Upravljanje Poslovnim procesima 2018/2019

1. OSNOVNI KONCEPTI

UPP je zasnovano na postavci da je svaki proizvod/usluga koju određena kompanija nudi tržištu **rezultat obavljanja niza aktivnosti**. Upravljanje aktivnostima u poslovnom procesu se najčešće **sprovodi** kroz neki **informacioni sistem**. Polazi se od pretpostavke da se aktivnosti realizuju u nekom organizovanom redosledu.

Organizaciju poslovnih procesa je neophodno uskladiti sa ljudskim potencijalima i drugim resursima kako bi se obezbedila veća efikasnost i konkurentnost.

Neophodno je omogućiti da dve različite grupe ljudi na isti način vide taj sistem :

- **Menadžeri** : Poboljšanje operativnih sposobnosti kompanije
- **IT sektor** : Formalne metode za opis poslovnih procesa i razvoj softverskih sistema za podršku UPP

Te dve grupe su gotovo uvek **disjunktne** i za opis aktivnosti i procesa koji se dešavaju u okviru firme nam treba **model koji će razumeti obe strane**.

Poslovni procesi su na **organizacionom nivou** neophodni za razumevanje načina funkcionisanja kompanija, ali i **IT sektoru** omogućavaju dizajn i realizaciju fleksibilnih i robusnih informacionih sistema (sposobni da se prilagođavaju novim zahtevima).

NAPOMENA : Nije sve poslovni proces niti se sve može naterati da bude poslovni proces.

Poslovni proces („uokvirena“ definicija) : Poslovni proces jeste proces koji treba da obavi neki **poslovni cilj** čiji je rezultat pozitivan i po onoga ko ga izvršava i po onoga ko ga inicira. Skup je aktivnosti i redosled je bitan. Aktivnosti se izvršavaju koordinisano, u odgovarajućem organizacionom i tehnološkom okruženju. Izvršava ga jedna organizacija, ali može imati interakcije sa poslovnim procesima drugih organizacija.

PRIMER : **Proces donošenja zakona**. Redosled aktivnosti i uloge u sistemu veoma precizne. Ljudi + softver. Kada se uvodi (novi) softver bitno je tehnološko okruženje, verovatno nešto već postoji (baze, papiri, HR…).

Osnova za **upravljanje poslovnim procesima** je **eksplicitna reprezentacija** poslovnog procesa sa jasno definisanim **akcijama** i **ograničenjima**. Sistem za upravljanje poslovnim procesima jeste generički softver koji obezbeđuje njegovo koordinirano izvršavanje na osnovu eksplicitne reprezentacije.

Eksplicitna reprezentacija može biti :

- Tekstualna : Nedostaje strukturiranost
- Formalna : Model poslovnog procesa, dijagrami

Model poslovnog procesa : Jeste skup aktivnosti i ograničenja za njihovo i izvršavanje. Model se ne može pokrenuti dok se ne odradi deploy i do tog momenta je samo formalna specifikacija.

Instanca poslovnog procesa : Jeste jedan konkretan slučaj izvršavanja modela poslovnog procesa i sastoji se od instanci aktivnosti.

Orkestracija procesa : isto što i upravljanje poslovnom procesom

Koreografija procesa : interakcija poslovnih procesa različitih organizacija
Dozvoljava različite interne implementacije poslovnih procesa.

Životni ciklus poslovnog procesa:

1. **Dizajn i analiza**:
   Sagledavanje poslovnih procesa, kao i organizacionog i tehnološkog okruženja. Obuhvata:
   a. Identifikaciju
   b. Reviziju
   c. Validaciju : Provera da li je model dobar. Provera usklađenosti sa realnim/željenim modelom
   d. Reprezentaciju odgovarajućim modelima.

   Ključna aktivnost je **modelovanje** i evolutivnog je karaktera. Eksplicitni modeli u grafičkoj notaciji su najbolji za komunikaciju među članovima tima. **Validacija** i **simulacija** igraju bitnu ulogu u ovoj fazi. Simulacija omogućuje da se poslovni proces prođe korak po korak i utvrde eventualne slabe tačke modela.

2. **Konfigurisanje**:
   Izbor **platforme za implementaciju**. Model poslovnog procesa se proširuje tehničkim detaljima neophodnim za izvršavanje u odabranom sistemu za upravljanje poslovnim procesima. Sistem za UPP je neophodno podesiti u skladu sa organizacionim okruženjem i povezati sa postojećim softverskim sistemima koji se koriste u poslovanju. Može zahtevati dodatni razvoj interfejsa.
   Nakon implementacije i podešavanja sledi testiranje i potom se sistem pušta u produkciju.

3. **Izvršavanje** :
   Stvarno izvršavanje poslovnog procesa. Pokreću se instance poslovnog procesa kako bi se postigli poslovno ciljevi. Sistem za upravljanje poslovnim procesima vrši kontrolu nad izvršavanjem poslovnog procesa na osnovu definisanog modela. Komponenta za nadzor sistema za upravljanje poslovnim procesima omogućuje uvid u stanje instanci poslovnog procesa tj. pregled aktivnosti, pokrenutih i završenih procesa. Prikupljaju se podaci u toku izvršavanja, obično u log fajlovima.

4. **Evaluacija** :
   Zaključuje se da li ovako modelovan proces obavlja ono što je predviđeno i prikupljaju se podaci za eventualni remodeling. Podaci se prikupljaju iz log fajlova ili process mining-om.

Tipovi učesnika u poslovnom procesu:

1. **Glavni menadžer** : Standardizacija i harmonizacija posl. procesa u organizaciji
2. **Poslovni inženjer** : Eksperti u određenoj poslovnoj oblasti, definisanje strateških i poslovnih ciljeva
3. **Dizajner poslovnih procesa** : Modelovanje poslovnih procesa
4. **Učesnici u izvršavanju** : Izvršavaju realne zadatke i imaju praktično znanje o načinu izvršavanja zadataka
5. **Obučeni radnici** : Učesnici u izvršavanju posl. procesa koji koriste soft. sistem za obavljanje zadataka
6. **Vlasnik poslovnog procesa** : Zadužen za efikasno i korektno izvršavanje posl. procesa
7. **Arhitekta sistema** : Zadužen za razvoj i konfigurisanje soft. sistema za upravljanje posl. procesima
8. **Programeri** : Razvoj softverskih komponenti neophodnih za implementaciju posl. procesa i razvoj interfejsa za povezivanje sa postojećim sistemima

Klasifikacija poslovnih procesa:

1. **Organizacioni / operativni** :

Različiti nivoi apstrakcije:

- Najviši nivo: Poslovni ciljevi i strategije
- Drugi nivo: Organizacioni poslovni procesi
- Treći nivo: Operativni poslovni procesi
- Implementacioni nivo

2. **Interni / koreografija** :
    - Interni : Primarni cilj pojednostavljenje i povećanje efikasnosti i procedura unutar organizacije
    - Koreografije : Podrazumevanju vezu sa drugim sistemima iz spoljnog sveta (komunikacioni i pravni aspekti)

3. **Stepen automatizacije** :
    - Potpuno automatizovani : Nije neophodan ljudski anganžman tokom izvršavanja, integracija enterprise aplikacija
    - Sistemi koji zahtevaju aktivnost korisnika
    - Potpuno manualni sistemi

4. **Stepen ponovljivosti** :
    - Često ponovljivi
    - Retko ponovljivi(isplati li se njihovo modelovanje i razvoj sistema za UPP?)

