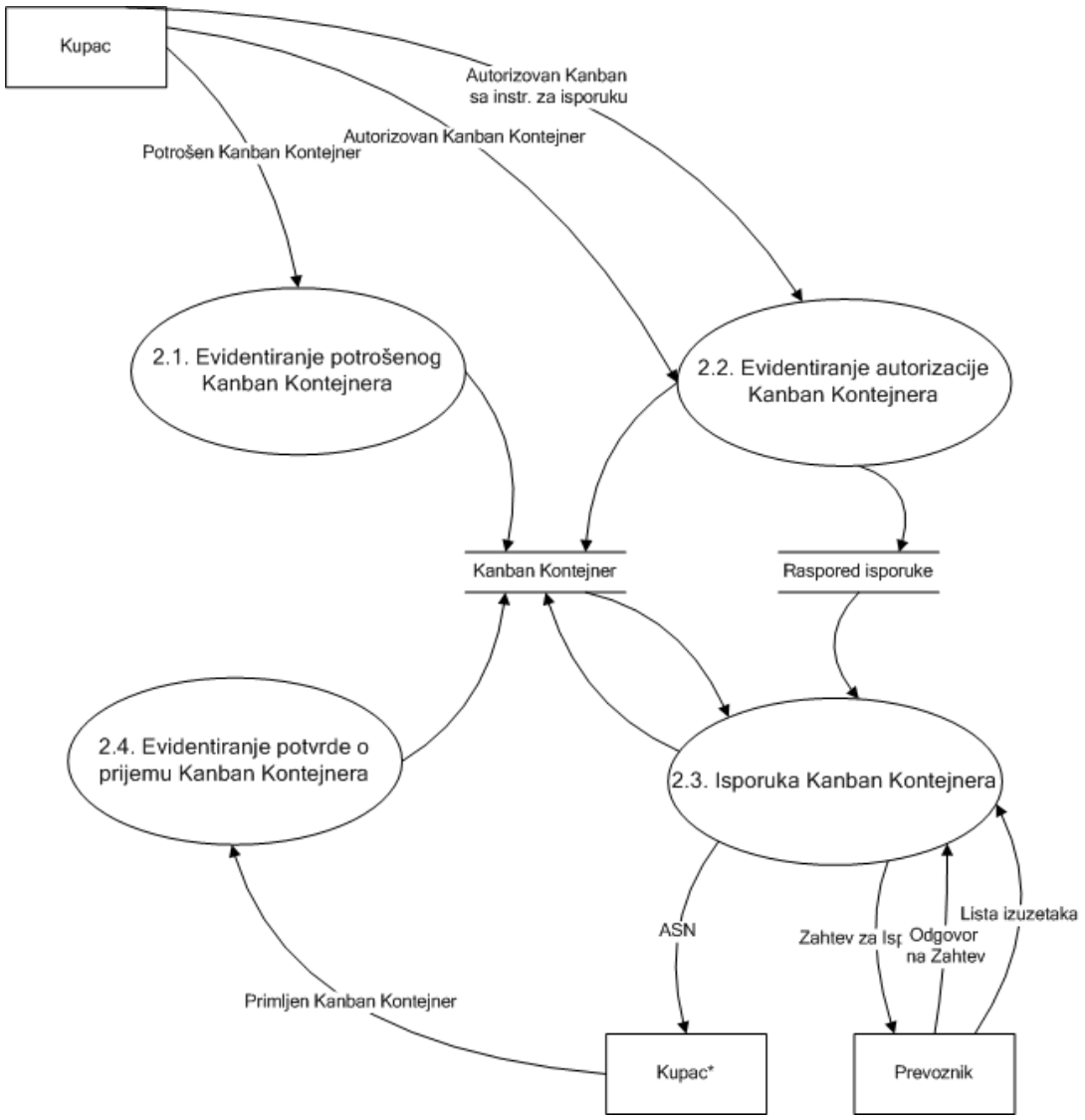
Slika 6.5 Prvi nivo dekompozicije IV&I AIAG eKanban poslovnog procesa

Funkcija *Definisanje uslova saradnje* se kao što je prikazano na Slici 6.6, dekomponuje na funkcije *Ugovaranje saradnje*, *Planiranje potražnje* i *Definisanje poslovnih pravila*. Iako funkcije na ovom nivou dekompozicije nisu primitivne, neće biti dalje dekomponovane. Definisanje uslova saradnje i pravila poslovanja je bitan preduslov uspešne saradnje, ali smo se u daljem radu opredelili za detaljnu analizu i razradu procesa *Izvršenje eKanbana*.



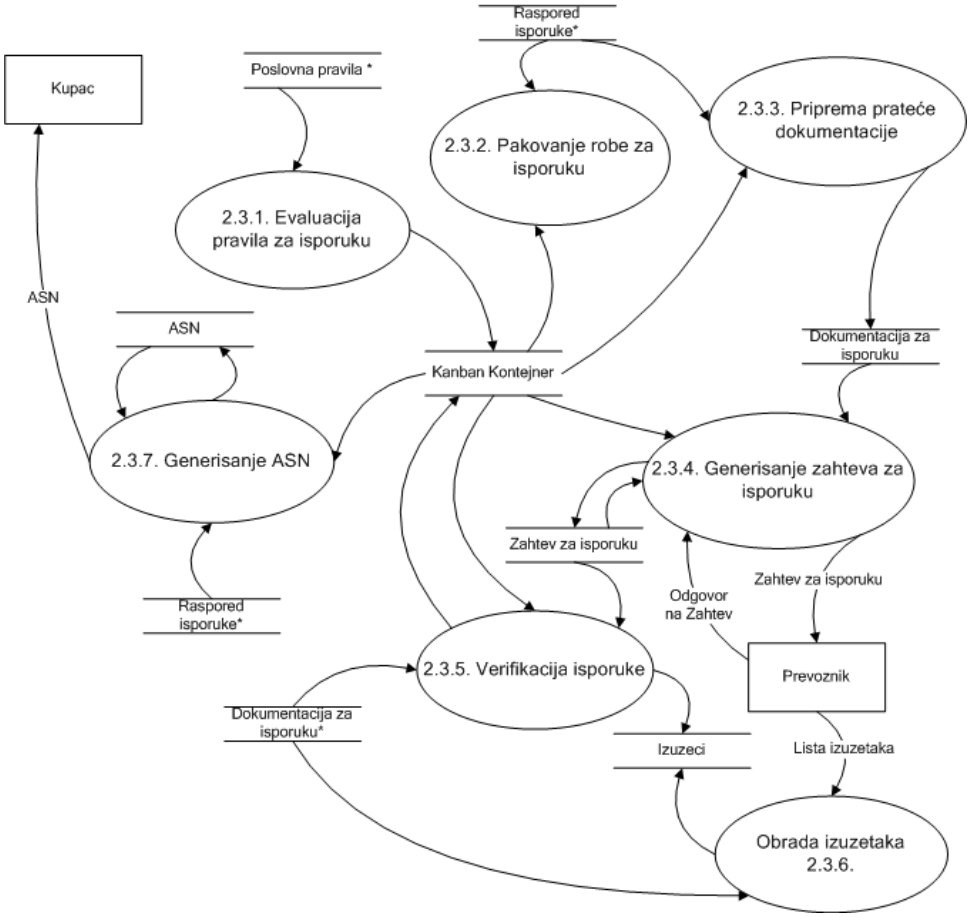
Slika 6.6 Drugi nivo dekomozicije – proces Definisanje uslova saradnje

Funkcija *Izvršenje eKanban procesa* se dekomponuje kao što je prikazano na Slici 6.7 na složenu poslovnu funkciju *Isporuka Kanban kontejnera* i tri primitivne funkcije *Evidentiranje potrošenog Kanban kontejnera*, *Evidentiranje autorizacije Kanban kontejnera* i *Evidentiranje potvrde o prijemu Kanban kontejnera*.



Slika 6.7 Drugi nivo dekompozicije – proces Izvršenje eKanban procesa

Funkcija *Isporuka Kanban kontejnera* se dekomponuje kao što je prikazano na slici 6.8 na primitivne funkcije *Evaluacija pravila za isporuku*, *Pakovanje robe za isporuku*, *Priprema prateće dokumentacije*, *Generisanje zahteva za isporuku*, *Obrada izuzetaka*, *Verifikacija isporuke* i *Generisanje ASN*.



Slika 6.8 Treći nivo dekompozicije – proces Isporuka Kanban kontejnera

Nakon funkcionalne specifikacije IV&I eKanban poslovnog sistema primenom metode SSA, i izrade logičkog modela funkcija moguće je izvršiti analizu interoperabilnosti. Na osnovu hijerarhijske dekompozicije sistema je lako identifikovati primitivne poslovne procese.

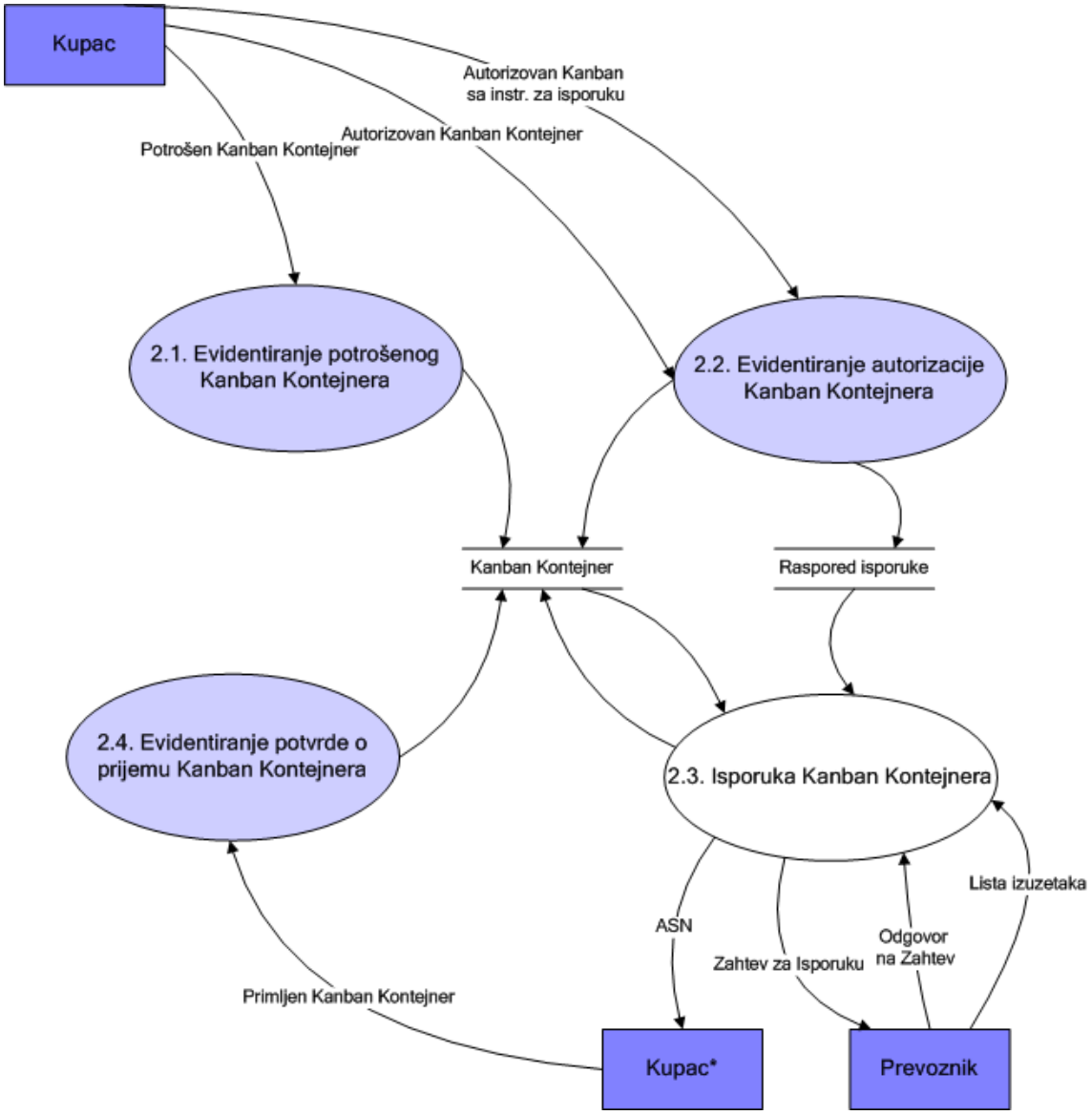
Polazni skup esencijalnih poslovnih procesa koji predstavlja osnovu za dalju analizu interoperabilnih zahteva obuhvata: skup procesa sa *DTP-a Izvršenje eKanban procesa* (slika 6.7) (Evidentiranje potrošenog Kanban kontejnera, Ediventiranje autorizacije Kanban kontejnera i Evidentiranje potvrde o prijemu Kanban kontejnera) i procese koji pripadaju *DTP-u Isporuka Kanban kontejnera* (slika 6.8)(Generisanje ASN, Generisanje zahteva za isporuku i Obrada izuzetaka).

Jedna od prednosti primene metode SSA je jednostavna identifikacija polaznog skupa esencijalnih poslovnih funkcija, koje odgovaraju primitivnim procesima SSA. Poslovni partneri su na dijagramima tokova podataka predstavljeni u vidu

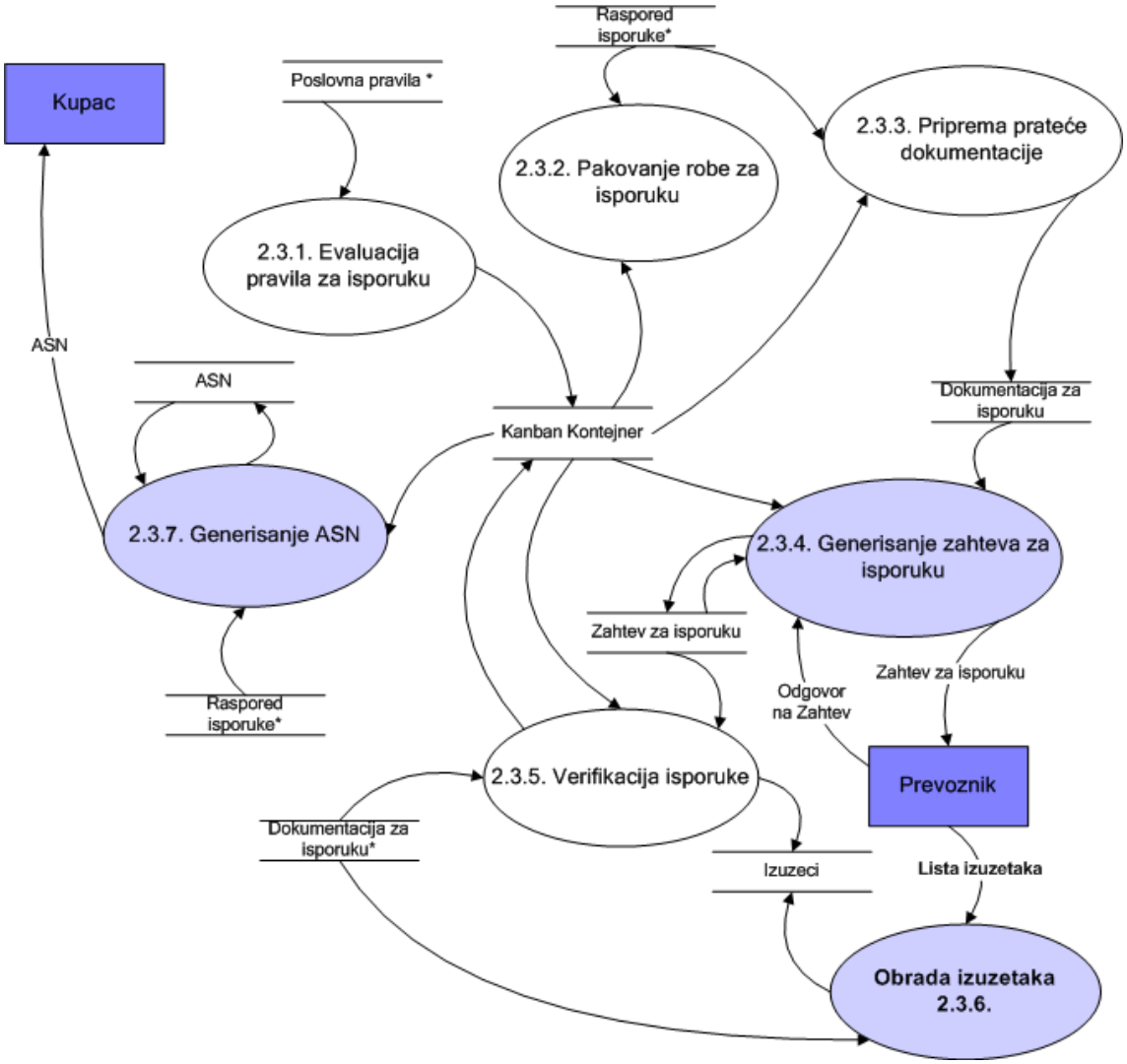
interfejsa, tako da ni njihova identifikacija ne predstavlja problem. Izlazni tok ka interfejsu, ili ulazni tok od interfejsa jasno ukazuje na postojanje kolaboracije, pa samim tim i potrebe za interoperabilnošću. Iako je komunikacija sa spoljnim interfejsima potreban uslov, ne možemo da predložimo definisanje opšteg pravila da ako primitivni proces komunicira sa interfejsom, on mora da podrži interoperabilnost. Uvođenje ovakvog pravila bi bilo suviše restriktivno, budući da informacioni sistem ne mora da implementira veze sa svim interfejsima, odnosno na dijagramu dekompozicije komunikacija može da bude prikazana radi lakšeg sagledavanja načina funkcionisana čitavog poslovnog sistema.

Za konkretan eKanban scenario, bitna je komunikacija sa oba poslovna partnera tako da svi interfejsi predstavljaju partnere koji učestvuju u kolaboraciji. Primera radi, ako bismo odlučili da je komunikacija sa prevoznikom van opsega posmatranog poslovnog sistema, u tom slučaju bi bilo potrebno da se iz polaznog skupa esencijalnih poslovnih procesa elimniše *Generisanje zahteva za isporuku*.

U skladu sa predlogom da se za potrebe specifikacije zahteva za interoperabilnošću izvrši minimalna moguća izmena postojećih modela i tehnika, interfejsi i primitivne poslovne funkcije koji su identifikovani kao zahtevi za interoperabilnošću su prikazni plavom bojom. Zahtevi za interoperabilnošću su specificirani na dijagramima tokova podataka koji su prikazani na slikama 6.9 i 6.10.



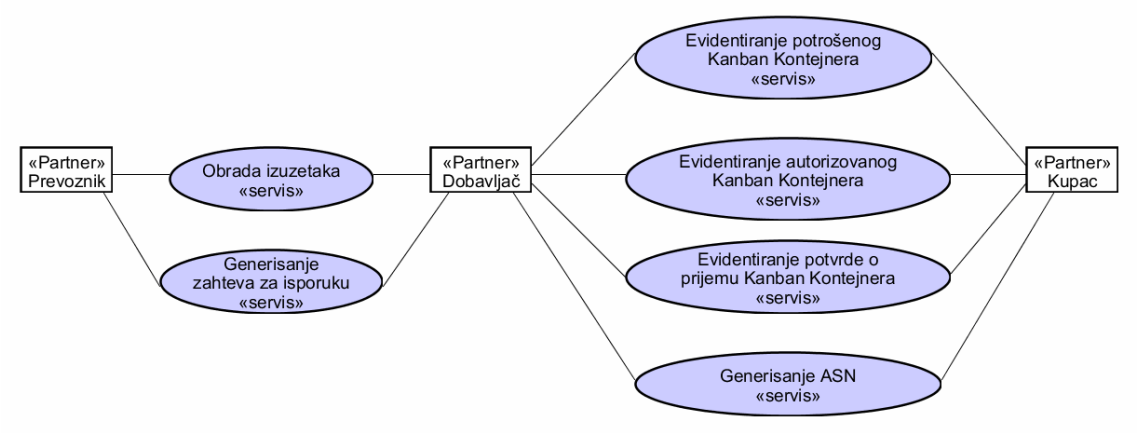
Slika 6.9 DTP-Izvršenje eKanban procesa – specifikacija zahteva za interoperabilnošću



Slika 6.10 DTP-Isporuka Kanban kontejnera – specifikacija zahteva za interoperabilnošću

Pored označavanja zahteva za interoperabilnošću na odgovarajućim dijagramima tokova podataka, u ovom koraku se predlaže i uvođenje novog dijagrama za specifikaciju interoperabilnih slučajeva korišćenja koji je prikazan na slici 6.11. Na predloženom dijagramu, koji predstavlja proširenje UML dijagrama slučajeva korišćenja, definisana su dva stereotipa:

- **<<servis>>** za označavanje primitivnih poslovnih procesa za koje je analizom utvrđeno da zahtevaju interoperabilnost;
- **<<partner>>** za specifikaciju kolaborativnih poslovnih partnera i definisanje veze sa relevantim slučajevima korišćenja.