5. **Stepen struktiriranosti:**
   Ako se poslovni proces nikada ne menja, a koristi ga 2-3 čoveka, nakodiraj. U suprotnom formiraj model.
   - Model propisuje sve aktivnosti i uslove za njihovo izvršavanje
   - Produkcioni poslovni procesi : Dobro strukturirani i visoko ponovljivi
   - Adhoc aktivnosti

Ciljevi:

1. **Bolje razumevanje funkcionisanja organizacije** : Putem eksplicitnih reprezentacija poslovnih procesa
2. **Fleksibilnost** : Mogućnost lakše adaptacije na promene poslovnog procesa
3. **Evolutivni razvoj**
4. **Smanjenje jaza** između poslovnih procesa i softverskih sistema koji pružaju podršku za njihovo izvršavanje

2. EVOLUCIJA POSLOVNIH PROCESA

Poslovni sistemi se najčešće sastoje od različitih informacionih sistema i **osnovni princip jeste razdvajanje nadležnosti** (separation of concerns). Kod **sistema za upravljanje poslovnim procesima** glavni cilj jeste **prilagodljivost promenama**.

Neophodno je **identifikovanje sličnih i povezanih funkcionalnosti** i njihovo objedinjavanje u **podsisteme** sa jasno definisanom **namenom** i **interfejsom**. Ovo omogućava **ponovnu iskoristivost modula**. Postiže se i povećana **fleksibilnost** jer se moduli lakše modifikuju nego monolitna aplikacija i ako se interfejsi ne menjaju moguće je u potpunosti zameniti implementaciju modula. Lokalne promene (u modulima) ne izazivaju promene celog sistema (skrivanje informacija).

Softverska arhitektura definiše **strukturnu organizaciju softverskih komponenti** i **resursa** soft. sistema. Komponente i resursi se predstavljaju **podsistemima**. Podsistemi u datoj arhitekturi imaju jasno definisanu **namenu** i **međusobnu povezanost**. Detalji podsistema se ne definišu na nivou soft. arhitekture, već samo njihove veze i spolja vidljivo ponašanje.

Razvoj aplikacija:

U početku su se aplikacije razvijale „od nule“. Prenosivost i ponovna upotreba modula prethodno razvijanih aplikacija bila je veoma mala jer je kod pisan isključivo za **specifične platforme**.

Pojavom **operativnih sistema** (apstrahuje hardversku platformu) omogućeno se pisanje aplikacija za operativni sistem, ne samo za hardver tj. raste prenosivost. I dalje postoji problem sa formatima podataka koje aplikacije koriste tj. svaka aplikacija poseduje sopstveni sistem za trajno čuvanje podataka (često se javljala redundansa).

Sistemi za upravljanje bazama podataka (DBMS) rešava ovaj problem i obezbeđuje nezavisnost fizičkog smeštanja i logičke organizacije podataka od same aplikacije.

Pojava **grafičkog korisničkog interfejsa** (GUI) je uveo standardizaciju u upravljanju programom. Drastično skraćeno vreme obuke, više nije neophodno zadavati precizne naredbe putem terminala i veliki broj ljudi postaje sposobno da korisni računar. Razvoj GUI-a kao posebnog podsistema omogućuje da se interfejs razvija modularno i da bude iskoristiv u različitim aplikacijama.

Često je u okviru jedne kompanije postojalo više aplikacija koje su imale različitu ulogu (evidencija zaposlenih, nabavka, prodaja itd.). Svaka od aplikacija je imala sopstveni DBMS u kom su se često nalazile iste informacija kao i u DBMS druge aplikacije. Podaci su često bili povezani i postojali su problemi sa ažuriranjem.

Enterprise resource planning (ERP) sistemi su razvijani tako da budu **sveobuhvatni sistemi** i da objedine prethodno fragmentirane podatke. Objedinjavanje sistema izvršeno je nad integrisanom i konzistentnom bazom podataka. Moduli ERP sistema jesu čisto serverske aplikacije kojima se obraćaju udaljeni klijenti i koje komuniciraju sa jednom bazom podataka nad kojom imaju određena prava pristupa. Garantuje se konzistentnost.

Heterogenost podataka : Isti podaci različito su strukturirani i definisani u različitim aplikacijama, različita semantika atributa.

PRIMER : Povezivanje artikala i njihovih cena iz dva preduzeća. Problem u formatu zapisa, borju decimala, cena obračunata sa ili bez PDV itd.

Integracija podataka : Uvodi se poseban srednji sloj

1. **Point-to-point** (tačkasta integracija) :
   Direktno povezivanje svih parova aplikacija. Problem : potrebno nam je **N^2 interfejsa**! Promene aplikacija dovode do potrebe promene i interfejsa. Može se poboljšati upotrebom **messaging servisa**. Srednji sloj bi tako bio zadužen samo za kontrolu i garanciju isporuke poruka i sadržao bi samo informacije kome je poruka namenjena i kome treba poslati odgovor. U osnovi je i dalje point-to-point komunikacija.

2. **Hub-and-spoke** :
   Bazira se na centralnom čvorištu (hub-u) i aplikacijama koje su na njega spojene (spokes) **bez direktne veze između aplikacija**. I ovaj princip je zasnovan na slanju i primanju poruka, ali se ne čuva eksplicitno ko je primalac. Broj veza je redukovan na **N** i hub na osnovu sadržaja i strukture poruke zaključuje kome je poruka namenjena. Veza postojećih aplikacija sa hub-om se ostvaruje putem **adaptera**. Razvoj adaptera može biti zahtevan posao.
   Srednji sloj se može realizovati pomoću **message broker-a**. Tako korisnik može **da definiše pravila i format komunikacije**. Implementacija adaptera se tada svodi na definisanje odgovarajućih pravila. Poboljšana je prilagodljivost promenama strukture sistema. Obavlja se i **mapiranje podataka**.
   Koristi se **publish/subscribe mehanizam**. Slično kao observer/observable tj. jedan modul objavljuje, a drugi moduli se pretplaćuju na primanje poruka.

Upravljanje radnim procesima (workflow menagement):

1. **Vrednosni lanci** :
   Prikaz poslovnih funkcija visokog nivoa. Omogućuje uvid u to kako kompanija funkcioniše. Opisuje globalni tok dobara i informacija. Ulaz su dobavljači, a izlaz oni koji koriste proizvode kompanije.
   Vrednosni sistem : Saradnja više firmi i veza njihovih vrednosnih lanaca.

2. **Organizacioni poslovni procesi** :
   Opisuje poslovne procese i komunikaciju između njih na visokom nivou. Sam poslovni proces je „crna kutija". Govore koji delovi firme moraju obaviti nešto i šta moraju obaviti da bi se vrednosni lanac izvršio. Mogu se detaljnije opisati grafovima kolaboracije procesa.

Radni proces : Automatizacija poslovnog procesa, tokom kog se dokumenti, informacije i zadaci prosledjuju od jednog do drugog učesnika, u skladu sa proceduralnim pravilima.

Sistem za upravljanje radnim procesima : Softverski sistem koji definiše, kreira i upravlja izvršavanjem radnog procesa korišćenjem odgovarajućeg softverskog alata koji je sposoban da interpretira definiciju procesa, obezbedi interakciju sa učesnicima i obezbedi korišćenje IT alata i aplikacija.

Single application workflow : Aktivnosti i njihov uzročni i vremenski redosled koje se izvršavaju u okviru jedne aplikacije.

Multiple application workflows : Aktivnosi koje se ralizuju kroz različite aplikacije, pri čemu je realizovana integracija ovakvih sistema.

Vrste workflow sistema :

1. **Sistemski radni procesi** : Sastoje se samo iz aktivnosti koje se izvršavaju automatski od strane softerskog sistema, bez ikakvog učešća korisnika.
2. **Interaktivni radni procesi** : Zahtevaju interakciju sa korisnikom i informacionim sistemima. Mora se uzeti u obzir struktura organizacije u kojoj se proces izvršava. Neophodno je definisati korisničke uloge tj. role kako bi se za svaku aktivnost moglo definisati koji je korisnici mogu izvršiti.

Problemi i izazovi :

1. Ovi sistemi značajno menjaju način obavljanje posla.
2. Predstavljanju procese, ali i zaposlene u kompaniji. Pitanje raspodele zadataka može biti problematično.
3. Ekspertski korisnici se mogu osećati sputano tj. obezbedi fleksibilnost aktivnosti tako da se ne naruši tok procesa.
4. Aplikativne sisteme treba integrisati sa workflow sistemom
5. Deo poslovne logike koji je implementiran na nivou programskog koda teško preslikati u procesni model
6. Granularnost aktivnosti u procesu i funkcije poslovnih aplikacija se ne moraju poklapati

3. MODELIRANJE POSLOVNIH PROCESA

Koncepti apstrakcije : Omogućuju nam da sagledamo i modelujemo podatke samo u pogledu konteksta informacionog sistema i sve ostalo odbacimo.

To su :

1. **Horizontalna** :
   a. Nivo **meta-meta modela**
   b. Nivo **metamodela** : Opisuju same modele (npr. grafičke notacije) tj. specifikacija se jezika za modelovanje.
   c. Nivo **modela** : Opisuju i klasifikuju slične entitete sa nivoa instanci u model.
   d. Nivo **instance** : Opis konkretnih entiteta.

2. **Vertikalna** : Identifikuju se specifični poddomeni koje treba modelovati.
   a. **Modelovanje funkcija** : Modelovanje **jedinica rada** koje se izvršavaju tokom poslovnog procesa tj. modelovanje **aktivnosti**. Najčešće prvo tekstualni opis pa potom precizna specifikacija neophodna za implementaciju funkcija u informacionom sistemu.

   b. **Modelovanje informacija** : Obrađuje problem reprezentacije podataka u poslovnom procesu. Izvršavanje poslovnog procesa može menjati postojeće ili stvarati nove podatke. Izvršavanje posl. procesa može zavisiti od određenih podataka. Međuzavisnost aktivnosti se mora uzeti u obzir tj. u određenom momentu određeni podaci moraju biti dostupni i usklađeni.

   c. **Modelovanje organizacije** : Reprezentacija (odgovarajuće) strukture poslovnog sistema (organizacionih celina, uloga zaposlenih itd.).

   d. Modelovanje IT okruženja : Sagledavanje informacionog okruženja u kom se izvršava poslovni proces. Ustanoviti gde su nam interfejsne tačke i u skladu sa tim modeluj poslovni proces.

3. **Agregaciona**

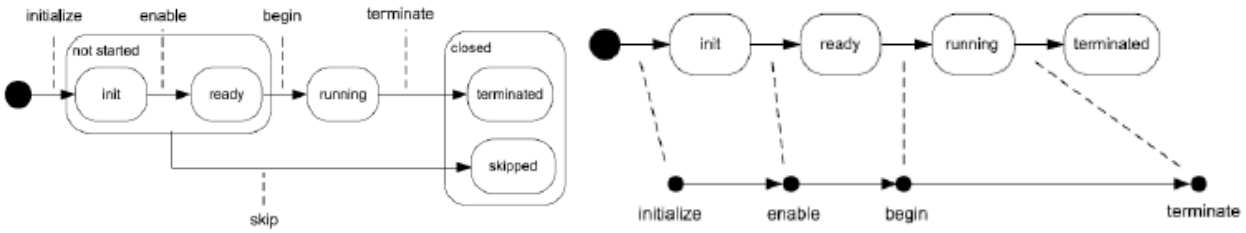
Funkcionalna dekompozicija : Kako doći od složenog problema do pojedinačnih aktivnosti.

Funkcionalna perspektiva : Jeste funkcionalna dekompozicija poslovnih funkcija visokog nivoa na detaljnije funkcije niskog nivoa.

Model aktivnosti : Kako se opisuje aktivnost u nekom poslovnom procesu i opisuje se skup sličnih aktivnosti. Opis tekstualan ili formalan. Template je i stvarima poput ko aktivnosti izvršava i koje podatke aktivnost koristi ovde nije mesto.

Instanca aktivnosti : Predstavlja stvarni rad koji se izvršava tokom obavljanja poslovnog procesa.

Životni ciklus instance aktivnosti : Jeste **uređen skup događaja**. Opisuje se događajima (event-ovima) koji se mogu desiti i u kom redosledu. Događaj se dešava u određenom trenutku tj. ne traje. Svaki **prelaz stanja** aktivnosti se predstavlja kao jedan događaj. Vreme koje instanca aktivnosti provodi u određenom stanju definisano je sa dva događaja : **događaj ulaska** u trenutno stanje i **događaj izlaska** iz trenutnog stanja.



Dijagram prelaska stanja i događaja za izvršenu aktivnost

Osnovni koncepti metamodela procesa :

1. **Model procesa** : Šablon za kreiranje instanci procesa. Sastoji se od skupa modela aktivnosti. Sastoji se od čvorova i usmerenih grana. Mota imati barem 2 čvora i 1 granu kako bi bio suštinski validan!
2. **Grana (ivica)** : Predstavlja veze između čvorova.
3. **Čvor** : Može predstavljati model aktivnosti, model događaja, model gateway-a.

Instanca procesa : Sastoji se od jedne ili više instanci aktivnosti, ima svoj životni ciklus, izvršavaju se određeno vreme i terminitaju (uspešno ili neuspešno). Redosled obavljanja aktivnosti je uslovljen modelom.

Interakcija procesa : Interakcija između poslovnih entiteta gde učesnici u pojedinim procesima komuniciranju jedino razmenom poruke. Specificira se kako se 2 ili više procesa mogu povezati kako bi se postigao određeni efekat tj. jeste specifikacija protokola.

Modelovanje podataka :

Osnova je za **integraciju heterogenih podataka** koji dolaze **iz različitih informacionih sistema**. Obezbediti **mapiranje** između tipova podataka. Problem atributa koji se ne mogu mapirati ili koji se moraju mapirati na različite tabele kompleksnim pravilima. Problem **semantičke heterogenosti** (kada ne postoji eksplicitan opis značenja pojedinih podataka).

Prenos podataka između aktivnosti jeste **tok podataka** i međuzavisnost podataka između tih aktivnosti se reprezentuje upravo tokom podataka. Moguća vizuelna reprezentacija grafičkom notacijom.

Šabloni podataka u radnim procesima : Opisuju kako upravljati podacima u poslovnim procesima.

1. **Vidljivosti podataka** : Podaci vidljivi u različitim scope-ovima (na nivou aktivnosti, podprocesa, procesa ili okruženja)
2. **Interakciji podataka** : Kako se podaci mogu prosleđivati između aktivnosti i procesa
3. **Prenosu podataka** : Šabloni po kom se podaci mogu prenositi (po vrednosti ili referenci)
4. **Rutiranju na osnovu podataka** : Vrednost podataka može imati implikacija na izvršavanje procesa tj. postojanje odr. podataka može **omogućiti izvršavanje pojedinih aktivnosti** ili se koriste za **evaluaciju uslova**

Modelovanje organizacije

Sistem za upravljanje poslovnim procesima mora imati **informaciju o organizacionoj strukturi** u okruženju u kom se poslovni proces izvršava. Cilj je iskoordinisati ko će, u kom momentu i pod kojim uslovima dobiti mogućnost nešto da obavi. Neophodno je **opisati sve resurse** (ljudske, fizičke ili servisne) koji su neophodni da se neki proces izvrši tj. **osnovni princip organizacionog modelovanja jeste resurs**.