Slika 6.11 Model interoperabilnih slučajeva korišćenja

Na Slici 6.11 su identifikovani partneri *Dobavljač*, *Kupac* i *Prevoznik* koji su prikazani stereotipom <<partner>>. Direktna asocijacija između dva partnera ili dva slučaja korišćenja nije dozvoljena. Prednost izrade *Modela interoperabilnih slučajeva korišćenja* je sistematičan prikaz esencijalnih poslovnih funkcija i mogućnost lake identifikacije kolaboracija između poslovnih partnera. U ovoj fazi analize sistema i specifikacije zahteva za interoperabilnošću, na dijagramu *Modela interoperabilnih slučajeva korišćenja* se uvodi stereotip <<servis>>, koji ukazuje na to da će označeni slučajevi korišćenja, u kasnijim fazama biti specificirani na specifičan način kako bi se omogućila realizacija interoperabilnosti. Rezultati primene prvog koraka predloženog metodološkog pristupa za specifikaciju aspekata interoperabilnosti su sitematizovni u Tabeli 6.1.

Tabela 6.1 Identifikacija zahteva za interoperbilnošću

SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI **FAZA:** Identifikacija **KORAK:** Identifikacija zahteva za interoperabilnošću	
Metodologija	FON Labis metodologija za projektovanje IS
Tehnika	Dijagrami tokova podataka (DTP)
Preduslov	Idenitifikacija modela esencijalnih poslovnih funkcija
Proširenje postojeće tehnike	Primitivne funkcije koje treba da podrže interoperabilnost i interfejse koji predstavljaju kolaborativne partnere treba označiti drugom bojom (npr. plavom).
Uvođenje nove tehnike	Model interoperabilnih slučajeva korišćenja

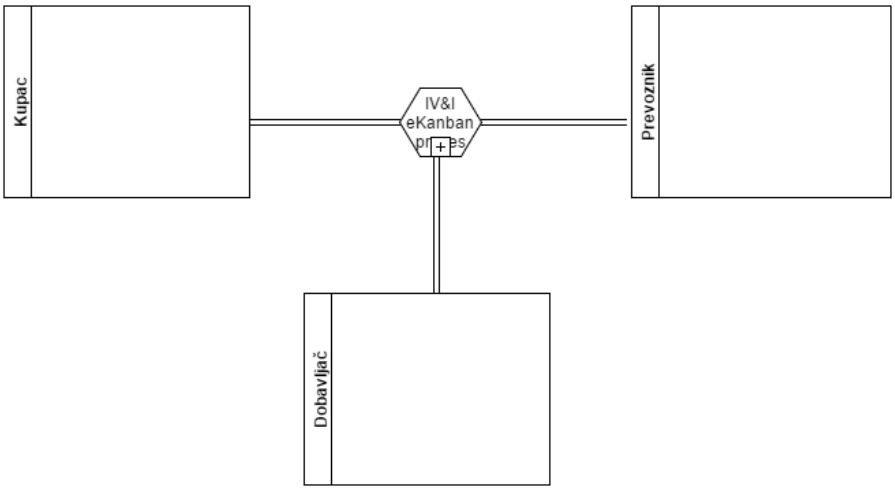
Rezultat	-Identifikacija poslovnih funkcija koje zahtevaju interoperabilnost i kolaborativnih poslovnih partnera. -Kreiran je Model interoperabilnih slučajeva korišćenja.
Prednosti	-Polazni model esencijalnih poslovnih funkcija je lako identifikovati na osnovu primitivnih funkcija SSA. -Laka identifikacija kolaborativnih poslovnih partnera i interoperabilnih primitivnih poslovnih procesa na osnovu njihove komunikacije sa interfejsima.

6.2.1.2. Definisanje kolaboracija

U drugom koraku faze identifikacije aspekata interoperabilnosti, primarni fokus je na definisanju kolaboracija između poslovnih partnera koje se realizuju međusobnom razmenom poruka. S obzirom na to da su poslovni partneri identifikovani u prethodnom koraku, postavlja se pitanje kako najlakše identifikovati poruke koje se razmenjuju u kolaboraciji.

6.2.1.2.1. Specifikacija dijagrama konverzacije

Za početnu analizu i postupnu specifikaciju kolaborativnih poslovnih procesa se preporučuje izrada dijagrama konverzacije, kojim se identifikuju partneri koji učestvuju u kolaboraciji i logički povezane grupe poruka koje se između njih razmenjuju. Na slici 6.12. je prikazan dijagram složene konverzacije između osnovnih učesnika IV&I eKanban poslovnog procesa. Za identifikaciju partnera je pogodan *Model interoperabilnih slučajeva korišćenja* koji je definisan u prethodnom koraku.

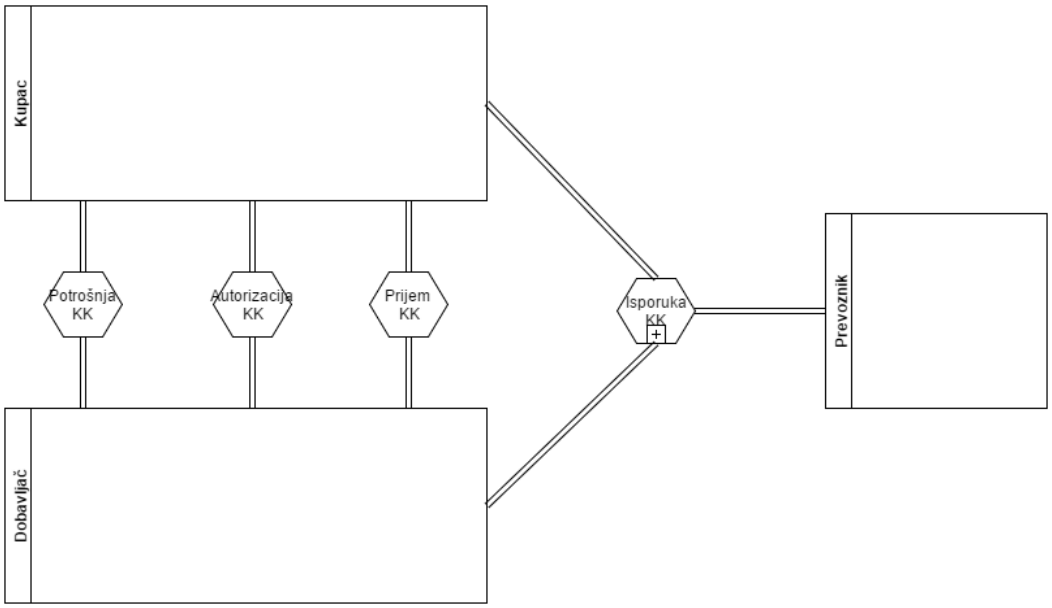


Slika 6.12 Dijagram konverzacije za IV&I eKanban CBP

Na slici 6.13 je prikazan dijagram konverzacije za kolaborativni poslovni proces *Izvršenje eKanban procesa*. Na dijagramu je prikazana složena konverzacija *Isporuka Kanban kontejnera*, kao i tri standardne konverzacije: *Potrošnja Kanban kontejnera*, *Autorizacija Kanban kontejnera* i *Prijem Kanban kontejnera*. Napomenućemo da je prilikom definisanja konverzacija pored funkcionalne dekompozicije sistema primenom metode SSA, bio od koristi i *Model interoperabilnih slučajeva korišćenja*. Analogija koja je identifikovana između slučajeva korišćenja i relevantnih konverzacija je prikazana u Tabeli 6.2.

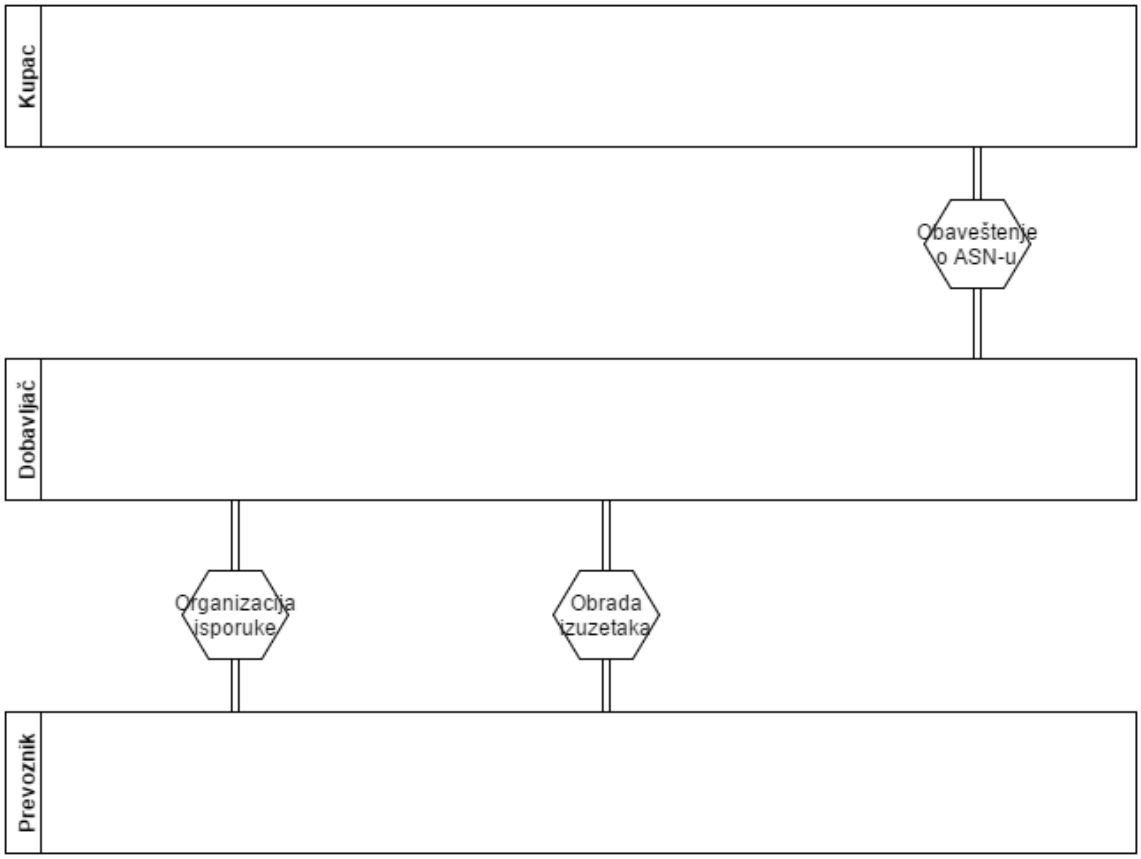
Tabela 6.2 Veza između Modela slučajeva korišćenja interoperabilnih poslovnih funkcija i konverzacija

Slučajevi korišćenja <<servis>>	***Konverzacije***
Evidentiranje potrošenog Kanban kontejnera	Potrošnja Kanban kontejnera
Evidentiranje autorizacije Kanban kontejnera	Autorizacija Kanban kontejnera
Evidentiranje potvrde o prijemu Kanban kontejnera	Prijem Kanban kontejnera
Generisanje ASN Obrada izuzetaka Generisanja zahteva za isporuku	Isporuka Kanban kontejnera



Slika 6.13 Dijagram konverzacije za Izvršenje eKanban procesa

Dekompozicija složene konverzacije *Isporuka Kanban kontejnera* na tri standardne konverzacije: *Organizacija isporuke*, *Obrada izuzetaka* i *Obaveštenje o ASN* je prikazana na slici 6.14.

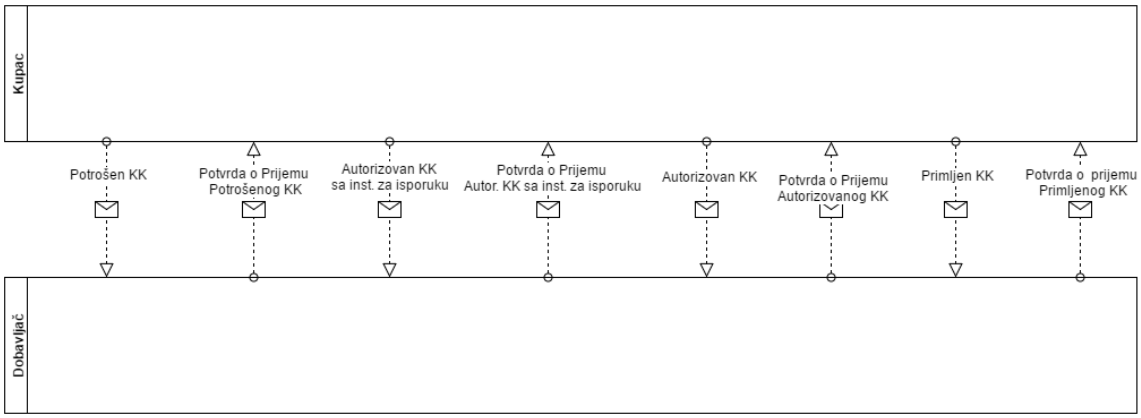


Slika 6.14 Dijagram konverzacije za Isporuku Kanban kontejnera

6.2.1.2.2. Specifikacija dijagrama kolaboracije

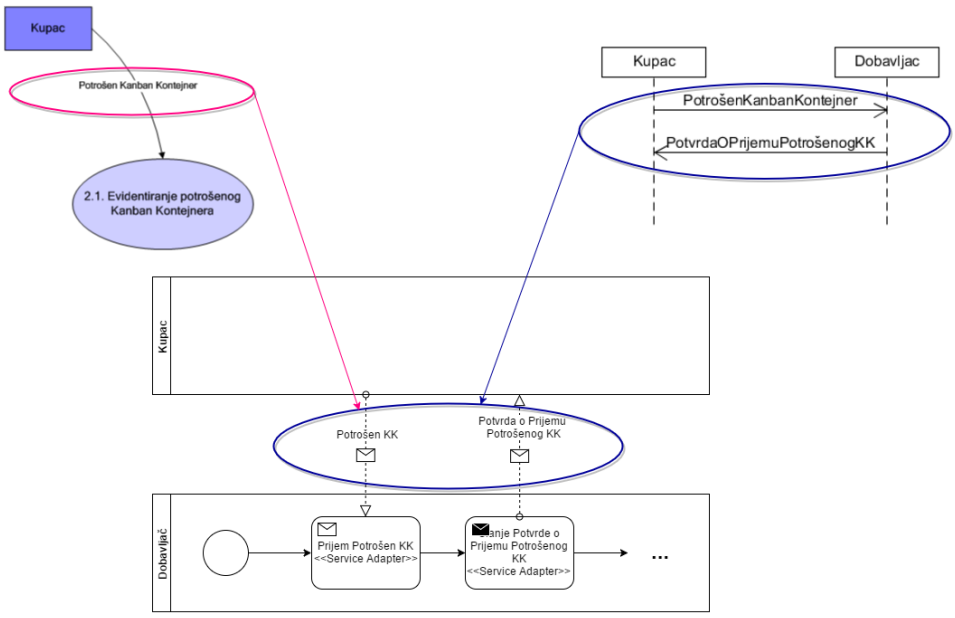
Za svaki dijagram konverzacije koji je kreiran u prethodnom koraku, potrebno je kreirati odgovarajući dijagram kolaboracije. Osnovna razlika dijagrama kolaboracije u odnosu na dijagram konverzacije je detaljan prikaz toka poruka koje se razmenjuju između učesnika. Na dijagramu konverzacije poruke su logički grupisane, što u velikoj meri olakšava njihovu identifikaciju.

Kreirani dijagrami konverzacije su poslužili kao osnova za identifikaciju dva osnovna kolaborativna poslovna procesa: *Realizacija Kanban kontejnera* i *Isporuka Kanban kontejnera*. Dijagram kolaboracije za *Realizaciju Kanban kontejnera* koja podrazumeva njegovu potrošnju, autorizaciju i prijem je prikazan na slici 6.15.



Slika 6.15 Kolaborativni poslovni proces Realizacija Kanban kontejnera

Primena metode strukturne sistemske analize za funkcionalnu specifikaciju sistema, je dodatno olakšala postupak identifikacije poruka. Identifikacija poruka je izvršena na osnovu ulazno-izlaznih tokova koji komuniciraju sa spoljnim interfejsima sistema. Pri analizi tokova podataka su uzeti u obzir samo tokovi ka onim interfejsima koji su u prvom koraku označeni kao kolaborativni partneri. Na slici 6.16 je ilustrovana veza između odgovarajućih koncepata dijagrama tokova podataka SSA i dijagrama kolaboracije, na primer toka podataka *Potrošen Kanban kontejner* i odgovarajuće poruke sa istim nazivom.



Slika 6.16 Postupak identifikacije kolaborativnog procesa na osnovu SSA

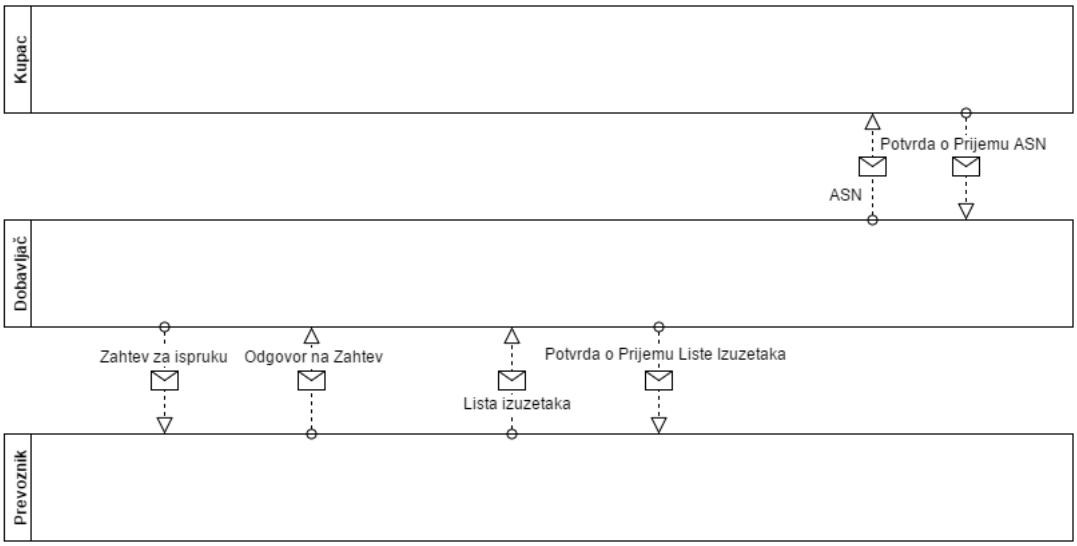
Iako dijagrami tokova podataka SSA olakšavaju postupak identifikacije poruka, preslikavanje u odgovarajuće poruke na dijagramu kolaboracije ne mora da bude

jednoznačno. U skladu sa preporukama koje su date u (Denis Gagné & Conrad Bock, 2014), za modelovanje kolaborativnog poslovnog procesa je primenjen opšti interakcioni patern za OAGIS poslovne scenarije. U pitanju je *Acknowledge – Respond* patern koji podrazumeva da jedan učesnik šalje zahtev (eng. *request*) drugom učesniku kolaboracije, koji po prijemu zahteva mora da pošalje odgovor (eng. *respond*) i na taj način potvrdi da je primio zahtev, i uz to naglasi da li će zahtev biti ispunjen ili ne. Ostale poruke kolaborativnog poslovnog procesa *Realizacija Kanban kontejnera*, koje su identifikovane po istom principu, su prikazane u Tabeli 6.3.

Tabela 6.3 Identifikacija poruka na osnovu konverzacija – kolaborativni proces Realizacija Kanban Kontejnera

Konverzacije	***Identifikovane poruke***
Potrošnja Kanban Kontejnera	➔Potrošen Kanban Kontejner 🡐Potvrda o Prijemu Potrošenog Kanban Kontejnera
Autorizacija Kanban Kontejnera	➔Autorizovan Kanban Kontejner sa instrukcijama za isporuku 🡐Potvrda o Prijemu Autorizovanog Kanban Kontejnera sa instrukcijama za isporuku ➔Autorizovan Kanban Kontejner 🡐Potvrda o Prijemu Autorizovanog Kanban Kontejnera
Prijem Kanban Kontejnera	➔Primljen Kanban Kontejner 🡐Potvrda o prijemu Primljenog Kanban Kontejnera

Dijagram kolaboracije za *Isporuku Kanban Kontejnera* je prikazan na slici 6.17.



Slika 6.17 Dijagram kolaboracije za isporuku Kanban kontejnera

Rezultati primene drugog koraka predloženog pristupa za specifikaciju aspekata interoperabilnosti, a koji se odnosi na postupak i preporuke za definisanje kolaboracija između poslovnih partnera su sistematizovani u Tabeli 6.4.