Dodela zadataka

Njom se ostvaruje veza između organizacione strukture i poslovnog procesa. Radni zadaci predstavljaju instance aktivnosti koje su dodeljene određenim zaposlenim na obavljanje. Dodela može biti :

1. **Direktna** : Tačno određenoj osobi. Može biti pretvrda. Može biti i posredna ako se dinamički dodeljuje u vreme izvršavanja.
2. **Role-based** : Mora se razrešiti ko od pripadnika role može rešiti zadatak. 2 pristupa, ili izvršioci u određenoj ulozi sami preuzimaju zadatak (kooperativni princip) ili se po nekom algoritmu zadatak dodeljuje jednom izvršiocu u datoj ulozi.
3. **Odložena dodela** : Odluka o tome ko zadatak obavlja se donosi u vreme izvršavanja, ali ne na osnovu uloge.
4. **Autorizacija** : Dodela izvršiocu zasnovana na osnovu njegove pozicije (tj. kreiranjem dozvole da se nešto obavi).
5. **Razdvajanje nadležnosti**: Omogućuje garanciju dodele zadataka različitim učesnicima (čak i u istim ulogama). **PRIMER** : 5 članova komisije treba potpisati nešto, ali nikako 1 član 5 puta.
6. **Obrada predmeta (Case handling)**: Koristi se kada je neophodno da izvršilac bude upoznat sa samim slučajem. Zadaci se tada uvek dodeljuju istoj osobi (u određenoj ulozi). **PRIMER:** Parnica na sudu, nećemo svaki put birati advokata/sudiju kada je isti slučaj u pitanju.
7. **Na osnovu prethodne istorije** : Na osnovu evidencije šta je određena osoba prethodno obavljala. **PRIMER:** Jedna osoba uvozi automobile druga lekove, iako obojica uvoze daj zadatak prema tome šta ko inače radi.
8. **Organizaciona dodela** : Koriste se stvarne pozicije korisnika u org. strukturi firme. Omogućuje složene algoritme alokacije.

Modelovanje operacija

Aktivnosti se razlikuju na nivou softverske podrške za njihovo izvršavanje. Modeluju se aktivnosti koje se realizuju pomoću softverskih funkcionalnosti. Uzimaju se u obzir tehnički detalji o tome kako se podešava process engine.

Fleksibilnosti sistema za upravljanje poslovnim procesima :

1. **Eksplicitna reprezentacija procesa** : Omogućuje da relativno jednostavno „presložimo“ proces a da ne moramo ponovo da kodiramo.
2. **Organizaciono modelovanje** : Fleksibilnost u pogledu dodele zadataka i razrešavanja uloga (koji korisnici u datoj ulozi mogu izvršiti zadatak, obavlja se u vreme izvršavanja).
3. **Standardizacija softverskih interfejsa** : Ključni za ostvarivanje komunikacije sa drugim informacionim sistemima.

4. ORKESTRACIJA PROCESA

Šabloni za kontrolu toka : Osnovni alat za izražavanje strukture tj. orkestraciju procesa. Nezavisni su od konkretnog jezika za opis procesa. Semantiku šablona posmatramo u smislu kako on utiče na redosled izvršavanja događaja.

Proces jeste uređena N-torka : **P = (N, E, *type*)** gde su :

- **$N = N_A \cup N_E \cup N_G$** tj. skupovi modela aktivnosti, događaja i grananja
- **E** tj. skup grana
- ***type*** **:** **$N_G \rightarrow C$** tj. preslikavanje koje svakom modelu grananja pridružuje odredjenu kontrolu toka

NAPOMENA: Model aktivnpsti označavamo velikim slovima (A, B…), njihove instance malim (a, b…), ukoliko ima više instanci dodaju se indeksi (a_1, a_2…)

Osnovni šabloni za kontrolu toka : Podržavaju ih svi jezici za modelovanje procesa.

1. **Sekvenca** : Definiše redosled događaja između **a** i **b** takav da se *enable* događaj instance **b** može desiti samo ako se desio *terminate* događaj instance **a**. Petlja ne narušava suštinu šablona, uslovljen je redosledom iz modela.

2. **I grananje** : Omogućuje da se jedan tok izvršavanja multiplicira u različite konkurentno izvršive tokove. Tj. posle *terminate* događaja **a** moguće je izvršiti i **b** i **c** i **d**. I grananje čeka da se sve forkovane grane završe i mora se desiti **I spajanje** tj. neophodno je izvršiti sinhronizaciju.

3. **I spajanje** : Tačka u kojoj se svi konkurentno izvršivi tokovi spajaju u jedan. Tj. svakom *enable* događaju **e** moraju prethoditi *terminate* događaji instanci i **b** i **c** i **d**.

4. Ekskluzivno ILI grananje (XOR split) : Tačka u procesu gde se bira jedna od mogućih putanja izvršavanja. Tj. posle svakog *terminate* događaja instance **a** desiće se *enable* događaj ili **b** ili **c** ili **d**, ali samo jednog od navedenih. Moramo imati **uslov odlučivanja**, ili je on rezultat izvršavanja neke (Java) klase. Ukoliko su uslovi pogrešni, mora postojati neka ***default* grana**. Ona je ili proglašena za podrazumevanu ili se prva navedena u modelu smatra podrazumevanom.

5. Ekskluzivno ILI spajanje (XOR join) : *Terminate* **b** ili **c** ili **d** instanca e dobija *enable* događaj.

Ostali šabloni za kotrolu toka :

1. **ILI grananje** : Bira se barem **jedna od mogućih** grana za izvršavanje. Izbor bilo koje neprazne grane je validan. Ekstremi su XOR ili čisti I. Posle svakog *terminate* događaja instance a sledi (neprazan) podskup *enable* događaja **b** i **c** i **d**.
   Problem kod spajanja : **Moramo znati koliko grana čeka i kojih**. Treba nam mehanizam za praćenje izvršavanja. Npr : roditelj svojoj deci podeli tokene i zapamti koliko je pušteno ili upamti listu otvorenih čvorova i terminiraj kada je prazna.

2. **ILI spajanje** : Tačka u procesu u kojoj se različite putanje izvršavanja spajaju. Potom se vrši sinhronizacija. Pretpostavka : Jedna grana izvršavanja koja je bila aktivirana ne može se ponovo aktivirati dok spajanje čeka na završetak ostalih aktiviranih grana. Ponašanje može biti problematično bez dodatnog znanja o samoj prirodi procesa.

3. **Višestruko spajanje** : Predstavlja tačku u procesu gde se dve ili više tačaka spajaju bez sinhronizacije. Prema tome, aktivnost koja sledi će biti aktivirana onoliko puta koliko je bilo aktiviranih dolaznih grana.
   Funkcionalno je ekvivalentno XOR spajanju, samo bez pretpostavke da je aktivirana samo jedna grana. Zato greškom može nastati I grananje bez I spajanja. Ako se to desi, formalno nismo u stanju da ustanovimo kada se proces završio.

4. **Diskriminator** : Čeka se da se **izvrši jedna od ulaznih grana** kako bi se aktivirala naredna aktivnost. Nakon što je naredna aktivnost aktivirana, diskriminator čeka da se i ostale aktivirane grane završe, ali **ne aktivira ponovo narednu aktivnost**. Tek kada se sve aktivirane grane završe biće moguće ponovno „okidanje“. Zgodno koristiti u kontekstu petlji kako ne bi bilo konfuzije između aktivnosti obavljenih u drugom ciklusu i okasnelih aktivnosti iz prvog ciklusa obrade.
   Prema tome, radi po principu ko prvi obavi dobija „nešto“ tj. koji god odgovor dobijam nastavljam dalje. Obično prvi aktivira se uzima, ostalo u logove. Princip „1 od N“.