Tabela 6.4 Specifikacija aspekata interoperabilnosti-Definisaje kolaboracija

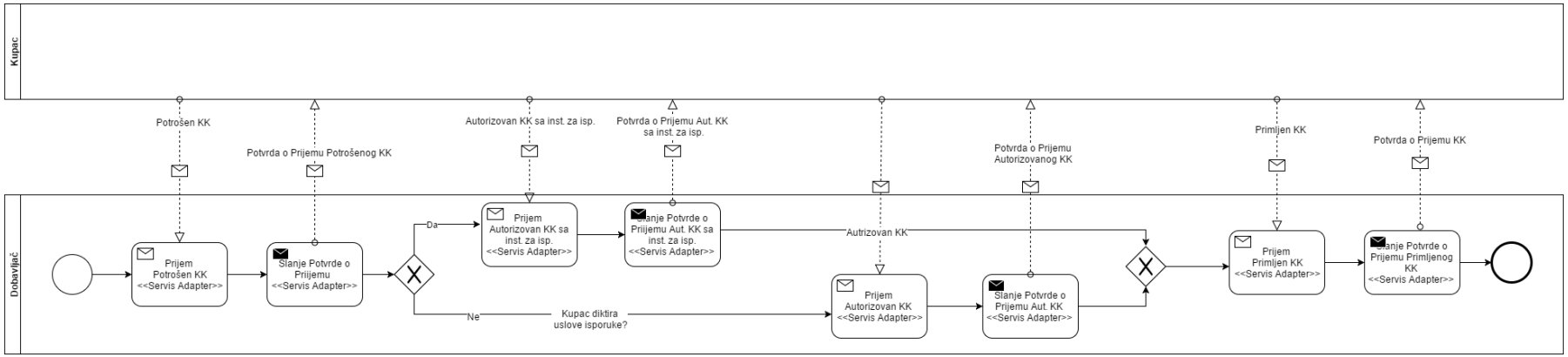
SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI **FAZA:** Identifikacija **KORAK:** *Definisanje kolaboracija*	
Metodologija	FON Labis metodologija za projektovanje IS
Tehnika	BPMN dijagram konverzacije i kolaboracije
Preduslov	Kreiranje Modela slučajeva interoperabilnih poslovnih funkcija
Proširenje postojeće tehnike	-
Uvođenje nove tehnike	-
Rezultat	Izrada dijagrama konverzacije Izrada dijagrama kolaboracije
Prednosti	Laka identifikacija poruka između poslovnih partnera na osnovu ulaznih i izlaznih tokova ka interfejsima

6.2.1.3. Specifikacija privatnog i javnog kolaborativnog procesa

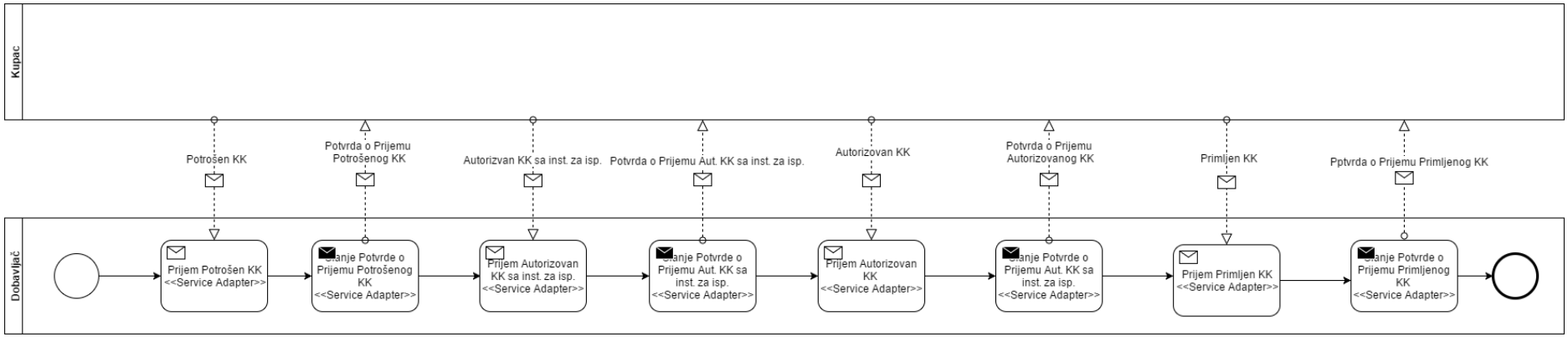
Nakon specifikacije dijagrama kolaboracije, u ovom koraku je pažnja usmerena na opis načina (dinamike) odvijanja kolaborativnog poslovnog procesa. Dijagram kolaboracije treba da opiše način povezivanja primitivnih kolaborativnih poslovnih procesa koji su identifikovani u prvom koraku predložene metodologije. S obzirom na to da je na *Dijagramu interoperabilnih slučajeva korišćenja* prikazana veza između poslovnih partnera i slučajeva korišćenja (tj. kolaborativnih primitivnih procesa), u ovom koraku preostaje da se precizira njihova orkestracija.

U skladu se preporukama predloženog metodološkog pristupa, na osnovu dijagrama kolaboracije se prvo kreira privatni poslovni proces. Na slici 6.18 je prikazan privatni poslovni proces *Realizacija Kanban Kontejnera*, koji je kreiran na osnovu odgovarajućeg kolaborativnog poslovnog procesa koji je prikazan na slici 6.15. Privatni kolaborativni poslovni proces *Realizacija Kanban Kontejnera* se analizira sa aspekta Dobavljača, pa je iz tog razloga Kupac prikazan koristeći princip „crne kutije“. Za svaku poruku sa dijagrama na slici 6.15 je definisana odgovarajuća

aktivnost. Na primer, za poruku *Potrošen Kanban Kontejner* koju Kupac šalje Dobavljaču definisana je aktivnost *Prijem Potrošenog Kanban Kontejnera*. Budući da nije bilo potrebe za uvođenjem dodatnih aktivnosti, privatni proces *Realizacija Kanban Kontejnera* sadrži samo aktivnosti za slanje i prijem poruka, pa njegova javna reprezentacija poseduje identičan skup aktivnosti (slika 6.19). Ipak, privatni proces ima dodatnu semantiku u odnosu na svoju javnu reprezentaciju, što je ilustrovano definisanjem uslova kojim se ispituje da li Kupac diktira uslove isporuke. Ukoliko Kupac diktira uslove isporuke, Dobavljač će primiti poruku o autorizaciji Kanban kontejnera zajedno sa instrukcijama za isporuku. U suprotom, Dobavljač sam organizuje isporuku Kanban kontejnera, tako da mu je od Kupca dovoljan samo signal za autorizaciju Kanbana.

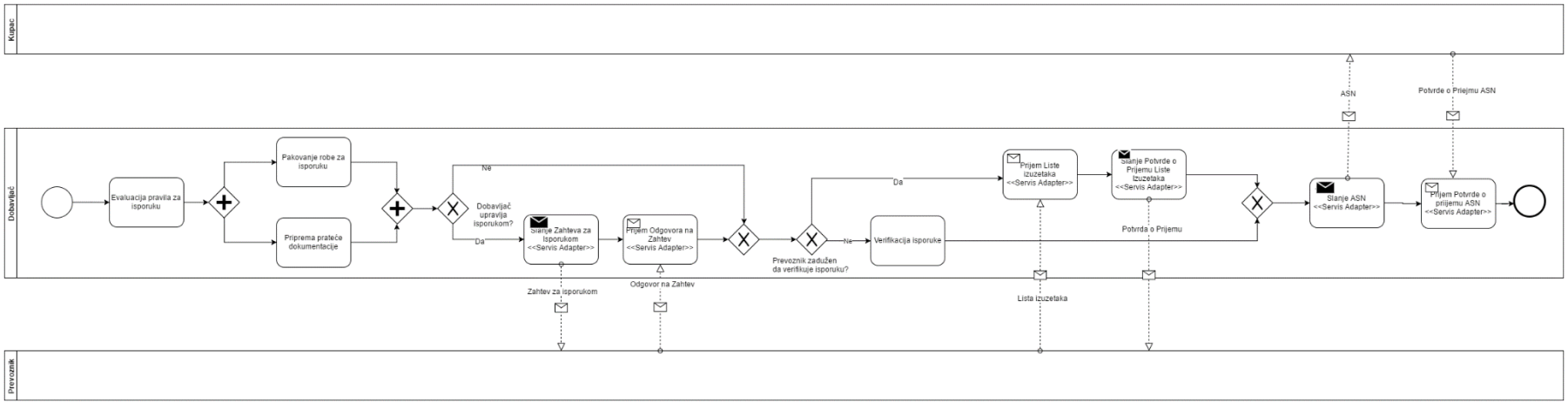


Slika 6.18 Privatni proces Realizacija Kanban kontejnera

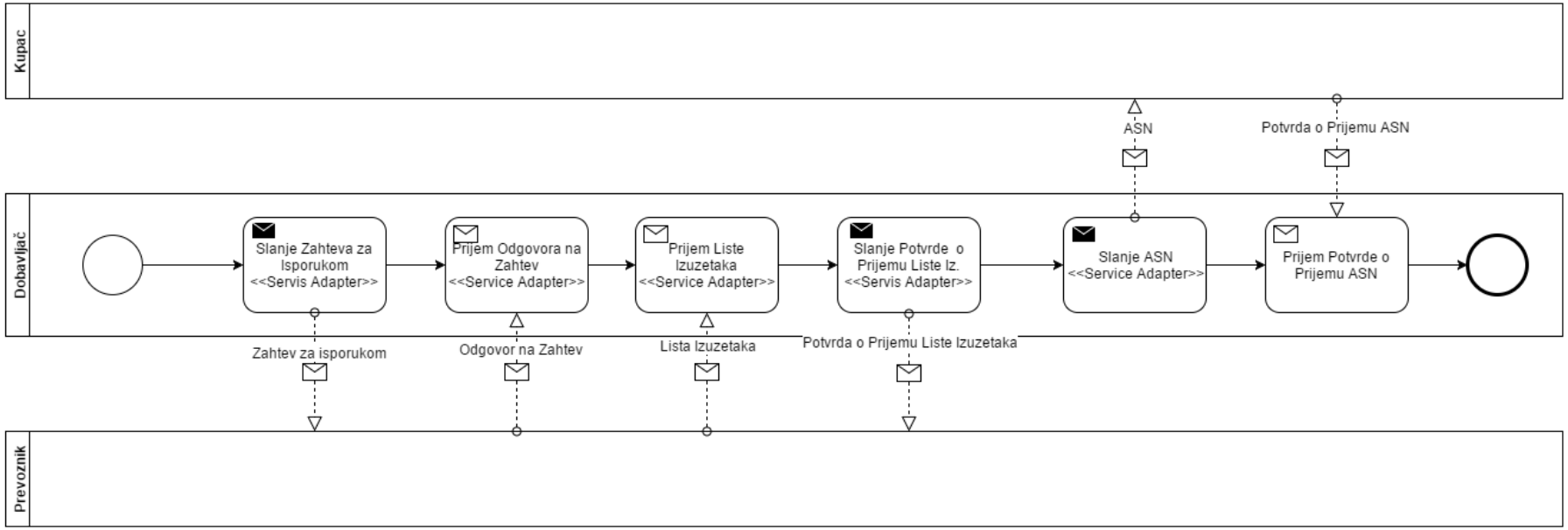


Slika 6.19 Javni proces Realizacija Kanban kontejnera

Na slici 6.20. je prikazan privatni poslovni proces *Isporuka Kanban Kontejnera*. Poslovni proces počinje aktivnošću *Evaluacija pravila za isporuku*, gde Dobavljač vrši internu kontrolu onih Kanban kontejnera čiji je status „autorizovan“. Za Kanban kontejnere za koje se utvrdi da ispunjavaju uslove isporuke, Dobavljač paralelno priprema prateću dokumentaciju i pakuje robu. Ove dve aktivnosti su na dijagramu kolaboracije prikazane pomoću paralelnog kontrolnog čvora (eng. parallel gateway) koji sledi posle aktivnosti *Evaluacija pravila za isporuku*. Aktivnost *Verifikacija isporuke* će biti izvršena ako je Dobavljač zadužen da proveri robu koju treba isporučiti pre slanja. Za sve dosada navedene aktivnosti je zajedničko da njihovo izvršenje ne zahteva eksplicitnu komunikaciju sa ostalim učesnicima kolaboracije. Naime, ove interne aktivnosti imaju privatni karakter i njihova realizacija zavisi samo od Dobavljača. Ostale aktivnosti kolaborativnog poslovnog procesa služe za slanje ili prijem poruka. Na slici 6.21 je prikazan javni poslovni proces koji je izveden iz privatnog procesa, apstrakcijom sledećih privatnih aktivnosti: *Evaluacija pravila za isporuku*, *Pakovanje robe za isporuku*, *Priprema prateće dokumentacije* i *Verifikacija isporuke*. Kao što se može videti na slici 6.21, javni poslovni proces *Isporuka Kanban Kontejnera* sadrži samo aktivnosti za slanje, odnosno prijem poslovnih poruka.



Slika 6.20 Privatni proces Isporuka Kanban kontejnera



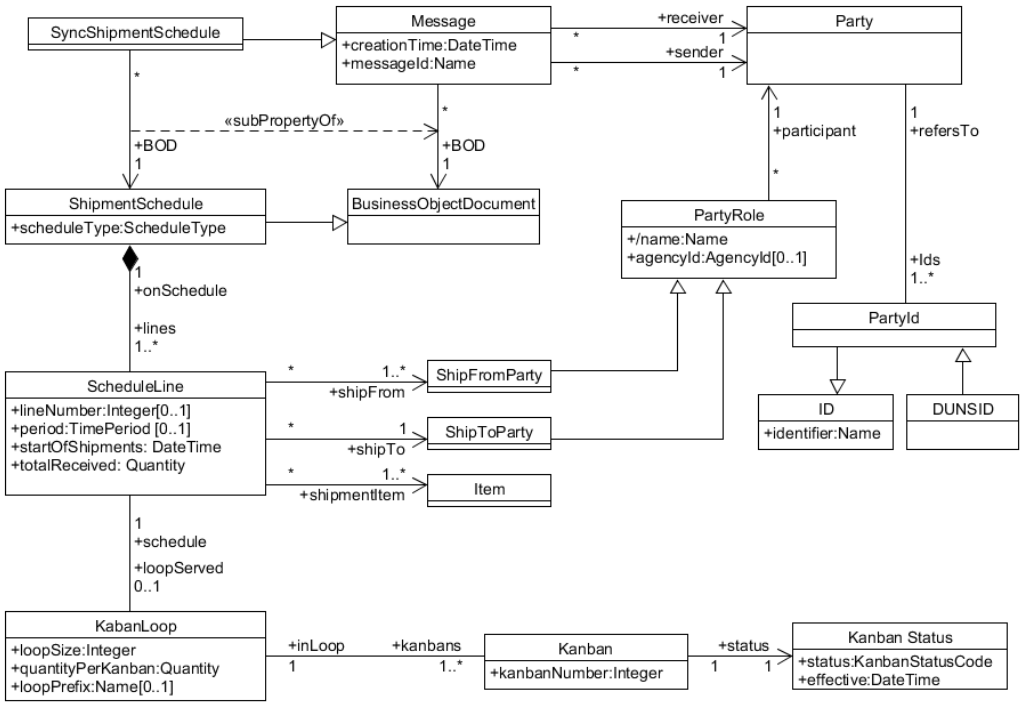
Slika 6.21 Javni proces isporuka Kanban kontejnera

Tabela 6.5 Specifikacija aspekata interoperabilnosti–privatni i javni proces

SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI **FAZA:** Identifikacija **KORAK:** Specifikacija privatnog i javnog kolaborativnog procesa	
Metodologija	FON Labis metodologija za projektovanje IS
Preduslov	Izrada dijagrama kolaboracije
Proširenje postojeće tehnike	-
Uvođenje nove tehnike	-
Rezultat	Kreiranje privatne reprezentacije kolaborativnog procesa Apstrakcija javne na osnovu privatne reprezentacije kolaborativnog pocesa
Prednosti	Laka identifikacija interaktivnih poslovnih aktivnosti na osnovu Modela interoperabilnih slučajeva korišćenja Olakšana identifikacija poslovnih aktivnosti na osnovu hijerarhijske dekompozicije dijagrama tokova podataka SSA

6.2.1.4. Definisanje globalne Referentne Ontologije

EKanban referentna ontologija koja je razvijena na National Institute of Standards and Thechnology (NIST-u) u saradnji sa timom IV&I industrijskih analitičara detaljno je opisana u (E. Barkmeyer & Kulvatunyou, 2007). EKanban referentna ontologija formalno reprezentuje poslovne koncepte i veze između koncepata koji su bitni za razumevanje eKanban poslovnog procesa . Na slici 6.22 je prikazan jedan deo eKanban referentne ontologije koji ilistruje samo neke od osnovnih koncepata.



Slika 6.22 Deo eKanban referentne ontologije (E. Barkmeyer & Kulvatunyou, 2007)

SyncShipmentSchedule je poruka koju šalje pošiljalac (eng. sender) (u ovom slučaju Kupac) primaocu (eng. receiver) (u ovom slučaju Dobavljač) da ažurira *ShipmentShedule*. *ShipmentSchedule* predstavlja BOD koji definiše detalje isporuke Kanban kontejnera koji su definisani ugovorom između Kupca i Dobavljača. Svaka *ScheduleLine* stavka *ShipmentSchedule* BOD-a odgovara jednom *KanbanLoop*-u, koji definiše broj Kanban kontejnera u petlji i količinu po Kanban kontejneru. Za isporuku svakog Artikla (eng. item) od ShipFromParty do ShipToParty, se definiše posebna Kanban petlja (Kanban Loop). Osnovna jedinica isporuke je Kanban. Osnovni cilj slanja SyncShipmentSchedule poruke je autorizacija jednog ili više Kanbana, i postavljanje njihovog statusa na „Autorizovan“.

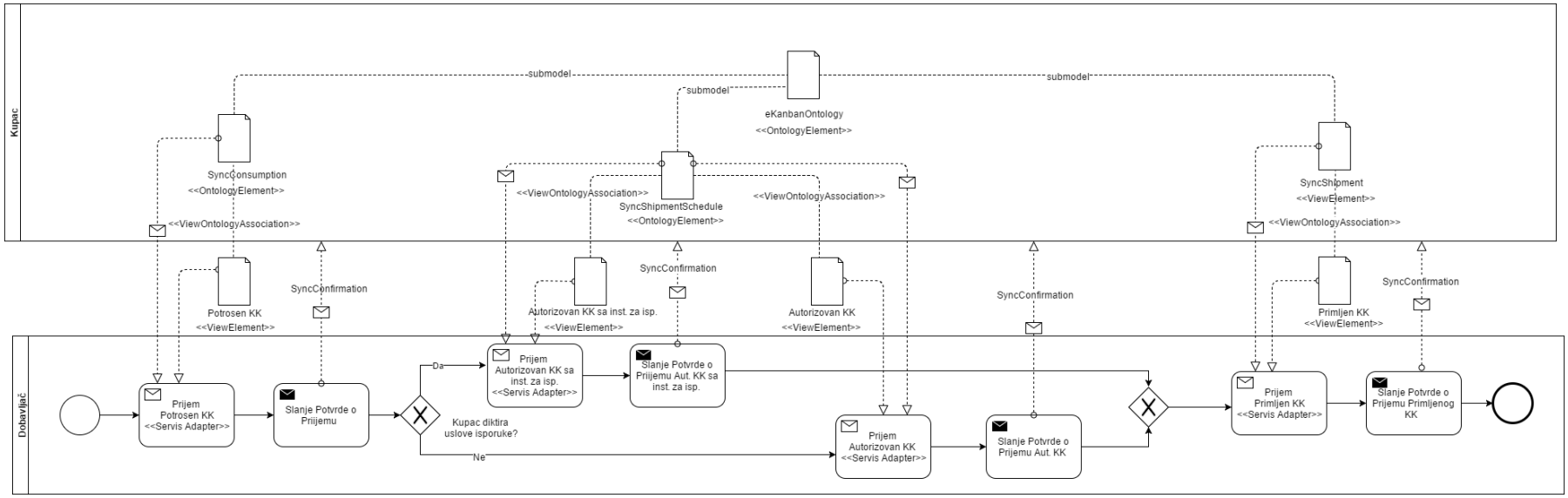
6.2.2. Realizacija aspekata interoperabilnosti

6.2.2.1. Specifikacija poruka privatnog kolaborativnog procesa kao pogleda nad globalnom referentnom ontologijom

Na slici 6.23 je prikazana veza između privatne reprezentacije procesa *Realizacija Kanban Kontejnera* i *eKanban* globalne referentne ontologije. *EKanban* globalna