5. **N od M spajanje** : Generalizacija diskriminatora. Tačka u kojoj se M paralelnih grana spaja u jednu. Čeka da se **izvrši N ≤ M ulaznih grana** pre nego što se aktivira naredna aktivnost. Isto kao i kod diskiminatora, kada se naredna aktivnost jednom aktivira ostalih M-N grana se ne terminira, ali je ponovno „okidanje“ moguće tek kad se svih M obavi.

Ekstremi : **N = M** onda je **I spajanje**, **N = 1** onda je **diskriminator**.
PRIMER : Tender, kada primim N od M ponuda idem dalje ali ne znači da prestajem da primam ponude.

6. **Proizvoljni ciklus** : Mesto u procesu u kom se jedna ili više aktivnosti može ponavljati tj. kombinacija svega prethodnog.

7. **Implicitna terminacija** : Instanca procesa treba da se **terminira kada više nema šta da se obavi**. To znači da nema ni jedne instance aktivnosti koja je u stanju *init* ili *redy*, niti aktivnosti koja je u stanju *running* i prema tome ne postoji ni jedna aktivnost koja bi mogla postati omogućena. Kada nema šta da se obavi, proces treba to da „shvati“ sam i ode u END.

8. **Višestruke instance**: Kreiranje više instanci određene aktivnosti, među kojima nije neophodna sinhronizacija. Ovaj šablon narušava sekvencijalnost i može izazvati terminaciju procesa. Vrste se razlikuju prema momentu u kom se utvrđuje broj multiplikacija instanci.
    a. **Poznat u momentu dizajna modela** : Instance aktivnosti sinhronizovane, sledeća aktivnost moguća tak kad su sve multiplikovane instance završene. Nove instance mogu biti kreirane dok se postojeće izvršavaju, ali se naredna aktivnost okida tak kada se sve aktivirane završe
    b. **Nije poznat u momentu dizajna modela** : Broj potrebnih instanci nije moguće znati pre nego što se multiplicirana aktivnost omogući.

9. **Odloženo odlučivanje** : Šablon baziran na **konceptu stanja**. Opisuju implicitno ponašanje procesa, ne na trenutnom slučaju koji se obrađuje, već je **uslovljeno okruženjem** ili drugim procesima. Obično postoji neki drugi proces koji predstavlja „okruženje“.
   Predstavlja mesto u procesu gde se **bira jedna od mogućih grana**. Izbor nije eksplicitan već okruženje nudi nekoliko alternativa. **Okruženje aktivira jednu od alternativnih putanja**. Odlučivanje se odlaže do poslednjeg mogućeg momenta, tj. sve dok jedna grana nije aktivirana.
   PRIMER: IOT senzori. Provala, koji god senzor prvi okine on šalje poruku policiji.

10. **Sekvencijalno izvršavanje bez prethodnog znanja**: Napravimo model koji čak nije ni konačan. Skup aktivnosti se izvršava sekvencijalno, ali se redosled utvrđuje u vreme izvršavanja. Praviti **n!** mogućih kombinacija za **n** aktivnosti nije praktično.

11. **Milestone** : Određenu aktivnost je moguće omogućiti samo kada je dostignuta određena referentna tačka u modelu (koja je pritom još važeća).

12. **Runtime šabloni** : Nisu deo vidljivog modela već karakterišu sam process engine.
    a. **Cancel activity pattern** : Da li dati process engine započete aktivnosti može otkazati?
    b. **Cancel case** : Zaustavljanje instance procesa ma gde da se u njemu nalazimo.

Petri mreže:

Jedna od najpoznatijih tehnika za formalnu i apstraktnu specifikaciju poslovnih procesa. Zasnovana na teoriji skupova, nije puno ekspresivan ali može biti formalno validiran. Uopštavanje je teorije automata uvođenjem pojma konkurentnosti. **Razlozi za primenu** :

1. Formalna semantika
2. Grafička reprezentacija
3. Analiza osobina procesa
4. Nezavisnost od izabranog alata za modelovanje

Osnovni koncepti :

1. **Mesta** (*places*) : Predstavljaju se krugovima. Trenutna su stanja u kojima se sistem nalazi.
2. **Prelazi** (*transitions*) : Predstavljaju se kvadratima. Vrše prelaz u novo stanje. Kasnije evoluirale u aktivnosti.
3. **Veze** (usmereni lukovi, *directed arcs*) : Predstavljaju se linijama i strelicama i moguće su isključivo između mesta i prelaza (i obrnuto).

Dinamika sistema se predstavlja se konceptom **tokena**. Struktura Petri mreže je fiksna, a tokeni se mogu nalaziti u različitim mestima (nikako u tranziciji) u mreži. Struktura sa određenim rasporedom tokena predstavlja **stanje mreže**.

Prelaz je omogućen kada se **u svim njegovim ulaznim tačkama nalazi bar po jedan token**. Kada se on desi, iz svakog ulaznog čvora se uklanja po jedan token i u svaki izlazni se postavlja po jedan. Ovo kretanje tokena se naziva **igra tokena**.

Petri mreža jeste uređena N-torka (P, T, F). Stanje mreže se može opisati kao binarni vektor. Pošto može istovremeno postojati više instanci procesa, tokeni u jednoj Petri mreži ne moraju pripadati istoj instanci procesa. Klase perti mreža :

1. **Mreže uslovnih događaja** : U svakom momentu svako mesto može da sadrži samo 1 token. Tokeni su nestrukturirani tj. nemaju identitet i ne mogu se razlikovati. Ako se u nekom mesti nalazi token, uslov za prelaz je zadovoljen i kada se okine tranzicija desi se događaj i stanje mreže se promeni.

2. **Mreže tranzicije mesta** : Proširenje osnovne klase. U jednom mesti mreže može se nalaziti proizvoljan broj tokena. Uvodi se pojam težine grane, tj. tranzicija je omogućena kada se u ulaznom stanju nalazi minimum onoliko tokena kolika je težina grane. Po okidanju tranzicije, iz ulazne grane se uklanja onoliko tokena kolika je težina. Mesta mogu služiti kao brojači.

3. **Kolor Petri mreže** : Uvodi se koncept boje, tj. imaju vrednost i mogu se identifikovati. Tokeni imaju tipizirane vrednosti, zna se kom domenu pripadaju i koje operacije su za njih validne. Možemo identifikovati koliko i kakvih tokena u nekom mestu imamo i na osnovu toga postavljati uslove i vršiti prelaze.

Event driven process chains :

Dosta **neformalna** notacija. Više okrenuta ka predstavljanju specifičnih **domenskih koncepata** i procesa nego na **formalnu tehničku specifikaciju**. Deo *ARIS-a* (Architecture of Integrated Information Systems) tj. model koji kaže šta ćemo sve imati kao „stubove“ na kojima gradimo sistem. Formiraju se i dodatni dijagrami koji bliže specificiraju određene aspekte.

1. Dijagram interakcija : Obezbeđuje informaciju o organizacionim celinama koje učestvuju u procesu.
2. Dijagram toka funkcija : Obezbeđuje uvid u funkcije koje se obavljaju prilikom interakcija.

Workflow mreže :

Petri mreže sa 3 dodatna uslova :

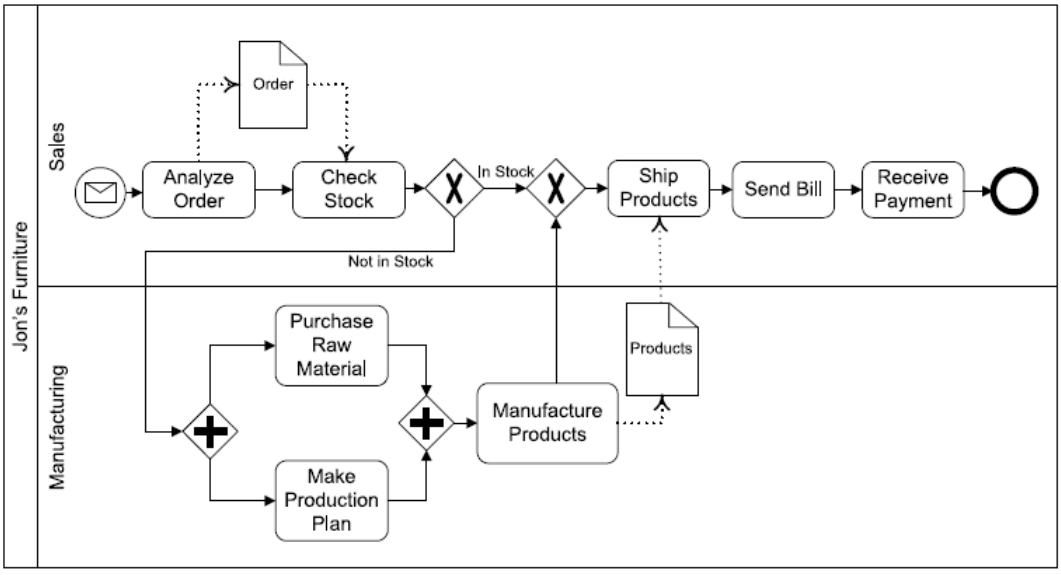
1. Postoji određeno **polazno mesto** koje nema ulaznih grana (*i*)
2. Postoji određeno **odredište** koje nema izlaznih grana (*o*)
3. Sva ostala mesta i tranzicije se nalaze između *i* i *o.*

Triggeri : Uvedeni kako bi se detaljnije opisali uslovi i okruženje u kom se procesi izvršavaju. Prikazuju se kao anotacije kojim se obezbeđuju dodatne informacije o tome ko ili šta je odgovorno za okidanje tranzicije koja je omogućena.

BPMN:

Definiše i **notaciju** i **metamodel** za predstavljanje modela poslovnih procesa. Osnovni elementi omogućuju izražavanje jednostavnijih struktura. Grafička notacija se dopunjuje skupom atributa. Organizacioni aspekti se predstavljaju pomoću *pool* i *swimlane* koncepata. **Grupe elemenata notacije** :

1. **Flow objects** : Elementi kojima se gradi model.
2. **Artefacts** : Dodatne informacije koje nisu relevantne za tok izvršavanja.
3. **Connecting objects** : Povezuju elemente dijagrama.
4. **Swimlanes** : Prikazuju organizacioni aspekt tj. uloge korisnika.



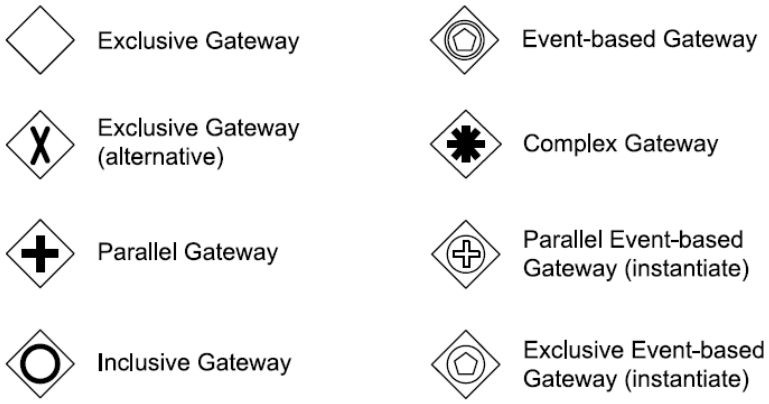
Primer BPMN dijagrama

Aktivnosti : Osnovni element poslovnog procesa. Jesu jedinice posla koje treba obaviti u procesu. Postoje različiti tipovi (user task, service task, send/recive task, script task…).

Događaji : Spona između realnog sveta i procesa koji će reagovati na pojavu određenog događaja. Događaji su ili catching ili throwing tipa i mogu biti i granični.

Throwing eventi nisu **blokirajući**, *catching* jesu! *Boundary* event služi za **exception handling** tj. ako se desi izuzetak čupaj proces kako se ne bi zaglavio jer već nešto nije u redu. Tako se model brani od neregularnih situacija. Postoje različiti tipovi događaja (message, timer, error, signal, termnate…).

Grananja :



Tipovi grananja

1. Exclusive gateway : XOR šablon
2. Parallel gateway : AND šablon
3. Event based gateway: Šablon odloženog izbora
4. Inclusive gateway : Ujedinjuje XOR i AND grananje
5. Complex gateway : Omogućuje da se definiše kombinovano složeno ponašanje, split i join

Instanciranje procesa : Često su potrebni određeni **preduslovi da bi se instanca procesa pokrenula**. BPMN omogućuje da se pri modelovanju specificira da je moguće da se pojavi :

1. Više alternativnih start događaja
2. Više neophodnih start događaja

Izvršivost i eksport formati : Ranije nije bilo moguće direktno izvršavati BPMN već se morao prvo prevesti u WS-BPEL. Sada je omogućena izvršivost modela.

5. KOREOGRAFIJA PROCESA

Koreografije procesa : Obezbeđuju **interoperabilnost orkestracija procesa** koje izvršavaju određeni učesnici u B2B (business to business) saradnji. Neophodno je **definisati pravila** koja učesnici moraju poštovati kako se ne bi stalno morali **ugovarati modeli saradnje** tj. pravila se definišu koreografijama. Postoje inicijative da se koreografije standardizuju u određenim domenima, ali joj **nedostaje opšta primenljivost**:

1. RodettaNet za nabavke
2. SWIFT za finansijske usluge
3. HL7 za zdravstvene usluge

Zahtevi koji se postavljaju za razvoj koreografije procesa zavisi od **broja partnera** i **nivoa automatizacije**. B2B saradnja je od presudnog značaja za uspešno funkcionisanje kompanija i kompanije moraju da sarađuju kako bi ponudile nove proizvode/usluge. Prema tome, dizajniranje sistema koji obezbeđuje interakciju mora biti pažljivo odrađeno, kako bi informacioni sistemi zainteresovanih strana mogli razmenjivati poruke i kako ne bi došlo do deadlock-a. **PRIMER** : Online aukcija (automobila).

Izolovanje samo jednog procesa nije dovoljno za korektno izvršavanje interakcije procesa tj. neophodno je modelovati i interakciju. Koriste se nivoi apstrakcije : metamodela, modela i instance. Instanca modela koreografije procesa jeste **konverzacija procesa**.

Koreografija procesa se sastoji iz više modela interakcije. Svaki od njih je povezan sa 2 modela komunikacionih aktivnosti(pruža mogućnost slanja i primanja poruka).

Faze razvoja : grubo, **dizajn koreografije** i **implementacija koreografije**. Neophodno je **izvršiti identifikaciju učesnika** (organizacija) u koreografiji, **definiciju kritičnih tačaka** (milestones) i **identifikaciju poruka**. Milestones jesu stanja koreografije u kojima su saradnjom učesnika postignuti rezultati. Može se predstaviti BPMN notacijom gde su **milestone** predstavljeni kao **događaji**. Ukoliko se neki milestone ne dostigne, moguće je da i cela koreografija **ne obavi zadatak** zbog kog je kreirana.