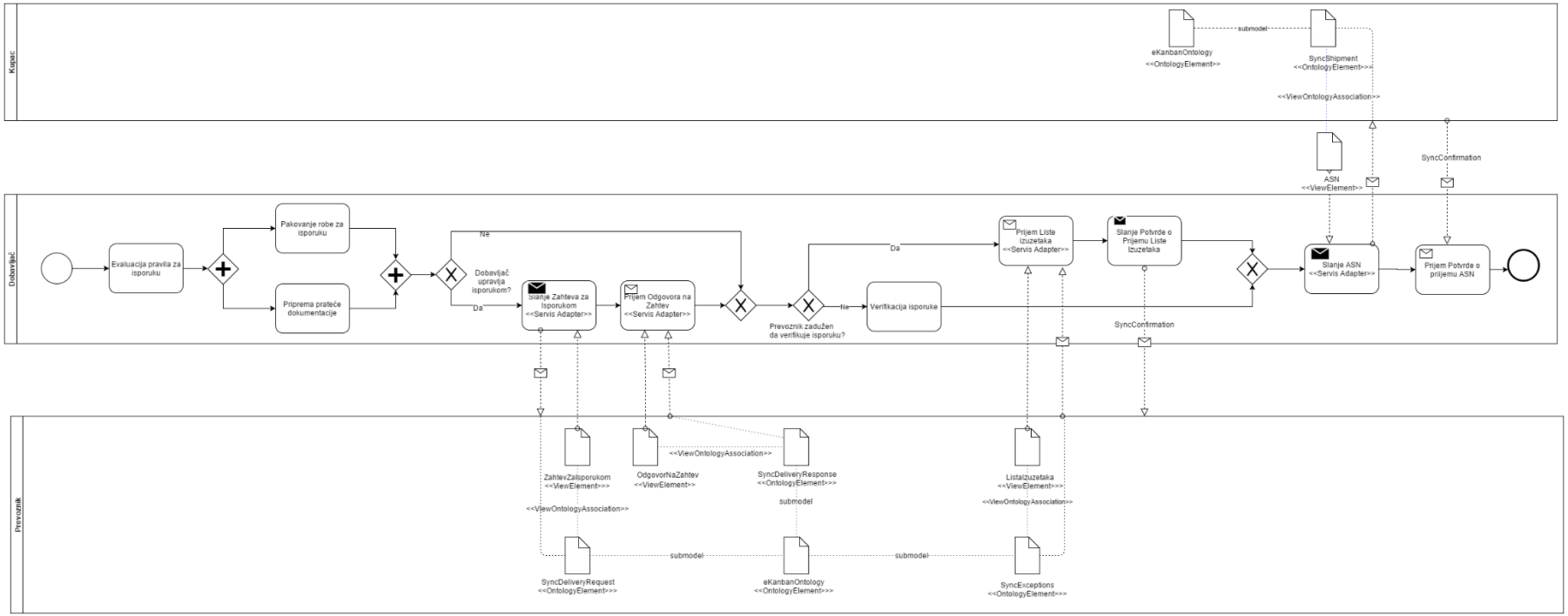
referentna ontologija i poruke koje su definisane kao njeni podmodeli su predstavljeni pomoću *Data Object* elemenata na koje je primenjen stereotip <<OntologyElement>> koji je definisan ekstenzijom BPMN-a. Na dijagramu 6.23 se može videti da se u modelu koristi eKanbanOntology referentna ontologija, na osnovu koje su kreirana tri podmodela: *SyncConsumption*, *SyncShipmentShedule* i *SyncSchipment* koji odgovaraju standardnim IV&I porukama. Kupac šalje Dobavljaču *SyncConsumption* poruku, čija je struktura definisana koristeći koncepte eKanban referentne ontologije i koja sadrži širi skup svojstava budući da predstavlja standardizovanu poruku koja može da se koristi u različitim eKanban poslovnim scenarijima. Za potrebe definisanja onog podskupa svojstava koji je neophodan za izvršenje aktivnosti *Prijem potrošenog Kanban Kontejnera*, kreiran je pogled *Potrošen Kanban Kontejner* nad *SyncConsumption* modelom referentne ontologije. Pogled *Potrošen Kanban Kontejner* je vizuelno prikazan na dijagramu kao *Data Object* element na koji je primenjen <<ViewElement>> stereotip i doveden je u vezu sa relevantom aktivnošću pomoću asocijacije <<ViewOntologyAssociaton>>, ali nije data definicija njegove strukture. Struktura pogleda koji su definisani nad referentnom ontologijom se definiše na osnovu *UML View Profila* i biće objašnjena u narednom koraku.

Poruka *Potvrda o Prijemu Potrošenog Kanban Kontejnera* koja je definisana u skladu sa OAGIS interakcionim Acknowledge-Respond paternom ima jednostavnu strukturu za potvrdu prijema *SyncConsumption* poruke, tako da nije bilo potrebe da se eksplicitno dovede u vezu sa eKanban referentnom ontologijom. Na osnovu jednog podmodela referentne ontologije može da se izvede više modela pogleda, na primer pogledi *Autorizovan KK sa instrukcijama za isporuku* i *Autorizovan KK* su izvedeni na osnovu *SyncShipmentSchedule* <<Ontology Element>>.



Slika 6.23 Veza privatnog procesa Realizacija Kanban kontejnera i referentne ontologije

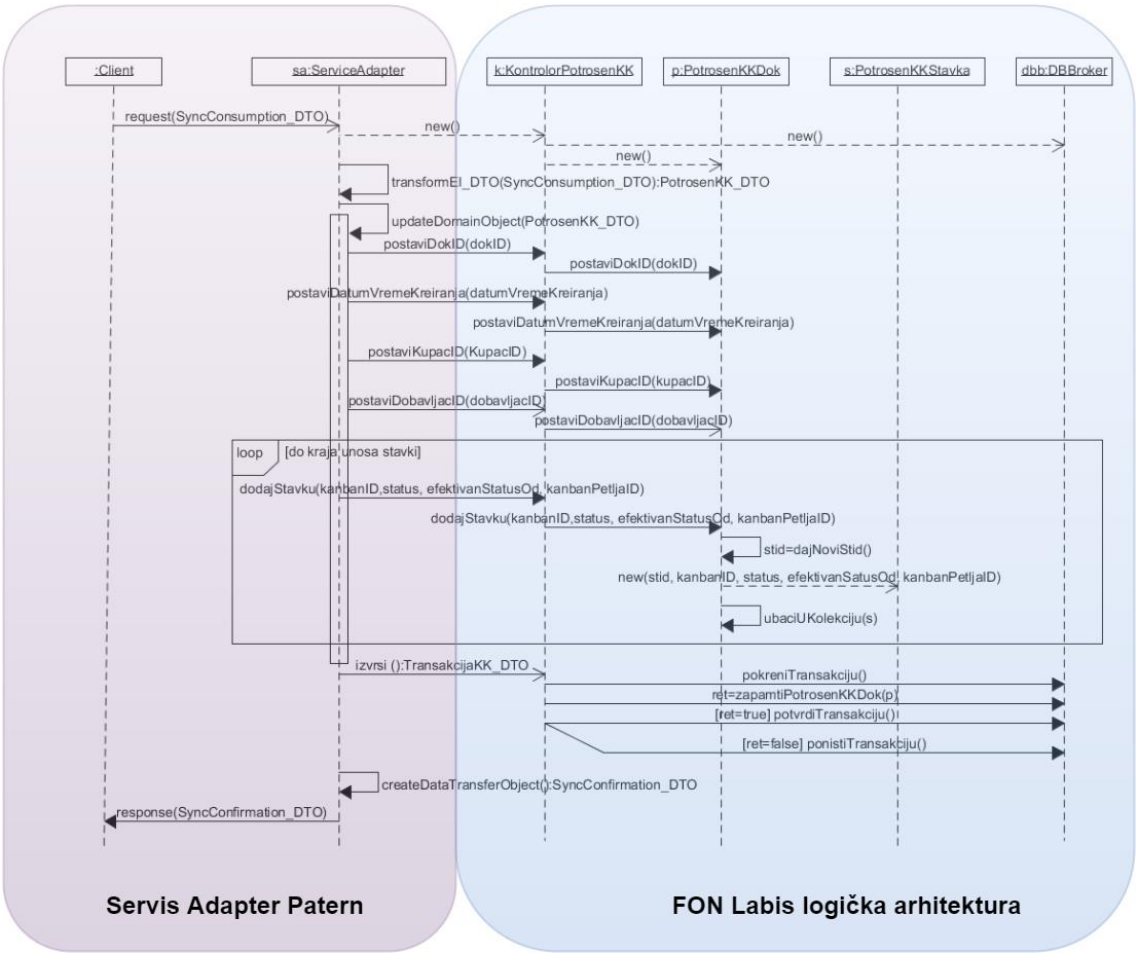
Na slici 6.24 je prikazana veza između privatnog procesa I*sporuka Kanban Kontejnera* i eKanban referentne ontologije. Na osnovu eKanban referentne ontologije su definisana tri podmodela, za tri odgovarajuće poruke: *SyncShipment*, *SyncDeliveryRequest* i *SyncExceptions*. Model pogleda *ZahtevZaIsporukom* koji reprezentuje potreban ulaz za realizaciju aktivnosti *Slanje Zahteva Za Isporukom* je predstavljen kao <<ViewElement>>. Njegova veza sa *SyncDeliveryRequest* <<OntologyElement>> elementom je iskazana pomoću asocijacije <<ViewOntologyAssociation>>. Slično, pogled *Lista Izuzetaka* je doveden u vezu sa *SyncExceptions* <<OntologyElement>> elementom, dok je pogled *ASN* nastao na osnovu *SyncShipment* <<OntologyElement>> elementa.



Slika 6.24 Veza privatnog procesa Isporuka Kanban kontejnera i referentne ontologije

6.2.2.2. Specifikacija Servis Adaptera

Na slici 6.25 je prikazan logički dijagram sekvenci za specifikaciju servisa kolaborativnog poslovnog procesa primenom *Servis Adapter* paterna na primeru prijema *SyncConsumption* poruke. Postupak kreiranja dijagrama sekvenci koji je prikazan na slici 6.25 je detaljno opisan u poglavlju (5.3.2). Osnovna uloga *Servis Adapter* paterna je da omogući laku manipulaciju podacima iz *request* objekta koji šalje klijent, i da ih prosledi do *Kontrolora* u formi koja je ne zahteva izmene postojeće poslovne logike aplikacije. Za komunikaciju *Servis Adaptera* sa *Klijentom* i standardnim *Kontrolorom* FON Labis logičke arhitekture aplikacija, u radu se predlaže primena DTO paterna kao što je prikazano na dijagramu 6.25.



Slika 6.25 Veza Servis Adapter paterna i FON Labis logičke arhitekture aplikacija

S obzirom na to da su funkcionalnosti i metode *Servis Adaptera* detaljno opisane u poglavlju (5.3.2), u ovom poglavlju ćemo ukazati na vezu između detaljne

specifikacije poruka privatnog kolaborativnog poslovnog procesa i *Servis Adapter* paterna. Na dijagramu privatnog kolaborativnog poslovnog procesa *Realizacija Kanban Kontejnera* (slika 6.23), standardna poruka *SyncConsumption* je označena stereotipom *<<OntologyElement>>*. Ova poruka predstavlja podmodel (*eng. submodel*) eKanban referentne ontologije, odnosno njena struktura je opisana pomoću koncepata *eKanban* ontologije. Na dijagramu sekvenci na slici 6.25 *SyncConsumption* poruka je predstavljena kao *request* poruka koju Klijent (aplikacija na strani Kupca) šalje aplikaciji na strani Dobavljača. Naime, klijent šalje eksterni DTO *SyncConsumption* objekat (*SyncConsumption_DTO*), koji *Server Adapter* transformiše u internu reprezentaciju (*PotrosenKK_DTO*) koristeći *UML View Profil*. U slučaju kada je potrebno poslati odgovor (eng. response) *Klijentu,* uloga *Servis Adaptera* je da primenom reverznog postupka generiše eksterni DTO na osnovu internog DTO objekta, koristeći *UML View Profil*. Ovaj postupak je detaljno objašnjen u narednom poglavlju.

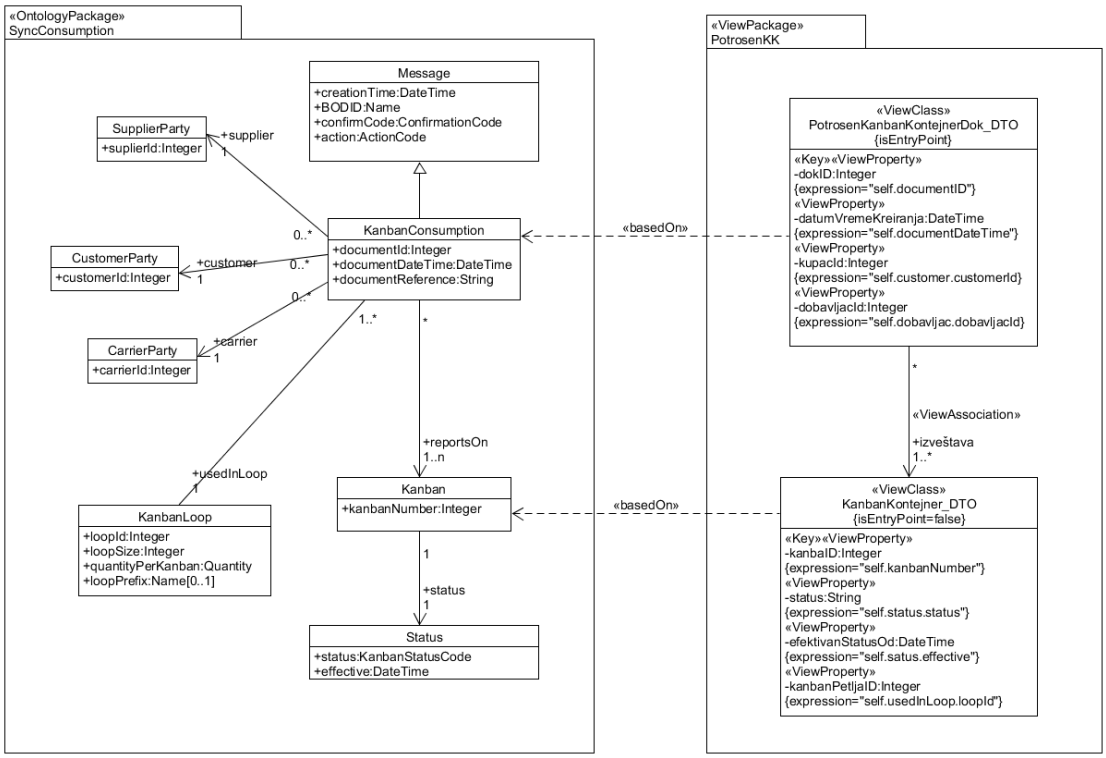
6.2.2.3. Specifikacija strukture poruka preko UML View Profila

Struktura standardne IV&I eKanban *SyncConsumption* poruke, koju Kupac šalje Dobavljaču da ga obavesti o potrošnji Kanban kontejnera, je definisana istoimenim paketom *SyncConsumption <<Ontology Package>>* (slika 6.26). Ova poruka je prikazana kao:

- *SyncConsumption <<OntologyElement>>* na privatnom dijagramu kolaboracije *Realizacija Kanban Kontejnera* (slika 6.23);
- *SyncConsumption_DTO* na logičkom dijagramu sekvenci (slika 6.25).

Radi preglednosti na dijagramu klasa u okviru paketa *SyncConsumption <<OntologyPackage>>* su prikazana samo ona svojstva koja su bitna za strukturu *SyncConsumption* poruke. Drugim rečima, *SyncConsumption <<Ontology Package>>* sadrži podskup klasa eKanban referentne ontologije, koji je relevantan za *SyncConsumption* poruku. Ova semantika je na modelu privatnog kolaborativnog poslovnog procesa *Realizacija Kanban Kontejnera* iskazana pomoću anotacije *submodel* između *SyncConsumption <<OntologyElement>>* i *eKanbanOntology*

<<OntologyElement>> (slika 6.23). Ova struktura ujedno odgovara i strukturi eksternog DTO objekta *SyncConsumtion_DTO* (slika 6.25).



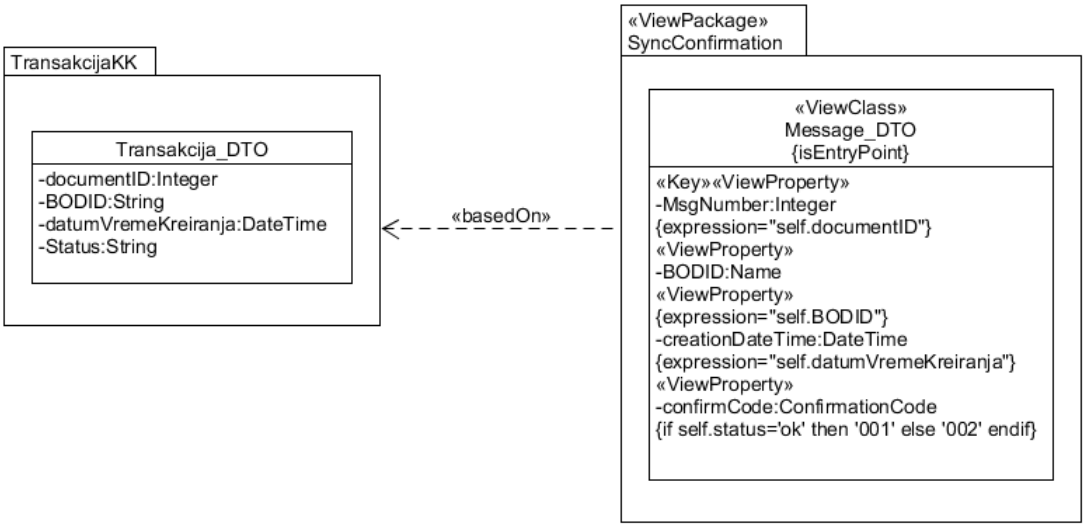
Slika 6.26 Specifikacija strukture PotrošenKK poruke korišćenjem UML View Profila

Poruka *PotrošenKK* se definiše u skladu sa strukturom i terminologijom lokalne referentne ontologije. Njena struktura je definisana paketom *PotrosenKK <<ViewPackage>>* (slika 6.26). Pomoću OCL pravila *UML View Profila* se definišu potrebna preslikavanja ka odgovarajućim elementima globalne referentne ontologije. Na primer, *View Property datumVremeKreiranja* se izvodi pomoću OCL izraza "self.documentDateTime". Ključna reč *self* u okviru OCL izraza se koristi da označi klasu referentne ontologije na koju se vrši mapiranje klase pogleda.

Poruka *PotrosenKK* je na privatnom modelu kolaboracije *Realizacija Kanban Kontejnera* prikazana kao *PotrošenKK <<ViewElement>>*. Asocijacija *<<ViewOntologyAssociation>>* ukazuje na to da se *PotrosenKK <<ViewElement>>* definiše kao pogled nad *SyncConsumption <<OntologyElement>>* primenom pravila *UML View Profila*. Struktura poruke koja je definisana paketom *PotrosenKK*

<<ViewPackage>> odgovara strukturi intetrnog DTO objekta PotrosenKK_DTO (slika 6.25).

UML View Profil predstavlja opšti mehanizam koji može da se primeni za definiciju pogleda nad bilo kojom referentnom ontologijom koja poseduje UML dijagram klasa kao sredstvo za vizuelnu reprezentaciju. Ovo svojstvo *UML View Profila* će biti ilustrovano na primeru kreiranja pogleda nad lokalnom referentnom ontologijom Dobavljača koji učestvuje u eKanban poslovnom procesu.



Slika 6.27 Specifikacija strukture SyncConfirmation poruke korišćenjem UML View Profila

Na slici 6.27 je prikazano da je moguće definisati strukturu *SyncConfirmation* poruke kao pogled nad entitetiom *Transakcija_DTO* lokalne ontologije Dobavljača. Ovo svojstvo *UML View Profila* je bitno za implementaciju funkcionalnosti *Servis Adapter* paterna. Dve osnovne uloge *Servis Adaptera* su: (1) transformacija eksternog u interni DTO objekat pomoću metode *transformEI_DTO()* i (2) kreiranje DTO objekta pomoću metode *createDataTransferObject()*. U prvom slučaju *Servis Adapter* vrši transformaciju eksternog DTO objekta (*SyncConsumption <<Ontology Package>>*) u interni DTO objekat (*PotrosenKK <<View Package>>*), koristeći OCL pravila definisana *UML View Profilom*. U drugom slučaju, zadatak *Servis Adaptera* je da kreira DTO objekat na osnovu primitivnih tipova podataka koje dobija od Kontrolora.