Behavioural aspect : Aspekt ponašanja koreografije, određen kritičnim tačkama i njihovim redosledom. Definišu interfejse za svaku od uloga koja učestvuje u koreografiji. Služe kao osnova za orkestraciju procesa koje određeni učesnik realizuje.

Dizajn koreografije procesa

1. **Dizajn strukture na visokom nivou** : Identifikuju se uloge učesnika i struktura komunikacije.
2. **Dizajn ponašanja na visokom nivou** : Definišu se kritične tačke i redosled njihovog izvršavanja.
3. **Scenarija saradnje** : Povezuju dostizanje određene kritične tačke sa ostvarenim komunikacijama između učesnika.
4. **Behavioural Interfaces** : Interfejs saradnje svakog učesnika koji se izvodi na osnovu scenarija saradnje.

Scenario saradnje : Razmatraju tek kada postoje neke kritične tačke u koreografiji.

- Specificira se **interakcija** koja je neophodna **između učesnika** kako bi koreografija napredovala od jedne tačke do druge.
- Interakcije koje se dešavaju između dve kritične tačke mogu biti opisane sa jednim ili više scenarija saradnje.
- Pojedinačni scenario treba biti jednostavan.

Kompatibilnost : Sposobnost skupa učesnika da uspešno sarađuju u skladu sa datom koreografijom. Može biti **strukturna** i **kompatibilnost ponašanja**. Mogući uzroci **nekompatibilnosti** :

1. Različiti formati poruka
2. Pogrešne i neusklađene interakcije (npr. za nastavak interakcije neophodno primiti poruku, a niko takvu poruku ne šalje)

Strukturna kompatibilnost : Posmatra se samo struktura komunikacije. Može biti :

- Jaka : Za svaku poruku koja može biti poslata, **postoji** učesnik koji će je primiti i za svaku poruku koja može biti primljena **postoji** onaj koji će je poslati.
- Slaba : Za svaku poruku koja može biti poslata, **može postojati** učesnik koji će je primiti i za svaku poruku koja može biti primljena **može postojati** onaj koji će je poslati. Nastaje jer su često orkestracije nastajale samostalno.

Kompatibilnost ponašanja : Uzima u obzir kontrolu toka između instanci procesa u konverzaciji. Neophodna je formalna, nedvosmislena reprezentacija.

Workflow modules : Predstavljaju se kao **posebne klasa Petri mreža** koje poseduju specifična komunikaciona mesta koja reprezentuju tok poruke među učesnicima. Kada učesnik šalje poruku, orkestracija traži tranziciju sa ulaznim komunikacionim mestom koje može sadržati poslatu poruku. Na prijemnoj strani postoji odgovarajuće prijemno komunikaciono mesto. **Petri mreža jeste workflow module ako ne postoji tranzicija koja je spojena i na ulazno i na izlazno mesto**. Kada se workflow moduli spoje dobija se **workflow net**.

Implementacija koreografije procesa

Svaka orkestracija mora obezbediti **spolja vidljivo ponašanje** koje je prethodno definisano (interfejsom). Čak i kada se interni procesi naknadno menjaju, to spolja vidljivo ponašanje mora ostati očuvano.

Svaka od uloga se može potencijalno pojaviti u više različitih organizacija sa sopstvenim orkestracijama procesa. Orkestracije moraju biti **u skladu sa očekivanim interfejsom za datu ulogu**.

Realizovana saradnja će biti uspešna ukoliko su :

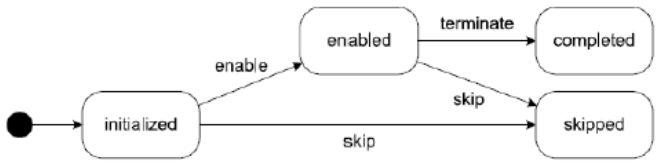
- Interfejsi ponašanja svih uloga međusobno kompatibilni.
- Interne orkestracije procesa za svakog učesnika u skladu sa zadatim interfejsom ponašanja za datu ulogu.

Šabloni interakcije

1. ***Send*** : Jednosmerna interakcija između da učesnika gledano iz perspektive pošiljaoca. Primalac može biti postavljen tokom modelovanja ili izvršavanja procesa
2. ***Recive*** : Jednosmerna interakcija između da učesnika gledano iz perspektive pošiljaoca. Sa stanovišta baferovanja poruka moguće je odbaciti poruke koje nisu očekivane ili sačuvati ih i naknadno obraditi.
3. ***Send/Recive*** : Jedan učesnik šalje poruku drugom koji zatim na nju odgovara. Pošto je konverzacija sačinjena iz više poruka, mora se znati koje poruke čine par.
4. ***Racing Incomming Messages*** : Jedan učesnik očekuje prijem određene poruke, dok ostali imaju mogućnost slanja. Poruke od različitik učesnika se utrkuju i samo prva će biti primljena i obrađena.
5. ***One-to-many send*** : Jedan korisnik šalje više poruka odjednom različitim korisnicima. Lista primalaca može biti poznata tokom modelovanja ili u vreme izvršavanja.
6. ***One-to-many recive*** : Poruke mogu biti primljene od više različitih pošiljalaca, a svaki pošiljalac može poslati tačno jednu poruku. Tipično primalac ne zna broj poruka koji treba da primi, već čeka neki određen broj.
7. ***One-to-many send/recive*** : Učesnik pošalje nekoliko poruka različitim korisnicima i čeka njihov odgovor. Tipično ne čeka sve poruke da pristignu, već se čekanje prekida kad stigne određeni broj.
8. ***Contigent request*** : Učesnik pošalje poruku određenom korisniku, čeka određeno vreme i ako odgovor ne stigne šalje zahtev drugom korisniku.
9. ***Atomic multicast notification*** : Korisnik šalje notifikaciju za više korisnika koji moraju da je prihvate. Nekad je dovoljno da jedan potvrdi notifikaciju, nekad neki određen broj ili čak sve.
10. ***Request with referral*** : Važno kod servisno orijentisanih okruženja gde postoji registar koji omogućuje povezivanje na servis tokom izvršavanja. Koristi se u slučaju da : Korisik A šalje poruku korisniku B i u poruci referiše korisnika C. Iako B ne zna za učesnika C unapred, na osnovu sadržaja poruke može odraditi interakciju sa njim.
11. ***Relaying request***: Tipično za slanje email poruka. Učesnik A šalje poruku učesniku B, koji je prosleđuje C, koji nakon toga ima interakciju sa A. B dobija kopiju poruka kako bi nadgledao konverzaciju.

Let ’s dance:

Jezik za opis **koreografija**. Baziran na šablonima za kontrolu toka i šablonima interakcije servisa. Glavni fokus je na prepoznavanje interakcija i njihovih zavisnosti. Osnovni elementi ovog modela su **elementarne interakcije**. Elementarne interakcije jesu kombinacija *Send* i *Recive* modela aktivnosti.



Životni ciklus interakcija

Razmena poruka (instanca interakcije) se može desiti samo ako je ona omogućena. U slučaju preskakanja moraju se preduzeti koraci za eliminaciju putanja koje su „slepe ulice“. Više elementarnih interakcija se može udružiti u kompozitnu, ali ni jedna ne može postati omogućena pre kompozitne koja ih sadržava. Instance interakcija mogu biti čuvane. Ponavljanja i paralelna grananja se modeluju ponavljanjem interakcija.