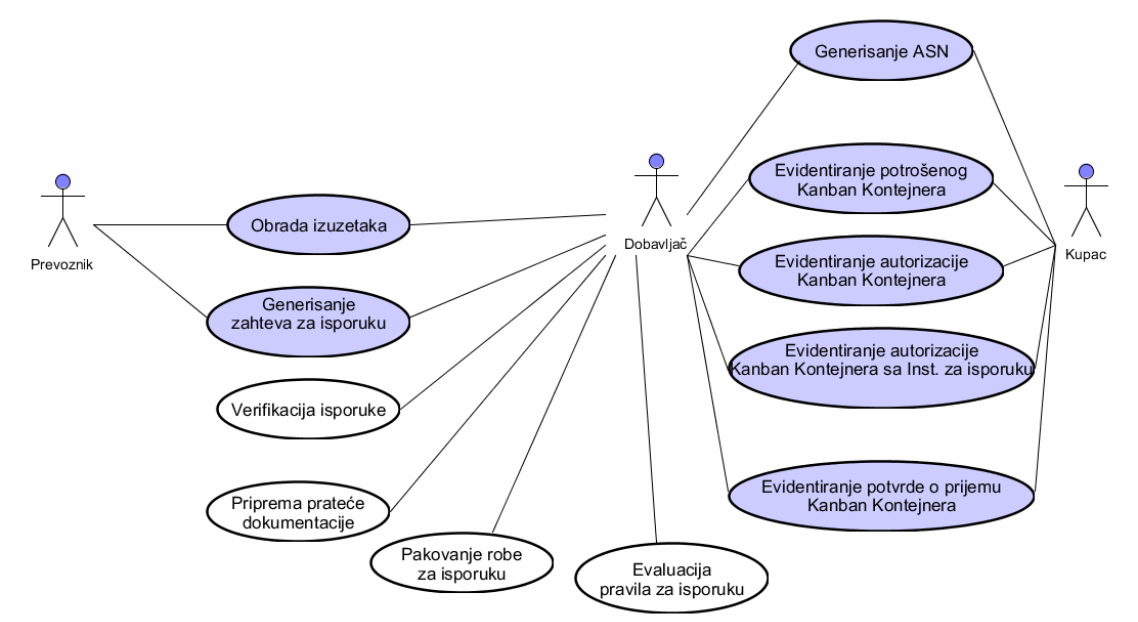
6.3. Larmanova metoda razvoja softvera

U ovom poglavlju će predložena metodologija za specifikaciju aspekata interoperabilnosti biti prikazana primenom Larmanove metode razvoja softvera čije su osnove detaljno opisane u poglavlju tri. Kao eksperimentalni scenario će se koristiti IV&I eKanban poslovni proces.

6.3.1. Identifikacija aspekata interoperabilnosti

6.3.1.1. Identifikacija zahteva za interoperabilnošću

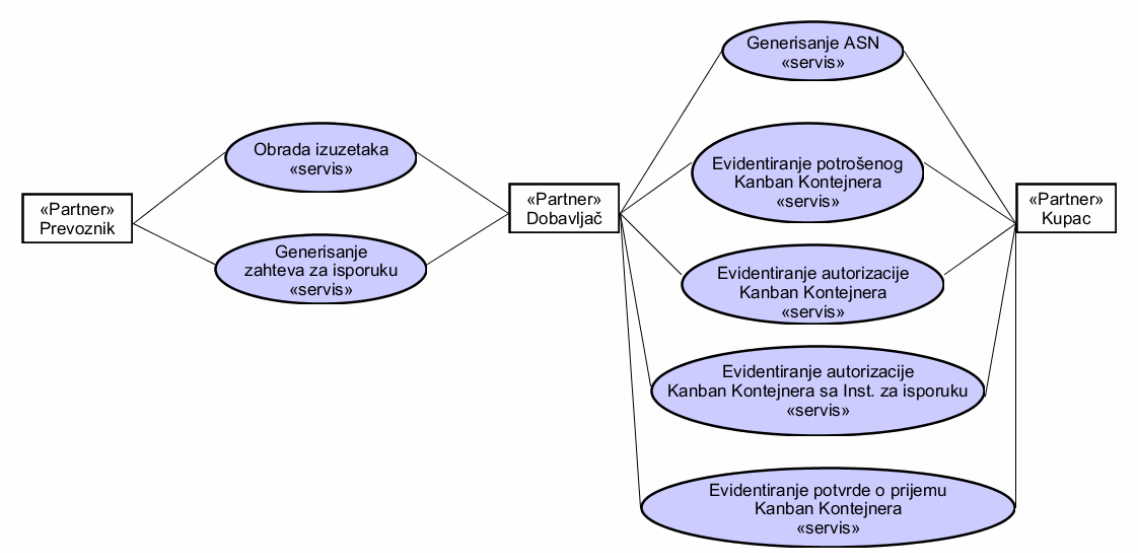
U ovom koraku je potrebno izvršiti identifikaciju zahteva za interoperabilnošću, koji se odnose na identifikaciju esencijalnih poslovnih funkcija koje zahtevaju interoperabilnost i partnera koji učestvuju u kolaboraciji. Za identifikaciju polaznog modela funkcija Larmanova metoda predlaže kreiranje *Dijagrama slučajeva korišćenja*. Na osnovu analize verbalnog opisa eKanban poslovnog procesa, prateće dokumentacije i niza intervjua sa AIAG poslovnim analitičarima definisani su slučajevi korišćenja. Identifikovani slučajevi korišćenja su prikazani na centralnom Dijagramu slučajeva korišćenja koji je prikazan na slici 6.28. Za svaki od identifikovanih slučajeva korišćenja treba da budu detaljno opisani osnovni i relevatni alternativnih scenariji.



Slika 6.28 Dijagram slučajeva korišćenja za IV&I eKanban poslovni scenario

Analizom Dijagrama slučajeva korišćenja su identifikovani kolaborativni slučajevi korišćenja. Model slučajeva korišćenja je olakšao identifikaciju kolaborativnih slučajeva korišćenja, budući da su na modelu eksplicitno prikazane veze između aktora i relevantnih slučajeva korišćenja. Za kolaborativne slučajeve korišćenja je karakteristično da u njihovoj realizaciji učestvuje više aktora, odnosno takav slučaj korišćenja na dijagramu ima veze ka dva ili više aktora. Definisani podskup kolaborativnih slučajeva korišćenja predstavlja kandidate za interoperabilne slučajeve korišćenja. Naime, na modelu slučajeva korišćenja mogu da postoje aktori koji su predstavljeni samo radi lakšeg sagledavanja načina funkcionisanja čitavog poslovnog sistema, pa njihovi slučajevi korišćenja nisu bitni sa aspekta realizacije interoperabilnosti.

U skladu sa predlogom da se za potrebe specifikacije zahteva za interoperabilnošću izvrši minimalna moguća izmena postojećih modela i tehnika, slučajeve korišćenja koji su identifikovani kao zahtevi za interoperabilnošću treba označiti plavom bojom (slika 6.28). U narednom koraku se predlaže da se označeni slučajevi korišćenja izdvojeno prikažu na posebnom *Modelu interoperabilnih slučajeva korišćenja* koji je prikazan na slici 6.29. Identifikovani partneri (Dobavljač, Kupac i Prevoznik) su prikazani odgovarajućim stereotipom <<partner>>, kao što je prikazano na slici 6.29.



Slika 6.29 Model interoperabilnih slučajeva korišćenja dobijen primenom Larmanove metode

Bitno je naglasiti da je dobijeni *Model interoperabilnih slučajeva korišćenja* nije u potpunosti identičan modelu prikazanom na slici 6.11, koji je dobijen kao rezultat primene Strukturne sistemske analize za funkcionalnu specifikaciju eKanban poslovnog sistema. Na primer, esencijani slučaj korišćenja *Evidentiranje autorizacije Kanban kontejnera sa instrukcijama za isporuku* je na DTP-u SSA predstavljen kao tok podataka. Osnovna prednost primene metode Strukturne sistemske analize u ovom koraku, se odnosi na upotrebu principa hijerarhijske dekompozicije sistema. Naime, postepenim savladavanjem složenosti sistema je moguća lakša identifikacija esencijalnih poslovnih funkcija. Rezultati primene prvog koraka pristupa koji se predlaže u tezi su dati u tabeli 6.6.

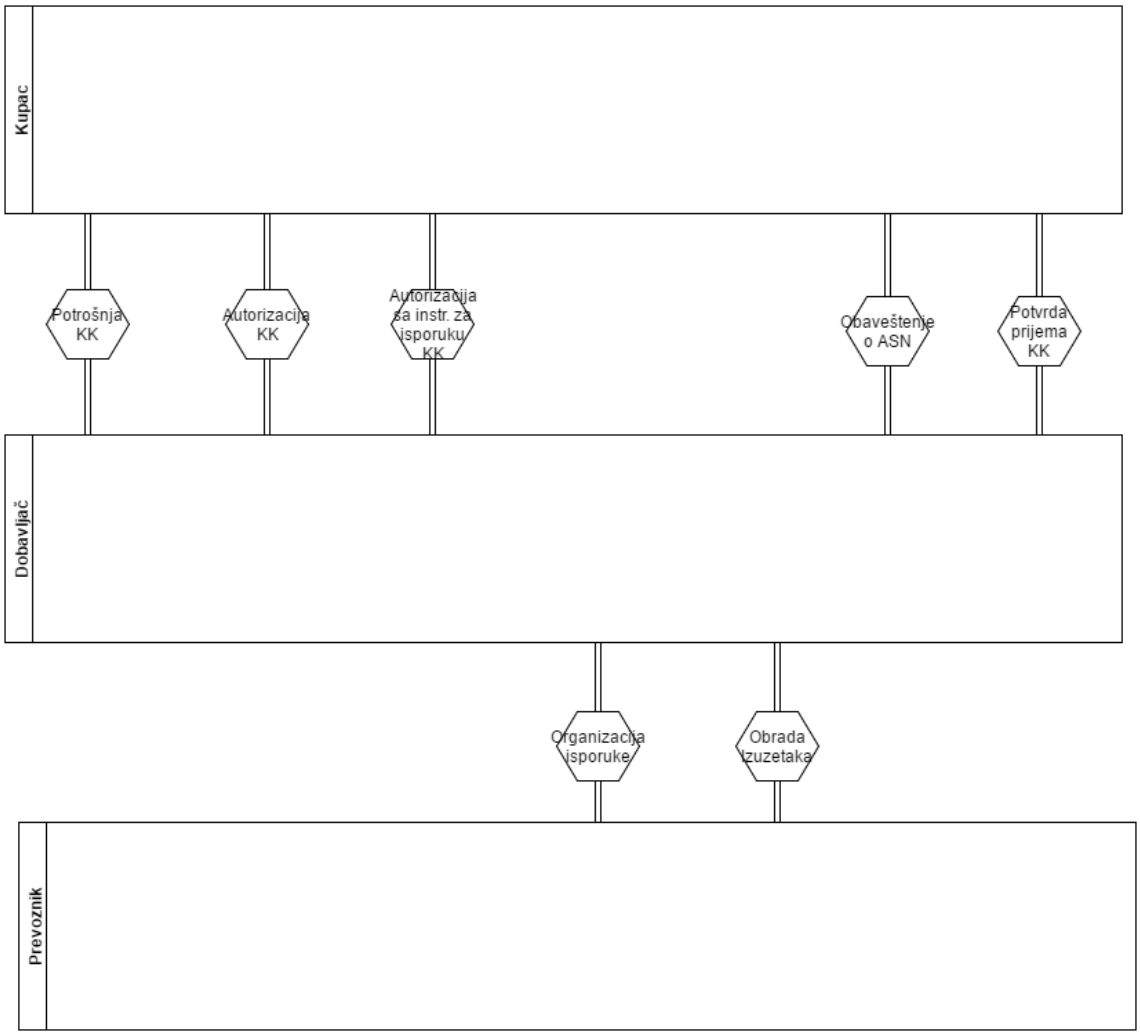
Tabela 6.6. Specifikacija aspekata interoperabilnosti – Larmanova metoda

SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI **FAZA:** Identifikacija **KORAK:** Identifikacija zahteva za interoperabilnošću	
Metodologija	Larmanova metoda razvoja softvera
Tehnika	Model slučajeva korišćenja
Preduslov	Idenitifikacija modela esencijalnih poslovnih funkcija
Proširenje postojeće tehnike	Slučajeve korišćenja koji treba da podrže interoperabilnost i aktore koji predstavljaju kolaborativne partnere treba označiti drugom bojom (npr. plavom).
Uvođenje nove tehnike	Model interoperabilnih slučajeva korišćenja
Rezultat	Identifikacija poslovnih funkcija koje zahtevaju interoperabilnost i kolaborativnih poslovnih partnera Kreiran je Model interoperabilnih slučajeva korišćenja
Prednosti	Polazni model esencijalnih poslovnih funkcija je lako identifikovati na osnovu dijagrama slučajeva korišćenja Laka identifikacija kolaborativnih poslovnih partnera i interoperabilnih slučajeva korišćenja na osnovu njihovih veza na dijagramu slučajeva korišćenja

6.3.1.2. Definisanje kolaboracija

S obzirom na to da su u prvom koraku identifikovani poslovni partneri i esencijalni slučajevi korišćenja koji zahtevaju interoperabilnost, u ovom koraku je potrebno definisati poruke koje se razmenjuju pri njihovoj kolaboraciji. U tezi se predlaže da se za identifikaciju kolaboracija koristi *Model interoperabilnih slučajeva korišćenja*.

Na osnovu identifikovanih slučajeva korišćenja su identifikovane logičke grupe poruka koje su prikazane zbirno na dijagramu konverzacije (slika 6.30).



Slika 6.30 Dijagram konvezacije za Izvršenje IV&I eKanban poslovnog procesa

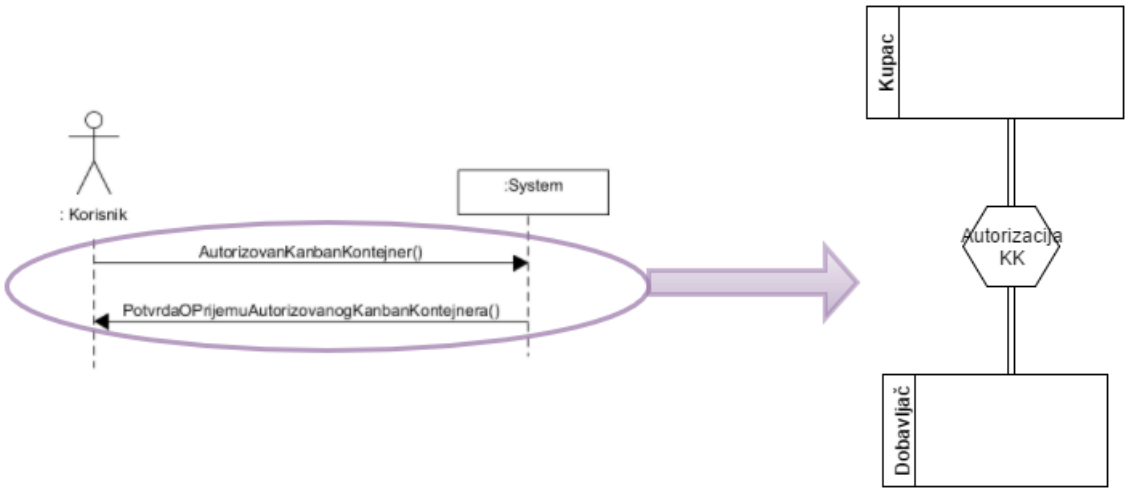
Možemo da primetimo da su sve poruke prikazane na jednom dijagramu konverzacije *Izvršenje IV&I eKanban poslovnog procesa*, dok se na primeru FON Labis metodologije predlaže kreiranje više dijagrama konverzacije različite složenosti. Za njihovo kreiranje je korišćena analogija sa hijerarhijskom dekompozicijom sistema primenom SSA metode. Za Larmanovu metodu je u ovoj fazi karakteristična objektna analiza koja je se bazira na tekstualnoj analizi verbalnog problema, pa se iz tog razloga predlaže drugi pristup za definisanje globalnih konverzacija.

Primenom Larmanove metode izvršena je detaljna analiza identifikovanih interoperabilnih slučajeva korišćenja. Opis slučaja korišćenja *Evidentiranje autorizovanog Kanban Kontejnera* je prikazan u tabeli 6.7.

Tabela 6.7 Slučaj korišćenja SK3: Evidentiranje autorizacije Kanban Kontejnera

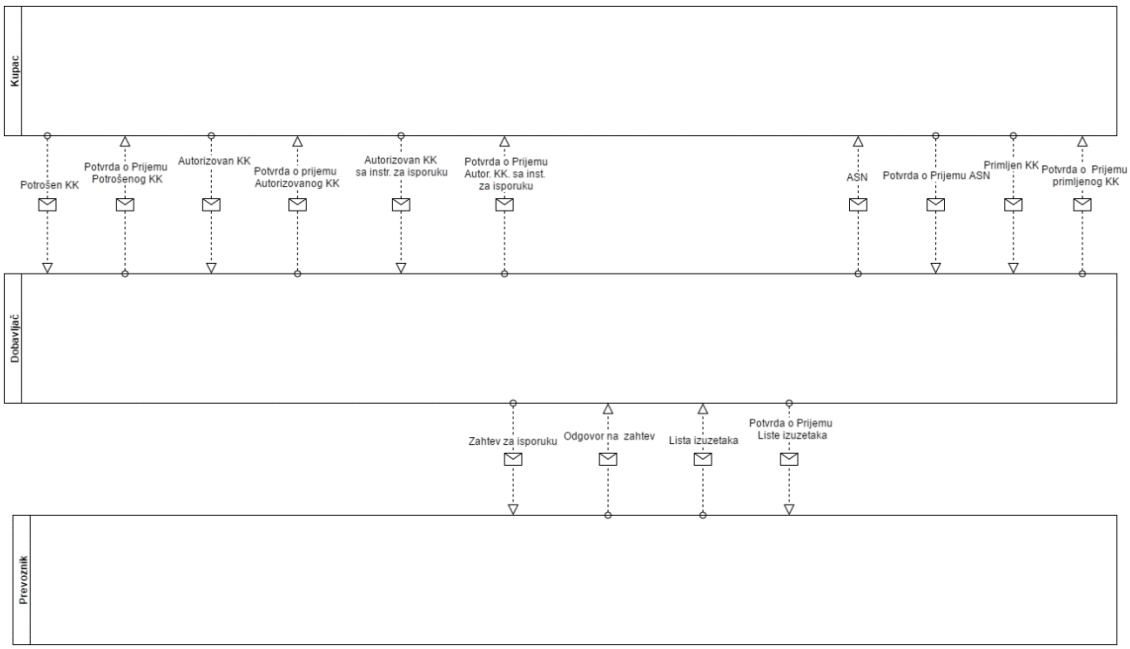
Slučaj korišćenja SK3: Evidentiranje autorizacije Kanban Kontejnera		
Okvir (Scope)	eKanban IV&I poslovni scenario	
Nivo (Level)	Korisnički cilj	
Primarni aktor	IV&I alat Kupca	
Interesne strane	IV&I alat Dobavljača	
Preduslovi	-Sistem je uključen i može da primi poruku -Postoji aktivan Kanban loop sa barem jednim Kanban kontejnerom čiji je status „PRAZAN“. -Definisana su pravila kada prazan Kanban kontejner (čiji je status postavljen na „PRAZAN“) treba da se autorizuje za isporuku. Postoje različiti tipovi pravila za autorizaciju: • Vremenska pravila-unapred određeni fiksni intervali vremena u kojima se autorizuju svi prazni Kanban kontejneri • Veličina pošiljke-unapred određena ukupna težina Kanbana kao uslov autorizacije • „ILI“ i „I“uslov-specifična pravila autorizacije koja mogu da obuhvate više različitih uslova.	
Garancija uspeha (Sucess Guarantee)	Status Kanban kontejnera u IV&I alatu i/ili ostalim alatima koji primaju Kanban signal je postavljen na „AUTORIZOVAN“. Dobavljač je informisan o autorizaciji Kanban kontejnera.	
Osnovni scenario	**Akcija učesnika**	**Odgovor sistema**
	1. IV&I alat kupca šalje poruku za Autorizaciju Kanban Kontejnera	2. Sistem prima poruku za Autorizaciju Kanban Kontejnera
		3. Šalje poruku za potvrdu prijema poruke za Autorizaciju
Alternativni scenario	-	

Na osnovu detaljnog opisa osnovnog scenarija se pravi poseban sistemski dijagram sekvenci. Na slici 6.31 je prikazan sistemski dijagram sekvenci za slučaj korišćenja *Autorizacija Kanban Kontejnera*.



Slika 6.31 Veza između sistemskog dijagrama sekvenci za slučaj korišćenja Autorizovan Kanban kontejner i konverzacije Autorizacija Kanban kontejnera

Na osnovu detaljne analize tekstualnih slučajeva korišćenja su identifikovane pojedinačne poruke. Pojedinačne poruke su prikazane na sistemskom dijagramu sekvenci i detaljnije su razrađene kreiranjem Ugovora za svaku od sistemskih operacija pojednačno. Dijagram kolaboracije koji je rezultat idenitifikacije poruka za svaki od slučajeva korišćenja sa *Modela interoperabilnih poslovnih procesa* je prikazan na slici 6.32. Opisani postupak i rezultati su sumirani u tabeli 6.8.



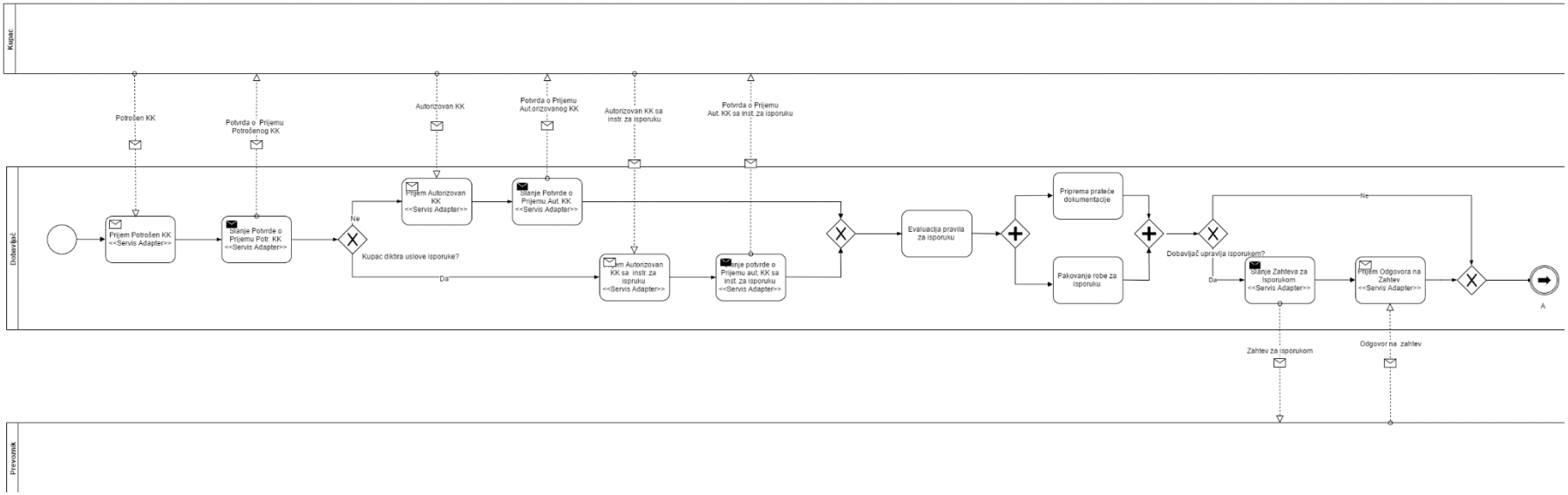
Slika 6.32 Dijagram kolaboracije za Izvršenje IV&I eKanban poslovnog procesa

Tabela 6.8 Definisanje kolaboracija

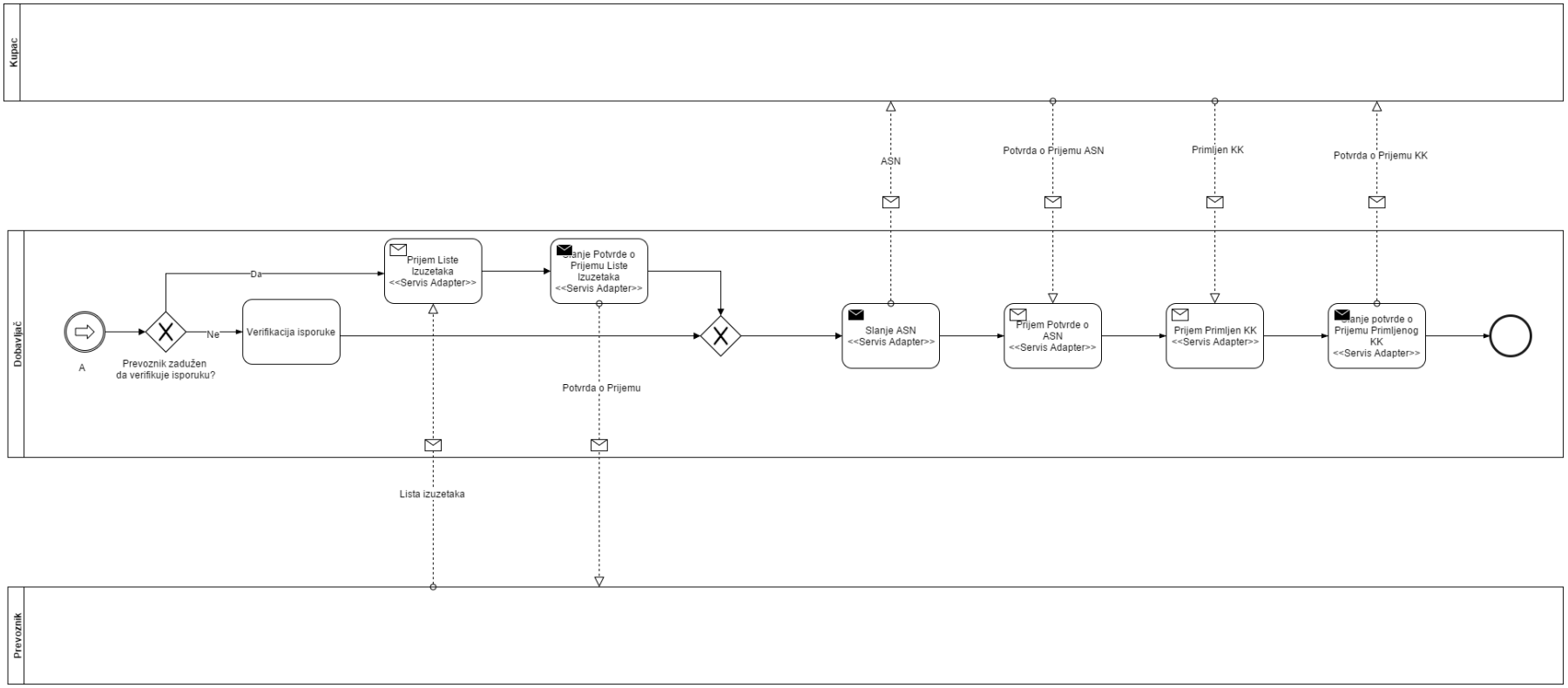
SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI **FAZA:** Identifikacija **KORAK:** *Definisanje kolaboracija*	
Metodologija	Larmanova metoda razvoja softvera
Tehnika	BPMN
Preduslov	Kreiranje Modela slučajeva interoperabilnih poslovnih funkcija
Proširenje postojeće tehnike	-
Uvođenje nove tehnike	-
Rezultat	Izrada dijagrama konverzacije Izrada dijagrama kolaboracije
Prednosti	Olakšana identifikacija poruka između poslovnih partnera na osnovu sistemskih dijagrama sekvenci

6.3.1.1. Specifikacija privatnog i javnog kolaborativnog procesa

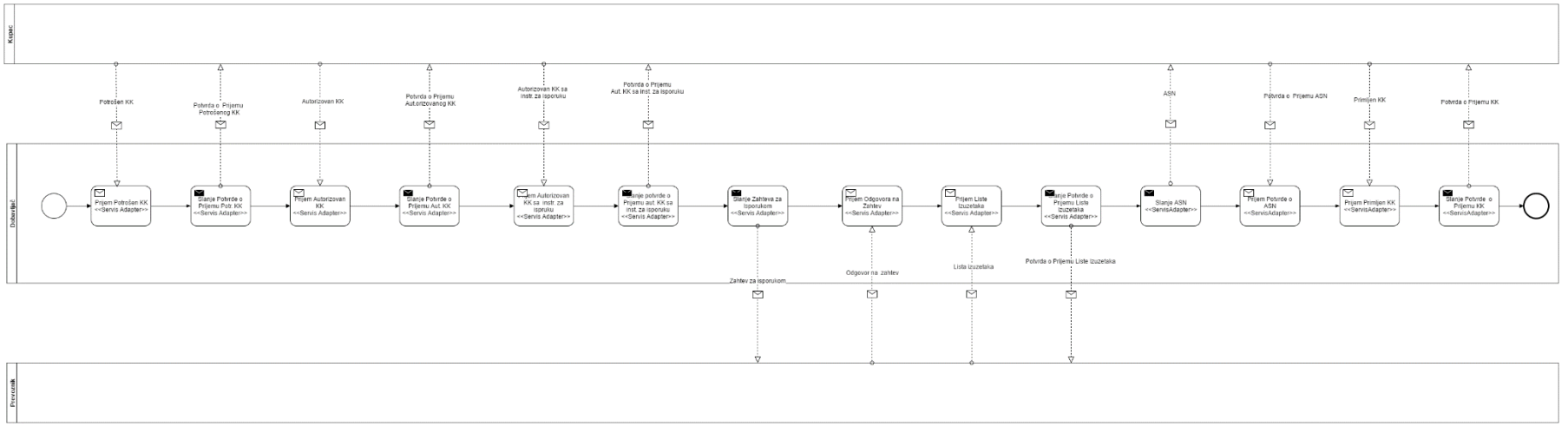
Na osnovu dijagrama kolaboracije je kreirana privatna reprezentacija kolaborativnog poslovnog procesa *Izvršenje IV&I eKanban*-a koja je prikazana na slikama 6.33 i 6.34. Na osnovu privatne reprezentacije kolaborativnog poslovnog procesa, apstrakcijom njegovih privatnih aktivnosti kreiran je privatni proces koji je prikazan na slici 6.35.



Slika 6.33 Privatni proces Izvršenje Kanban kontejnera –prvi deo



Slika 6.34 Privatni proces Izvršenje Kanban kontejnera –drugi deo



Slika 6.35 Javni proces Izvršenje Kanban kontejnera

Tabela 6.9 Specifikacija privatnog i javnog procesa – Larmanova metoda

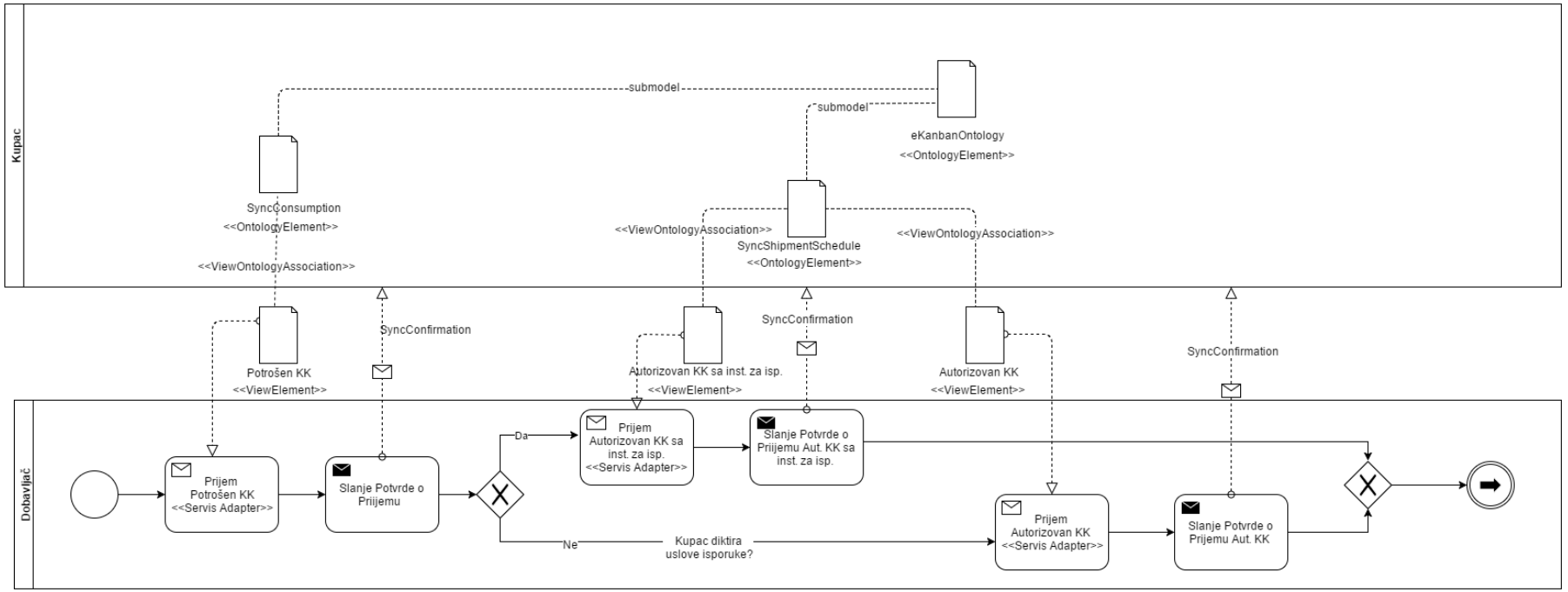
SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI **FAZA:** Identifikacija **KORAK:** Specifikacija privatnog i javnog kolaborativnog procesa	
Metodologija	Larmanova metoda
Tehnika	BPMN
Preduslov	Izrada dijagrama kolaboracije
Proširenje postojeće tehnike	Uvođenje stereotipa <<Servis Adapter>>
Uvođenje nove tehnike	-
Rezultat	Kreiranje privatne reprezentacije kolaborativnog procesa Apstrakcija javne na osnovu privatne reprezentacije kolaborativnog pocesa
Prednosti	Laka identifikacija interaktivnih poslovnih aktivnosti na osnovu Modela interoperabilnih slučajeva korišćenja Olakšana identifikacija poslovnih aktivnosti dijagrama slučajeva korišćenja

Rezultati koraka specifikacije privatnog i javnog procesa primenom Larmanove metode su prikazani u tabeli 6.9.

6.3.2. Realizacija aspekata interoperabilnosti

6.3.2.1. Specifikacija poruka privatnog kolaborativnog procesa kao pogleda nad globalnom Referentnom Ontologijom

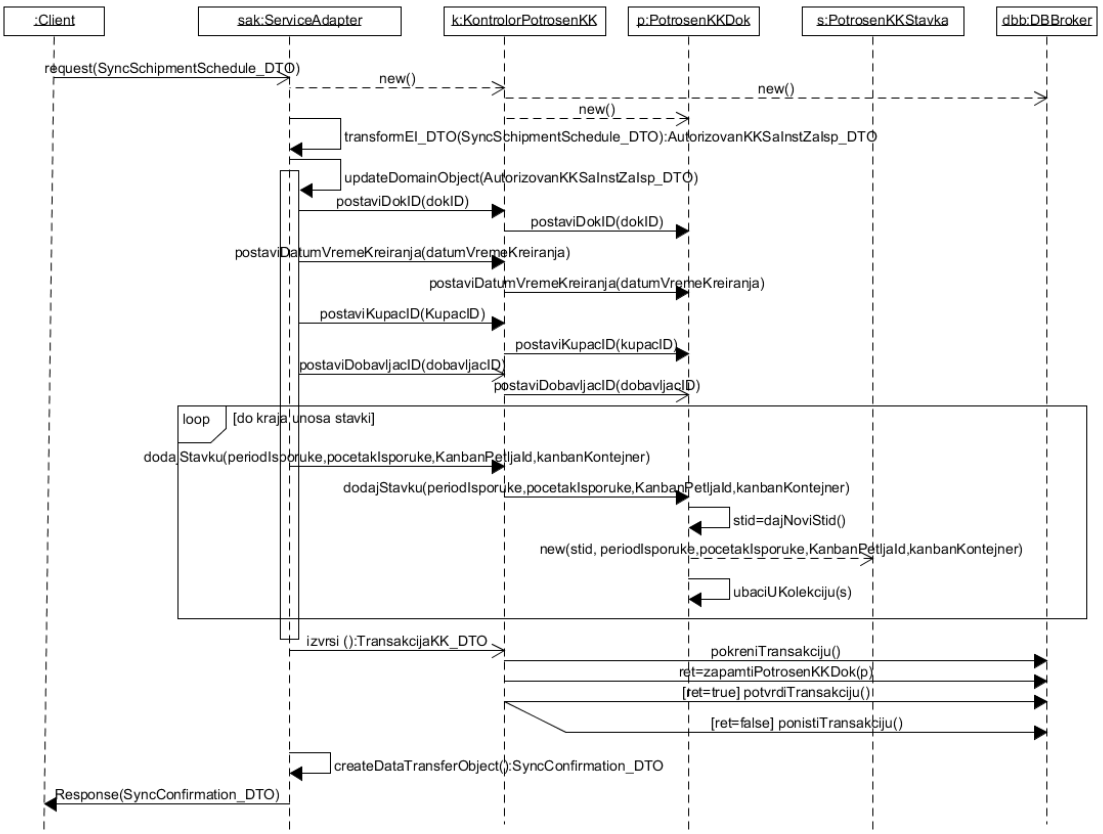
Na slici 6.36. je prikazan deo privatnog procesa „Izvršenje Kanban Kanban Kontejnera“ i njegova veza sa eKanban referentnom ontologijom.



Slika 6.36 Privatni proces Izvršenje Kanban kontejnera i njegova veza sa eKanban referentnom ontologijom

6.3.2.2. Specifikacija Servis Adaptera

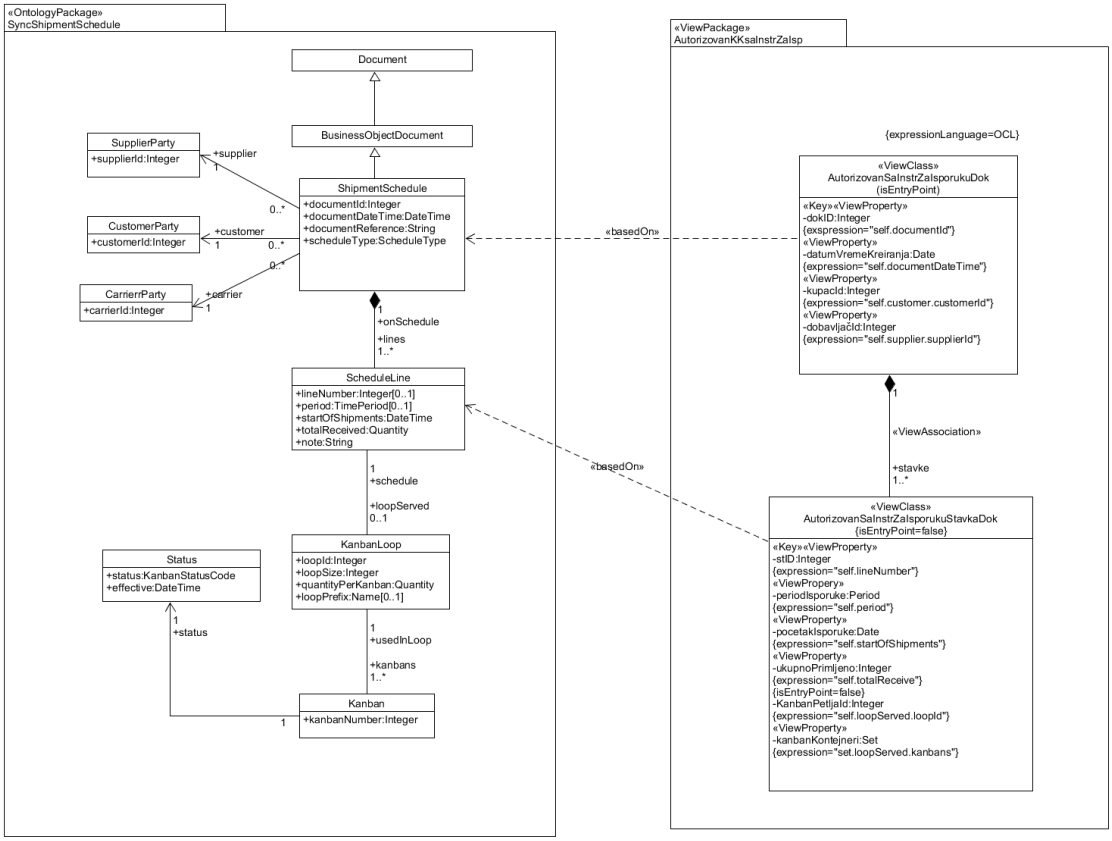
U ovom poglavlju će biti prikazan način specifikacije pojedinačnih servisa primenom Servis Adapter paterna. Za ilustraciju predloženog postupka je odabrana aktivnost *Autorizacija Kanban Kontejnera sa instrukcijama za isporuku* kolaborativnog poslovnog procesa *Izvršenje Kanban Kontejnera*. Kupac šalje Dobavljaču poruku *SyncShipmentSchedule* kako bi signalizirao da je Kanban kontejner potrošen. Na slici 6.37 je prikazan odgovarajući dijagram sekvenci koji ilustruje pravila za specifikaciju Servis Adapter paterna.



Slika 6.37 Dijagram sekvenci Autorizovan KK sa inst. za isporuku

6.3.2.3. Specifikacija strukture poruka preko UML View Profila

Na slici 6.38 je prikazana struktura standardne IV&I poruke *SyncSchipmentSchedule*, koju Kupac šalje Dobavljaču da ga autorizuje za isporuku Kanban kontejnera. Odgovarajuća struktura lokalne poruke je prikazana paketom *AutorizovanKKSaInstrZaIsp*.



Slika 6.38 Autorizovan KK sa inst. za isporuku kao pogled nad eKanban referentnom ontologijom

6.4. Zaključna razmatranja

U ovom poglavlju je prikazana primena predloženog pristupa za specifikaciju aspekata interoperabilnosti na primeru dovoljno kompleksnog realnog sistema. Za potrebe validacije predloženog pristupa je odabran B2B eKanban poslovni protokol koji je standardizovan od strane AIAG kao deo IV&I projekta. Ilustrovana je primena predloženog pristupa koristeći FON Labis metodologiju za razvoj IS i Larmanovu metodu razvoja softvera. U oba slučaja je pokazano, da je primenom predloženog pristupa moguća specifikacija aspekata interoperabilnosti.

Osnovna prednost FON Labis metodološkog pristupa se ogleda u primeni metode stukturne sistemske analize za funkcionalnu dekompoziciju sistema. Kandidate za suštinske poslovne procese je lako identifikovati na osnovu primitivnih poslovnih funkcija koje komuniciraju sa spoljnim interfejsima. Hijerarhijska dekompozicija sistema u velikoj meri olakšava identifikaciju kolaborativnih poslovnih procesa i definisanje polaznih globalnih konverzacija između poslovnih partnera. Izlazni i

ulazni tokovi ka interfejsima su pogodni za identifikaciju poruka kolaborativnih poslovnih procesa. Hijerarhijska dekompozicija sistema je bila od koristi i pri definisanju sekvence aktivnosti privatnog kolaborativnog poslovnog procesa.

Prednosti primene FON Labis metodologije u odnosu na Larmanovu metodu su evidentne u fazi *identifikacije* sistema. Larmanova metoda, kao objektno-orijentisana metoda razvoja softvera nema tehniku koja bi bila analogna SSA. Ipak, i Larmanova metoda nudi izvesne pogodnosti i olakšava identifikaciju sistema. Na primer, idenitifikacija interoperabilnih slučajeva korišćenja i kolaborativnih poslovnih partnera je moguća na osnovu polaznog dijagrama slučajeva korišćenja. Sistemski dijagrami sekvenci su se pokazali korisni pri identifikaciji poruka između kolaborativnih poslovnih partnera.

Obe metodologije predlažu izradu logičkog konceptualnog modela, koji je bio od koristi pri definisanju struture lokalnih poruka i njihovog mapiranja na koncepte globalne referentne ontologije. Postupak specifikacije opšte logičke arhitekture u skladu sa predloženim Servis Adapter paternom je identičan za obe metodologije.

7. ZAKLJUČAK

Osnovni cilj ove doktorske disertacije je bio da se definiše novi originalni pristup za specifikaciju aspekata interoperabilnosti u metodološkim pristupima razvoju informacionih sistema.

Predloženi pristup za specifikaciju aspekata interoperabilnosti je baziran na opštem sistemsko-teorijskom pristupu razvoju softvera. Pokazano je da su tri osnovne faze sistemsko-teorijskog modela životnog ciklusa: identifikacija, realizacija i implementacija dovoljno opšte i da se identifikovani koraci za svaku od faza mogu primeniti za specifikaciju aspekata interoperabilnosti u različitim metodološkim pristupima za razvoj informacionih sistema.

U fazi *identifikacije* je pokazano da je moguće izvršiti identifikaciju osnovnih elemenata kolaborativnih poslovnih procesa i da je BPMN tehnika pogodna za njihovu reprezentaciju na različitim nivoima apstrakcije: počev od opštih dijagrama konverzacije, do mogućnosti definisanja privatnog i javnog kolaborativnog procesa. Glavni doprinos disertacije je vezan za fazu *realizacije*, gde je omogućena detaljna specifikacija privatnog kolaborativnog poslovnog procesa i njegova veza sa referentnom ontologijom. U prvom koraku faze *realizacije* je izvršena ekstenzija BPMN meta-modela, koja je omogućila da se poruke BPMN privatnog kolaborativnog poslovnog procesa definišu kao pogledi nad referentnom ontologijom. U narednom koraku je predložen način specifikacije servisa, korišćenjem Servis Adapter paterna koji može da omogući lakšu transformaciju specifikacije u programski kod. U poslednjem koraku je definisan UML Profil, za detaljnu specifikaciju strukture poruka. Naime, UML Profil je omogućio mapiranje lokalne konceptualne šeme na odgovarajuće koncepte globalne referentne ontologije, definisanjem odgovarajućih OCL pravila. Predloženi metod za specifikaciju aspekata interoperabilnosti u metodološkim pristupima razvoju IS je uspešno primenjen na IV&I eKanban poslovnom scenariju za razmenu B2B poslovnih poruka. Time je potvrđena, a i praktično demonstrirana generalna

hipoteza disertacije da je moguće definisati proširenje postojećih metodoloških pristupa za razvoj IS kako bi se obezbedila podrška za realizaciju interoperabilnosti. Potvrđene su i pojedinačne hipoteze, odnosno moguće je:

- Identifikovati i sistematizovati interoperabilne zahteve
- Formalizovati proširenje postojećih modela i tehnika
- Formalizovati uvođenje novih modela i tehnika
- Verifikovati ostvarene rezultate proširenja metodoloških pristupa na primeru dovoljno kompleksnog realnog sistema.

7.1. Ostvareni doprinosi

Na osnovu razmatranja u prethodnim poglavljima, sa ciljem sistematizacije predloženog rešenja, navode se naučni i stručni doprinosi ove disertacije.

Naučni doprinosi su:

- Definisanje novog originalnog pristupa za specifikaciju aspekata interoperabilnosti koji je zasnovan na opštem sistemsko-teorijskom pristupu.
- Razvoj novih modela za detaljnu specifikaciju kolaborativnih procesa i informacionih tokova.
- Razvoj novih modela za specifikaciju informacionih zahteva kao pogleda nad zajedničkom referentnom ontologijom.
- Definisanje proširenja BPMN meta-modela za asocijaciju definisanih pogleda nad zajedničkom referentnom ontologijom sa relevantnim aktivnostima kolaborativnog poslovnog procesa.
- Analiza i kritički osvrt na dosadašnja istraživanja u predmetnoj oblasti sa posebnim akcentom na relevantne metodološke pristupe.

Ostvareni stručni i praktični doprinosi ove disertacije su:

- Sistematizacija dostignuća oblasti metodoloških pristupa za razvoj informacionih sistema i doprinos pregledu radova sa temom specifikacije aspekata interoperabinosti.
- Sistematizacija kriterijuma za validaciju specifikacije aspekata interoperabilnosti u metodološkim pristupima za razvoj informaconih sistema.
- Primena predloženog pristupa u procesu specifikacije aspekata interoperabilnosti izabranog domena poslovanja.

7.2. Pravci daljeg istraživanja

Dalji pravac istraživanja biće na primeni predloženog pristupa za specifikaciju aspekata interoperabilnosti na ostalim metodološkim pristupima. Jedna od tema u okviru ovog pravca bila bi istraživanje mogućnosti da se i drugi jezici za modelovanje poslovnih procesa, izaberu kao osnova za modelovanje kolaboracije.

Drugi pravac istraživanja imao bi za cilj izradu CASE alata koji bi podržao predloženi pristup. Taj CASE alat treba da poseduje grafički editor za definisanje pogleda nad referentnom ontologijom, odnosno treba da omogući mapiranje lokalne konceptualne šeme na relevantne koncepte referentne ontologije u skladu sa definicijom UML View Profila. Takođe sastavni deo CASE alata, bio bi i grafički editor za kreiranje modela kolaboracije u skladu sa predloženom ekstenzijom BPMN meta-modela, kako bi se na dijagramu omogućila adekvatna reprezentacija poruka i njihove veze sa Referentnom Ontologijom.

Posebno interesantan pravac istraživanja je razvoj generatora koda koji treba da na osnovu predloženog pristupa specifikacije aspekata interoperabilnosti, što više automatizuje proces implementacije. Usled čestih promena u uslovima opšte globalizacije poslovanja, mogući dalji pravac istraživanja je adaptivni pristup specifikaciji aspekata interoperabilnosti.

8. LITERATURA

Abbott, R. J. (1983). Program design by informal English descriptions. *Communications of the ACM*, *26*(11), 882–894.

Abran, A., Moore, J. W., Bourque, P., & Dupuis, R. (Eds.). (2004). *Guide to the software engineering body of knowledge, 2004 version: SWEBOK ; a project of the IEEE Computer Society Professional Practices Committee*. Los Alamitos, Calif.: IEEE Computer Society.

Aier, S., & Schönherr, M. (2005). Evaluating Integration Architectures – A Scenario-Based Evaluation of Integration Technologies. In D. Draheim & G. Weber (Eds.), *Trends in Enterprise Application Architecture* (pp. 2–14). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/11681885_2

Alexander, C. (1999). The origins of pattern theory: the future of the theory, and the generation of a living world. *IEEE Software*, *16*(5), 71–82. http://doi.org/10.1109/52.795104

Ambler, S. W., Nalbone, J., & Vizdos, M. J. (2005). *The Enterprise Unified Process: Extending the Rational Unified Process (First edition)*. Upper Saddle River, NJ: Prentice Hall.

Aničić, N. (2006). Integracija poslovnih aplikacija primenom tehnologija semantičkog Weba, Doktorska disertacija. Univerzitet u Beogradu, Fakultet Organizacionih Nauka.

ATHENA A1.1. (2004). *First Version of State of the Art in Enterprise Modelling Techniques and Technologies to Support Enterprise Interoperability Work package – A1.1* (Enterprise modeling in the Context of Collaborative Enterprises). Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Appliction (ATHENA) Project No 507849. Retrieved from http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/A1/d-a1-1.1

ATHENA A1.2.1. (2005). *Model and Report: First Set of Requirements Work package – A1.2.1.* (No. A1.2.1.).

ATHENA A2.2. (2005). *Specification of a Cross-Organisational Business Process Model* (No. D.A2.2). Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Appliction (ATHENA) Project No 507849.

ATHENA A6.3. (2006). *Model-driven and Adaptable Interoperability Framework* (Deliverable No. A6.3). Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Appliction (ATHENA) Project No 507849. Retrieved from http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/A6/d-a6.3

ATHENA DA1.3.1. (2005). *Report on Methodology description and guidelines definition* (Deliverable No. DA1.3.1). Retrieved from http://interop-

vlab.eu/ei_public_deliverables/athena-deliverables/A1/d-a1-3.1/?searchterm=athena%20POP

ATHENA D.A2.1. (2005). *Cross-Organisational Business Process requirements and the State of the Art in Research, Technology and Standards* (Deliverable No. A2.1) (pp. 1–111). Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Appliction (ATHENA) Project No 507849. Retrieved from http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/A2/d-a2.1

ATHENA D.A4.2. (2007). *Specification of Interoperability Framework and Profiles, Guidelines and Best Practices* (Deliverable No. D.A4.2) (pp. 1–215). Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Appliction (ATHENA) Project No 507849. Retrieved from http://interop-vlab.eu/ei_public_deliverables/athena-deliverables/A4/d-a4.2

ATHENA WD.B5.7.1. (2007). *B5.10 Inventory Visibility Sub-Project : ATHENA Enterprise Modelling Area (A1) Results Validation Work package – B5.5, Version 1.0* (Deliverable No. WD.B5.7.1) (pp. 1–83). Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Appliction (ATHENA) Project No 507849.

Avison, D., & Fitzgerald, G. (2006). *Information systems development* (4th edition). London: McGraw-Hill Education.

Baclawski, K., Kokar, M. K., Kogut, P. A., Hart, L., Smith, J., Letkowski, J., & Emery, P. (2002). Extending the Unified Modeling Language for ontology development. *Software and Systems Modeling*, *1*(2), 142–156. http://doi.org/10.1007/s10270-002-0008-4

Barkmeyer, E., & Denno, P. (2007). On capturing information requirements in process specifications. , Springer, London, 365-376. In *Enterprise Interoperability II* (pp. 365–376). Springer, London.

Barkmeyer, E., & Kulvatunyou, B. (2007). An ontology for the e-Kanban business process. *National Institute of Standards and Technology, NISTIR*, *7404*.

Barnickel, N., Böttcher, J., & Paschke, A. (2010). Incorporating Semantic Bridges into Information Flow of Cross-organizational Business Process Models. In *Proceedings of the 6th International Conference on Semantic Systems* (p. 17:1–17:9). New York, NY, USA: ACM. http://doi.org/10.1145/1839707.1839729

Bass, L., Clements, P., & Kazman, R. (2012). *Software Architecture in Practice*. Addison-Wesley.

Bastida, L., Berre, A., Elvesaeter, B., Hahn, C., & Johnses, S. (2009). *Model-driven metodology and architecture specification. Deliverable D2.1, SHAPE, Project number 216408*.

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., … Dave, T. (2001). Manifesto for Agile Software Development. Retrieved May 29, 2016, from http://agilemanifesto.org/

Bente, S., Bombosch, U., & Langade, S. (2012). *Collaborative enterprise architecture: enriching EA with lean, agile, and enterprise 2.0 practices*. Morgan Kaufman.

Bernus, P., Nemes, L., & Williams, T. J. (Eds.). (1996). *Architectures for Enterprise Integration*. Boston, MA: Springer US. Retrieved from http://link.springer.com/10.1007/978-0-387-34941-1

Berre, A., & Doumeingts, G. (2004). *State of the art of methods to derive IT specifications from models and to develop IT specific components based on IT modelling, WP7, INTEROP Network of Exellence-Contract no.: IST-508011*.

Berre, A.-J., Elvesæter, B., Figay, N., Guglielmina, C., Johnsen, S. G., Karlsen, D., … Lippe, S. (2007). The ATHENA Interoperability Framework. In R. J. Gonçalves, J. P. Müller, K. Mertins, & M. Zelm (Eds.), *Enterprise Interoperability II* (pp. 569–580). Springer London. Retrieved from http://link.springer.com/chapter/10.1007/978-1-84628-858-6_62

Berre, A.-J., Hahn, A., Akehurst, D., Bezivin, J., Tsalgatidou, A., Vermaut, F., … Linington, P. F. (2004). *State-of-the art for Interoperability architecture approaches* (Deliverable No. D9.1) (pp. 1–366). SINTEF, Network of Excellence-Contract no.: IST-508 011. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106.9713&rep=rep1&type=pdf

Blaha, M. R., & Rumbaugh, J. R. (2004). *Object-Oriented Modeling and Design with UML* (2 edition). Upper Saddle River, NJ: Pearson.

Blanc, S. (2005). Interoperability problems: Management of evolution of collaborative enterprises. Presented at the Interop-ESA '05, Doctoral Symposium, Geneva.

Booch, G., Maksimchuk, R. A., Engle, M. W., Young, B. J., Conallen, J., & Houston, K. A. (2007). *Object-Oriented Analysis and Design with Applications* (3 edition). Upper Saddle River, NJ: Addison-Wesley Professional.

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide* (2 edition). Upper Saddle River, NJ: Addison-Wesley Professional.

Chandrasekaran, B., Josephson, J. R., & Benjamins, V. R. (1999). What Are Ontologies, and Why Do We Need Them? *IEEE Intelligent Systems*, *14*(1), 20–26. http://doi.org/10.1109/5254.747902

Chen, D., & Doumeingts, G. (2003). European initiatives to develop interoperability of enterprise applications—basic concepts, framework and roadmap. *Annual Reviews in Control*, *27*(2), 153–162. http://doi.org/10.1016/j.arcontrol.2003.09.001

Chen, D., Doumeingts, G., & Vernadat, F. (2008). Architectures for enterprise integration and interoperability: Past, present and future. *Computers in Industry*, *59*(7), 647–659. http://doi.org/10.1016/j.compind.2007.12.016

Chen, D., & Vernadat, F. B. (2003). Enterprise Interoperability: A Standardization View. In *Enterprise Inter- and Intra-Organizational Integration*. Boston, MA: Springer US.

Coad, P., & Yourdon, E. (1990). *Object Oriented Analysis* (2 edition). Englewood Cliffs, N.J: Prentice Hall PTR.

Cockburn, A. (2001). *Writing Effective Use Cases*. Addison-Wesley. Retrieved from http://www.amazon.com/Writing-Effective-Cases-Alistair-Cockburn/dp/0201702258

Cockburn, A. (2006). *Agile Software Development: The Cooperative Game* (2 edition). Upper Saddle River, NJ: Addison-Wesley Professional.

Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional.

Constantine, L. L., & Lockwood, L. A. D. (1999). *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*. Pearson Education.

Coplien, J. O., & Schmidt, D. (1995). *Pattern Languages of Program Design* (1 edition). Reading, Mass: Addison-Wesley Professional.

Cranefield, S., & Purvis, M. (1999). UML as an Ontology Modelling Language. In *In Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99* (pp. 46–53).

Curtis, B., Kellner, M. I., & Over, J. (1992). Process Modeling. *Commun. ACM*, *35*(9), 75–90. http://doi.org/10.1145/130994.130998

Daigneau, R. (2012). *Service design patterns: fundamental design solutions for SOAP/WSDL and RESTful Web services*. Upper Saddle River, NJ: Addison-Wesley.

Davenport, T. H. (1993). *Process Innovation: Reengineering Work Through Information Technology*. Boston, MA, USA: Harvard Business School Press.

Denis Gagné, & Conrad Bock. (2014). *Feasibility of Integrating OAGIS and BPMN* (White Paper No. #20141014BPMNV1.0). U.S. National Institute of Standards and Technology.

Doumeingts, G., Vallespir, B., & Chen, D. (1998). GRAI Grid Decisional modelling. In *Handbook on architectures of information systems* (pp. 321–346). Springer. Retrieved from http://link.springer.com/content/pdf/10.1007/3-540-26661-5_14.pdf

Erl, T. (2004). *Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*. Upper Saddle River, NJ: Prentice Hall.

Erl, T. (2007). *SOA: Principles of Service Design* (1st edition). Upper Saddle River, NJ: Prentice Hall.

ESPRIT Consortium AMICE (Ed.). (1993). *CIMOSA: Open System Architecture for CIM*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-642-58064-2

EUP. (2016). Enterprise Unified Process (EUP): Strategies for Enterprise Agile. Retrieved June 4, 2016, from http://www.enterpriseunifiedprocess.com/

Fowler, M. (1996). *Analysis Patterns: Reusable Object Models* (1 edition). Menlo Park, Calif: Addison-Wesley Professional.

Fowler, M. (2003). *Patterns of enterprise application architecture*. Boston: Addison-Wesley.

Fraunhofer ISST. (2009). *Semantic Interoperability Centre Europe-Study on Methodology, Version1.2*.

Gao, F., Derguech, W., & Zaremba, M. (2011). Extending BPMN 2.0 to Enable Links between Process Models and ARIS Views Modeled with Linked Data. In W. Abramowicz, L. Maciaszek, & K. Węcel (Eds.), *Business Information Systems Workshops* (pp. 41–52). Springer Berlin Heidelberg. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-25370-6_5

Gašević, D., Djurić, D., & Devedžić, V. (2006). *Model driven architecture and ontology development*. Berlin: Springer.

Giaretta, P., & Guarino, N. (1995). Ontologies and knowledge bases towards a terminological clarification. *Towards Very Large Knowledge Bases: Knowledge Building & Knowledge Sharing*, *25*, 32.

Gómez-Pérez, A. (2004). *Ontological engineering : with examples from the areas of knowledge management, e-commerce and the semantic Web / Asunción Gómez-Pérez, Mariano Fernández-López, and Oscar Corcho.* London ; Springer-Verlag,.

Graham, I. (2008). *Requirements modelling and specification for service oriented architecture*. Chichester, England ; Hoboken, NJ: Wiley.

Gruber, T. R. (1993). A Translation Approach to Portable Ontology Specifications. *Knowl. Acquis.*, *5*(2), 199–220. http://doi.org/10.1006/knac.1993.1008

Gruber, T. R. (1995). Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*, *43*(5), 907–928.

Guarino, N. (1997). Understanding, Building and Using Ontologies. *Int. J. Hum.-Comput. Stud.*, *46*(2–3), 293–310. http://doi.org/10.1006/ijhc.1996.0091

Hay, D. C. (2006). *Data Model Patterns: A Metadata Map*. San Francisco, CA: Elsevier.

Hepp, M. (Ed.). (2008). *Ontology management: semantic web, semantic web services, and business applications*. New York, NY: Springer.

Highsmith, J., & Cockburn, A. (2001). Agile software development: the business of innovation. *Computer*, *34*(9), 120–127. http://doi.org/10.1109/2.947100

IEEE. (1991). IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. *IEEE Std 610*, 1–217. http://doi.org/10.1109/IEEESTD.1991.106963

ISA. (2010). *European Interoperability Framework (EIF) for European public services*. European Commision, Bruxelles: Interoperability Solutions for European Public Administrations.

Isaias, P., & Issa, T. (2015). *High Level Models and Methodologies for Information Systems*. New York, NY: Springer New York. Retrieved from http://link.springer.com/10.1007/978-1-4614-9254-2

Jacobson, I., Booch, G., & Rumbaugh, J. (1999). *The Unified Software Development Process* (1 edition). Reading, Mass.: Addison-Wesley Professional.

Jakob Freund, Bernd Rucker. (2012). *Real-Life BPMN: Using BPMN 2.0 to Analyze, Improve and Automate Processes in Your Company*. Lexington, KY: Camunda.

Janković, M. (2007). *Interoperabilnost modela poslovnih sistema, Magistarska teza*. Univerzitet u Beogradu, Fakultet Organizacionih Nauka.

Jankovic, M., Ivezic, N., Knothe, T., Marjanovic, Z., & Snack, P. (2007). A Case Study in Enterprise Modelling for Interoperable Cross-Enterprise Data Exchange. In R. J. Gonçalves, J. P. Müller, K. Mertins, & M. Zelm (Eds.), *Enterprise Interoperability II* (pp. 541–552). Springer London. Retrieved from http://link.springer.com/chapter/10.1007/978-1-84628-858-6_59

Jankovic, M., Ljubicic, M., Anicic, N., & Marjanovic, Z. (2015). Enhancing BPMN 2.0 informational perspective to support interoperability for cross-organizational business processes. *Computer Science and Information Systems*, *12*(3), 1101–1120. http://doi.org/10.2298/CSIS141112013J

Jörg Ziemann, Thomas Matheis, & Jörn Freiheit. (2007). Modelling of Cross-Organizational Business Processes Current Methods and Standards, *2*(2), 23–31.

Kalfoglou, Y. (2001). Exploring Ontologies. In S. K. Chang (Ed.), *Handbook of Software Engineering and Knowledge Engineering* (pp. 863–887). World Scientific Publishing. Retrieved from http://eprints.soton.ac.uk/257428/

Kogut, P., Cranefield, S., Hart, L., Dutra, M., Baclawski, K., Kokar, M., & Smith, J. (2002). UML for ontology development. *The Knowledge Engineering Review*, *17*(1), 61–64. http://doi.org/10.1017/S0269888902000358

Kosanke, K., & Nell, J. G. (Eds.). (1997). *Enterprise Engineering and Integration*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-642-60889-6

Kulak, D., & Guiney, E. (2003). *Use Cases: Requirements in Context* (2 edition). Boston: Addison-Wesley Professional.

Labis. (2015). FON Labis metodologija. Univerzitet u Beogradu, Fakultet organizacionih nauka, Laboratorija za informacione sisteme.

Lankhorst, M. (2013). *Enterprise Architecture at Work*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-642-29651-2

Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition*. United States of America: Addison-Wesley Professional, Pearson Education, Inc.

Lazarević, B., Marjanović, Z., Aničić, N., & Babarogić, S. (2006). *Baze Podataka*. Beograd, Srbija: Univerzitet u Beogradu, Fakultet Organizacionih Nauka.

Lazarević, B., & Nešković, S. (1997). Sistemsko-teorijska kritika objektno-orjentisanih pristupa razvoju softvera. *Info Science*, *5*(4), 17–23.

Lazarević, B., & Nešković, S. (1998). Metodologija modeliranja poslovnih procesa, *INFO*(2), 14–18.

Lazarević, B., & Nešković, S. (1999). Nove arhitekture i pristupi objektno-orijentisanom razvoju softvera. *INFO*, *2–3*.

Lippe, S., Greiner, U., & Barros, A. (2005). A survey on state of the art to facilitate modelling of cross-organisational business processes. *XML4BPM*, *1*, 7–22.

Mallek, S., Daclin, N., & Chapurlat, V. (2011). An approach for interoperability requirements specification and verification. In *Enterprise Interoperability* (pp. 89–102). Springer. Retrieved from http://link.springer.com/chapter/10.1007/978-3-642-19680-5_9

Martin, J., & Odell, J. J. (1997). *Object-Oriented Methods: A Foundation, UML Edition* (2 Sub edition). Upper Saddle River, N.J: Prentice Hall PTR.

Martin, R. C. (2002). *Agile Software Development, Principles, Patterns, and Practices* (1st edition). Upper Saddle River, N.J: Pearson.

Miers, D., Stephen A.White. (2008). *BPMN Modeling and Reference Guide*. Lighthouse Point, Florida: Future Strategies Inc., Lighthouse Pt, FL.

Nešković, S. (2006). *Metodologija i CASE alat za razvoj informacionih sistema postepenom konkretizacijom apstraktnih specifikacija, Doktorska disertacija*. Univerzitet u Beogradu, Fakultet Organizacionih Nauka.

OMG. (1999a). Requirements for UML Profile, Analysis and Design Platform Task Force,.

OMG. (1999b). White paper on Profile mechanism, Analysis and Design Platform Force.

OMG. (2006). *Object Constraint Language (OCL) Specification ver. 2.0, OMG Document formal/2006-05-01*. Retrieved from http://www.omg.org/spec-OCL/

OMG. (2010a). *Unified Modeling Language (UML) Specification ver. 2.0 Infrastructure, OMG Document ptc/2010-11-16*. Retrieved from http://www.omg.org

OMG. (2010b). *Unified Modeling Language (UML) Specification ver. 2.4 Superstructure, OMG Document ptc/2010-11-14*. Retrieved from http://www.omg.org/

OMG. (2014). *Ontology Definition Metamodel Version 1.1* (No. formal/2014-09-02). Retrieved from http://www.omg.org/spec/ODM/1.1/

OMG BPMN 2.0. (2011). *BPMN 2.0 Specification Version 2*. Retrieved from http://www.omg.org/spec/BPMN/2.0/

Open Group. (2000). TOGAF: The Open Group Architecture Framework, Document No. 1910, Version 6.

Ottosson, A. (2005). *An Analysis of a Content of a Method Chunk Repository concernng Interoperability Problems. Master Thesis HS-EA-DVA-2005-001, University of Skovde*.

Peltz, C. (2003). Web Services Orchestration and Choreography. *Computer*, *36*(10), 46–52.

Petit, M., Krogstie, J., & Sindre, G. (2004). *Knowledge map of research in interoperability in INTEROP NoE, Contract no.:IST-508011*.

Pfleeger, S. L., & Atlee, J. M. (2006). *Softversko inženjerstvo Teorija i praksa, prevod trećeg izdanja*. Beograd, Srbija: Računarski fakultet.

Pilone, D., & Pitman, N. (2005). *UML 2.0 in a nutshell* (1st ed). Beijing ; Sebastopol, CA: O'Reilly Media.

PMOV. (2000). Prošireni model objekti-veze. Univerzitet u Beogradu, Fakultet organizacionih nauka, Laboratorija za informacione sisteme. Retrieved from http://pisbp.fon.rs/download/PMOV.pdf

Poler, R., Van Sinderen, M., & Sanchis, R. (Eds.). (2009). *Enterprise Interoperability: second IFIP WG 5.8 International Workshop, IWEI 2009, Valencia, Spain, October 13-14, 2009, Proceedings*. Berlin ; Heidelberg: Springer.

Rozanski, N., & Woods, E. (2011). *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives* (2 edition). Upper Saddle River, NJ: Addison-Wesley Professional.

Rumbaugh, J., Jacobson, I., & Booch, G. (2004). *The Unified Modeling Language Reference Manual,* (2 edition). Boston: Addison-Wesley Professional.

RUP. (2016). RUP The Rational Unified Process product. Retrieved from http://www.ibm.com/software/rational

Russell, S. J., & Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall.

Scheer, A.-W. (1992). *Architecture of Integrated Information Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-642-97389-5

Scheer, A.-W. (1994). *Business Process Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-642-79142-0

Scheer, A.-W. (2013). *ARIS — Business Process Modeling*. Springer Science & Business Media.

Scheer, A.-W., Abolhassan, F., Jost, W., & Kirchmer, M. (Eds.). (2004). *Business Process Automation*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-540-24702-9

Schekkerman, J. (2003). *How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework* (2nd edition). Victoria, B.C.: Trafford Publishing.

Schulz, K. A., & Orlowska, M. E. (2004). Facilitating cross-organisational workflows with a workflow view approach. *Data & Knowledge Engineering*, *51*(1), 109–147. http://doi.org/10.1016/j.datak.2004.03.008

Sheth, A. (Ed.). (2011). *Semantic services, interoperability, and web applications: emerging concepts*. Hershey, PA: Information Science Reference.

Silver, B. (2011). *BPMN method and style: with BPMN implementer's guide* (2. ed). Aptos, Calif: Cody-Cassidy Press.

Sommerville, I., & Sawyer, P. (1997). *Requirements Engineering: A Good Practice Guide*. Chichester, England: Wiley. Retrieved from http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0471974447.html

Sowa, J. F. (2000). *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks/Cole.

Sowa, J. F., & Zachman, J. A. (1992). Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, *31*(3), 590–616.

SSA. (2000, Oktobar). Strukturna Sistemska Analiza. Univerzitet u Beogradu, Fakultet organizacionih nauka, Laboratorija za informacione sisteme. Retrieved from http://pisbp.fon.rs/download/SSA.pdf

Tarabanis, K. (2006). The Connection, Communication, Consolidation, Collaboration Interoperability Framework (C4IF) For Information Systems Interoperability. *IBIS – Interoperability in Business Information Systems*, *1*(1). Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.430.6288&rep=rep1&type=pdf

UMM (2011). *UN/CEFACT's Modeling Methodology (UMM), UMM Meta Model - Foundation Module Version 2*. Retrieved from http://www1.unece.org/cefact/platform/download/attachments/44204303/Specification_UMM_Foundation_Module_V2.0_TechnicalSpecification.pdf

Unhelkar, B., Ghanbary, A., & Younessi, H. (2010). *Collaborative business process engineering and global organizations: frameworks for service integration*. Hershey, PA: Business Science Reference.

Vernadat, F. (1996). *Enterprise Modeling and Integration* (1996 edition). London ; New York: Springer.

Vernadat, F. (1996). *Enterprise modeling and integration principles and applications*. London, UK: Chapman&Hall.

Vernadat, F. B. (2007). Interoperable enterprise systems: Principles, concepts, and methods. *Annual Reviews in Control*, *31*(1), 137–145. http://doi.org/10.1016/j.arcontrol.2007.03.004

Vlajić, S. (2005). Projektovanje softvera (skripta). Univerzitet u Beogradu, Fakultet Organizacionih Nauka, Institut za informacione sisteme.

Vujasinovic, M., Ivezic, N., Barkmeyer, E., & Marjanovic, Z. (2010). Semantic B2B-integration Using an Ontological Message Metamodel. *Concurrent Engineering*, *18*(3), 219–232. http://doi.org/10.1177/1063293X10379764

Vujasinovic, M., Ivezic, N., Kulvatunyou, B., Barkmeyer, E., Missikoff, M., Taglino, F., … Miletic, I. (2009). A semantic-mediation architecture for interoperable supply-chain applications. *International Journal of Computer Integrated Manufacturing*, *22*(6), 549–561. http://doi.org/10.1080/09511920802616781

Weske, M. (2007). *Business process management: concepts, languages, architectures*. Berlin ; New York: Springer.

Weske, M. (2012a). *Business Process Management*. Berlin, Heidelberg: Springer Berlin Heidelberg. Retrieved from http://link.springer.com/10.1007/978-3-642-28616-2

Weske, M. (2012b). Process Orchestrations. In *Business Process Management* (pp. 125–242). Springer. Retrieved from http://link.springer.com/10.1007/978-3-642-28616-2_4

Wiegers, K. E., & Beatty, J. (2013). *Software requirements* (Third edition). Redmond, Washington: Microsoft Press, s division of Microsoft Corporation.

Winter, R., & Fischer, R. (2006). Essential Layers, Artifacts, and Dependencies of Enterprise Architecture. In *2006 10th IEEE International Enterprise Distributed Object Computing Conference Workshops (EDOCW'06)* (pp. 30–30). http://doi.org/10.1109/EDOCW.2006.33

Wirfs-Brock, R. J. (1993). Designing scenarios: Making the case for a use case framework. *The Smalltalk Report*, *3*(3), 9–20.

Yourdon, E., & Argila, C. A. (1996). *Case Studies in Object-Oriented Analysis and Design with Cdrom* (1st ed.). Upper Saddle River, NJ, USA: Prentice Hall PTR.

Zachman, J. (2002). The zachman framework for enterprise architecture. *Zachman International*, 79.

Zachman, J. (2016). Zachman International - Enterprise Architecture. Retrieved June 4, 2016, from https://www.zachman.com/

Zachman, J. A. (1987). A Framework for Information Systems Architecture. *IBM Systems Journal, Zachman International, Enterprise Architecture*, *26*(3), 276–292.

Ziemann, J. (2010). *Architecture of Interoperable Information Systems: An Enterprise Model-Based Approach for Describing and Enacting Collaborative Business Processes*. Berlin, Germany: Logos Verlag Berlin GmbH.

Ziemann, J., Matheis, T., & Freiheit, J. (2007). Modelling of Cross-Organizational Business Processes - Current Methods and Standards. *Enterprise Modelling and Information Systems Architectures*, *2*(2).

Zimmermann, O., Krogdahl, P., & Gee, C. (2004, June 2). Elements of Service-Oriented Analysis and Design [CT316]. Retrieved May 29, 2016, from http://www.ibm.com/developerworks/library/ws-soad1/

BIOGRAFIJA AUTORA

Marija Janković je rođena 20.04.1978. u Beogradu. Prvu beogradsku gimnaziju, prirodno-matematički smer završila je u Beogradu kao đak generacije i nosilac Vukove diplome. Nakon završene gimnazije, 1997. godine je upisala Fakultet organizacionih nauka Univerziteta u Beogradu, smer Informacioni sistemi. Diplomirala je 2004. godine na smeru za Informacione sisteme sa prosečnom ocenom 9.07 (devet, 7/100), odbranivši diplomski rad na temu „Realizacija modula sprovođenje nastave u Oracle Java XML razvojnom okruženju“, sa ocenom 10 (deset). Na poslediplomske studije, na smer za informacione sisteme upisuje se 2004. godine. Školske 2006/2007. godine položila je sve planom predviđene ispite sa prosečnom ocenom 10 (deset). Oktobra 2007. godine odbranila je magistarski rad pod naslovom “Interoperabilnost modela poslovnih sistema”. Magistarski rad je 07.04.2008. godine nagrađen “Plaketom za objavljeni naučni rad” , Diskobolos, Društva za informatiku Srbije.

Radno iskustvo

- Od januara 2005. do aprila 2006. godine radila je u Centru za razvoj informacionih sistema Fakulteta Organizacionih Nauka, na poslovima razvoja i implementacije informacionog sistema za studentske servise.
- Od aprila 2006. do juna 2007. godine angažovana je kao saradnik na Fakultetu organizacionih nauka.
- Od juna 2007. zaposlena je na Fakultetu organizacionih nauka u zvanju asistent.

Nastavne aktivnosti

Od aprila 2006. do juna 2007. godine radila je kao saradnik u nastavi na Fakultetu organizacionih nauka na predmetima Baze podataka i Projektovanje informacionih sistema. Od juna 2007. godine radi kao asistent na Fakultetu organizacionih nauka

u Beogradu na predmetima osnovnih studija: Baze podataka, Projektovanje informacionih sistema, Analiza i logičko projektvanje IS i Fizički projekat IS u izabranom softverskom okruženju. Takođe, angažovana je i na predmetima diplomskih (master) studija modula Informacioni sistemi. Prilikom evaluacije od strane studenata njen pedagoški rad je redovno ocenjivan visokom ocenom.

Istraživačko iskustvo

Od septembra 2009. do oktobra 2010. godine radila je kao gost istraživač na National Institute of Standards and Technology (NIST), Manufacturing Engineering Laboratotry (MEL), Manufacturing System Integration Division (MSID), Enterprise Systems Group (EIS), Gaithersburg, MD, USA.

Pregled projekata

U periodu od juna 2006. godine do marta 2007. godine učestvovala je na FP-6 ATHENA (Advanced Technologies for Interoperability of Heterogeneous Enterprise Networks and their Application) istraživačkom projektu. Direktno je bila uključena u realizaciju pod projekta FP6-2004-IST-ATH-B5.10 Inventory Visibility Sub-Project: ATHENA Enterprise Modelling Area (A1) Results Validation.

Pregled objavljenih radova

Marija Janković je objavila više naučnih radova u časopisima međunarodnog i nacionalnog značaja, kao i u zbornicima sa domaćih i međunarodnih konferencija, u kojima je prezentovala rezultate rada na tezi.

Kategorija M14:

1. Janković M., Koković Z., Ljubičić V., Marjanović Z., Knothe T.: Enterprise Modeling Based Application Development for Interoperability Problem Solving, Monograph: K.Mertins, R. Ruggaber, K. Popplewell, X.Xu (Eds.):Enterprise Interoperability III: New Challenges and Industrial Approaches, Springer London, pp. 547-558, 2008 (ISBN 978-1-84800-220-3; DOI 10.1007/978-1-84800-221-0_43)

2. Janković M.,Marjanović Z., Knothe T., SnackP.: Using POP* as an Exchange Model: A Case Study, Monograph: P. Cunningham and M. Cunningham (Eds.): Expanding the Knowledge Economy: Issues, Applications, Case Studies, IOS Press, Netherlands, Volume 4, Part 1, pp. 162-169, 2007 (ISBN 978-1-58603-801-4)

3. Janković M., Ivezić N., Knothe T., Marjanović Z., Snack P.: A Case Study in Enterprise Modeling for Interoperable Cross-Enterprise Data Exchange, Monograph: R. J. Goncalves, J. P. Muller, K. Mertins and M. Zelm (Eds.): Enterprise Interoperability II: New Challenges and Approaches, Springer, London, pp. 541-552, 2007 (ISBN-978-1-84628-857-9; DOI 10.1007/978-1-84628-858-6_59)

Kategorija M23:

1. Janković M., Ljubičić M., Aničić N., Marjanović Z.: Enhancing BPMN 2.0 Informational Perspective to Support Interoperability for Cross-Organizational Business Processes, Computer Science and Information Systems, Special Issue on Collaborative e-Communities, Vol. 12, Number 3, pp. 1101-1120, 2015 (IF=0.477) (DOI: 10.2298/CSIS141112013J)

2. Michalopoulos D., Mavridis I., Janković M.: GARS: Real-Time System for Identification, Assessment and Control of Cyber Grooming Attacks, Computers&Security, Vol. 42, pp. 177-190, 2014 (IF=1.031)(doi:10.1016/j.cose.2013.12.004)

Kategorija M33:

1. Janković M., Ljubičić M., Aničić N., Marjanović Z.: Enhancing BPMN 2.0 Informational Perspective to Support Interoperability for Cross-Organizational Business Processes, Proceedings: ICIST 2014 4th International Conference on Information Society and Technology, Zdravković M., Trajanović, M., Konjović, Z. (Eds.), Society for Information Systems and Computer Networks, pp. 278-284, 2014, (ISBN: 978-86-85525-14-8)

Kategorija M53:

1. M. Janković, Z. Marjanović: „Interoperabilnost modela poslovnih sistema: EKanban studija slučaja“, časopis INFO M, str. 30-42, Volumen 24/2007, ISSN 1451-4397 UDC 659.25, Beograd, 2007.

Kategorija M63:

2. I. Novičić, M. Janković, N. Aničić, Z. Marjanović: "Realizacija programskog sistema “Studentski servisi” u Oracle Java XML razvojnom okruženju ", Infofest 2005. str., 219-225

Izjava o autorstvu

Potpisana: Marija Janković

Broj indeksa:

Izjavljujem

da je doktorska disertacija pod naslovom

SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI U METODOLOŠKIM PRISTUPIMA RAZVOJU IS

- rezultat sopstvenog istraživačkog rada,
- da disertacija u celini ni u delovima nije bila predložena za sticanje druge diplome prema studijskim programima drugih visokoškolskih ustanova,
- da su rezultati korektno navedeni i
- da nisam kršila autorska prava i koristila intelektualnu svojinu drugih lica.

U Beogradu,
10.06.2016. godine

Potpis autora

_____

Izjava o istovetnosti štampane i elektronske verzije doktorskog rada

Ime i prezime autora: Marija Janković

Broj indeksa:

Studijski program: Informacioni sistemi

Naslov rada: Specifikacija aspekata interoperabilnosti u metodološkim pristupima razvoju IS

Mentor: Prof. dr Zoran Marjanović

Izjavljujem da je štampana verzija mog doktorskog rada istovetna elektronskoj verziji koju sam predala radi pohranjenja u Digitalnom repozitorijumu Univerziteta u Beogradu.

Dozvoljavam da se objave moji lični podaci vezani za dobijanje akademskog zvanja doktora nauka, kao što su ime i prezime, godina i mesto rođenja i datum odbrane rada.

Ovi lični podaci mogu se objaviti na mrežnim stranicama digitalne biblioteke, u elektronskom katalogu i u publikacijama Univerziteta u Beogradu.

U Beogradu,
10.06.2016. godine

Potpis autora

_____

Izjava o korišćenju

Ovlašćujem Univerzitetsku biblioteku „Svetozar Marković“ da u Digitalni repozitorijum Univerziteta u Beogradu unese moju doktorsku disertaciju pod naslovom:

SPECIFIKACIJA ASPEKATA INTEROPERABILNOSTI U METODOLOŠKIM PRISTUPIMA RAZVOJU IS

koja je moje autorsko delo.

Disertaciju sa svim prilozima predala sam u elektronskom formatu pogodnom za trajno arhiviranje.

Moju doktorsku disertaciju pohranjenu u Digitalnom repozitorijumu Univerziteta u Beogradu i dostupnu u otvorenom pristupu mogu da koriste svi koji poštuju odredbe sadržane u odabranom tipu licence Kreativne zajednice (Creative Commons) za koju sam se odlučila.

1. Autorstvo

2. Autorstvo – nekomercijalno

3. Autorstvo – nekomercijalno – bez prerade

4. Autorstvo – nekomercijalno – deliti pod istim uslovima

5. Autorstvo – bez prerade

6. Autorstvo – deliti pod istim uslovima

U Beogradu,
10.06.2016. godine

Potpis autora

_____