BPMN modelovanje koreografija :

Konzerzacioni dijagrami : Prikaz saradnje na visokom nivou apstrakcije tj. ko sa kime komunicira. Neformalna su predstava saradnje procesa. Služe i kao polazna osnova za razvoj orkestracije procesa učesnika.

Konverzacija : Skup logički povezanih poruka razmenjenih između učesnika i kada god se neka poruka razmeni desila se saradnja procesa.

Zadaci koreografije : Jesu osnovni elementi. Predstavljaju poruku ili povezane poruke razmenjene među učesnicima. Svaki skup poruka inicira tačno jeda učesnik. Zadatak sa dva učesnika predstavlja razmenu poruka između njih, tipično kao *request/response* ponašanje. Mogu predstavljati razmenu poruka i između više učesnika.

6. OSOBINE POSLOVNIH PROCESA

Bitan aspekat kod upravljanja poslovnim procesom jeste razmatranje osobina modela. Ako neka osobina postoji u modelu, tada će i sve instance procesa ispoljavati tu osobinu.

Podaci koji mogu nastajati, biti menjani i brisani tokom izvršavanja procesa jesu **aplikativni podaci**. Dva aspekta :

- **Podaci nad kojima aktivnosti vrše neku manipulaciju** : Rešava se na **operativnom nivou**. **PRIMER**: Parametri koji se koriste za pozivanje servisa i njihov tip moraju biti specificirani kod servisno orijentisanih arhitektura.
- **Međuzavisnost podataka** između različitih aktivnosti u procesu : Rešava se na **nivou procesa**. Aktivnosti procesa manipulišu podacima koji opisuju jedan **slučaj obrade**. **Redosled aktivnosti** je često čvrsto vezan uz podatke koji se prosleđuju između aktivnosti. Međuzavisnost podataka određena je **tokovima podataka** tj. kontrole toke moraju pratiti tokove podataka. Ukoliko kontrola toka ne prati tok podataka, instance procesa bi došle u blokirano stanje.

Usklađenost s životnom ciklusom objekta : Pošto model procesa definiše redosled aktivnosti, a aktivnosti izazivaju promene stanja objekta, proces i životni ciklus objekta moraju biti usklađeni.

Strukturna ispravnost : Koriste se Petri mreže za utvrđivanje strukturnih grešaka jer one dozvoljavaju formalnu reprezentaciju strukturne korektnosti modela**. Strukturna greška** jeste **tranzicija** bez odgovarajućih ulaznih i izlaznih mesta. Obavlja se radi utvrđivanja postoje li greške u modelu, mrtve ili beskonačne petlje.

DEFINICIJA : Model procesa je strukturno ispravan ako :

- Postoji tačno jedan **početni čvor**, koji je jedini koji nema **ulaznih grana**
- Postoji tačno jedan **krajnji čvor**, koji je jedini koji nema **izlaznih grana**
- Svi ostali se nalaze na putanji između početnog i krajnjeg

Teorema korektnosti : Crtanje grafova koji predstavljaju stanja nije pogodno za veće modele (javlja se eksplozija stanja) i vreme izvršavanja raste eksponencijalno sa složenošću modela. Koriste se neki od **alata za Petri mreže** kako bi se na osnovu njih proverila korektnost.

Osnovna ideja je da se od *workflow mreže* izvede Petri mreža dodavanjem **tranzicije** *t** koja spaja krajnje i početno mesto. Prema tome, *workflow mreža* je korektna **akko** je u skup veza F uključene i veze {(o, t*), (t*, i)} i ako je ona **životna** i **ograničena**.

- **Živa za svaku tranziciju** : Postoji sekvenca opaljivanja koja će iz početnog stanja doći do svake tranzicije
- **Ograničena** : Kada se desi neka tranzicija, postoji sekvenca opaljivanja koja dovodi u krajnji čvor. Ovo mora biti zadovoljeno kako se tokeni ne bi nagomilavali u čvorovima.

Za velike mreže algoritmi i dalje pokazuju eksponencijalnu složenost, ali postoji podskup mreža za koje je rešenje polinomijalno : **mreže slobodnog izbora**, ali nisu poželjne kao model poslovnog procesa jer ponašanje sistema zavisi od redosleda izvršavanja u konkurentnim granama.

Relaksirani kriterijum korektnosti : Postoji **formalni način provere modela** za koje je neophodna prihvatljiva reprezentacija procesa, a gde je kriterijum korektnosti pretvrd.

DEFINICIJA : *Workflow sistem* jeste korektan **akko** je svaka tranzicija element neke od korektnih sekvenci okidanja. **Korektna sekvenca okidanja** jeste ona sekvenca okidanja koja vodi do stanja mreže iz kog je moguće dostići konačno stanje.

Slabi kriterijumi krektnosti:

Polazne predpostavke :

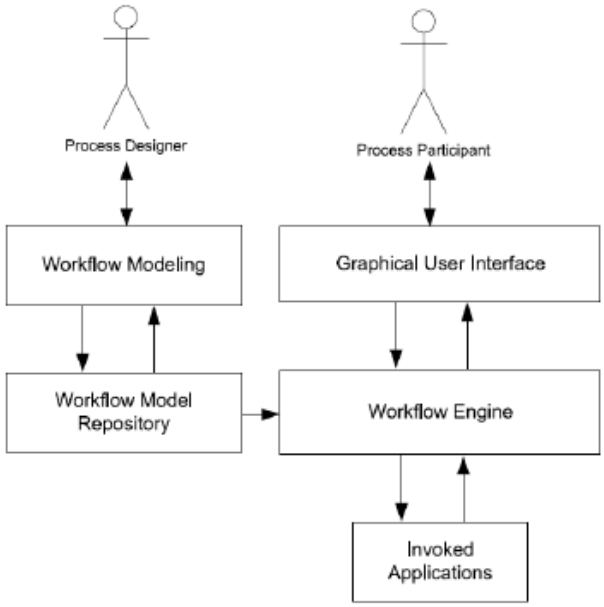
1. Svaka orkestracija u okviru koreografije je reprezentovana kao web servis.
2. Svaki web servis ispoljava određeno konverzaciono ponašanje.
3. Ne moraju se koristiti sve funkcionalnosti koje servis nudi.
4. Koreografija može ispoljavati željeno ponašanje čak i ako se ne koriste sve mogućnosti web servisa.

DEFINICIJA: *Workflow sistem* zadovoljava **kriterijum slabe korektnosti** akko važi da za svako stanje mreže koje je dostižno iz početnog stanja, postoji sekvenca okidanja koja od njega vodi u konačno stanje.

7. ARHITEKTURA SISTEMA ZA UPRAVLJANJE POSLOVNIM PROCESOM

Kod svih sistema za upravljanje poslovnim procesom se jasno razlikuju :

1. **Vreme izrade** (*Build time*) :
   Model poslovnog toka služi kao šablon za izvršavanje. Zavisi od izabrane platforme i može se predstaviti kao izvršni skript ili programski kod. Mora se razviti tako da je u skladu sa operativnim poslovnim procesom. On proširuje operativni model poslovnog procesa detaljima neophodnim za izvršavanje. Može se snimiti kao odgovarajući model podataka u DBMS.
2. **Vreme izvršavanja** (*Run time*) :
   Izvršavanje poslovnog toka počinje kada se **pokrene jedna instanca poslovnog procesa** za koju je kreiran model poslovnog toka. Instancu pokreće *workflow engine*. Striktna separacija na *build i run time* postoji jer ovako pokrenuta **instanca poslovnog toka na zavisi od modela na osnovu kog je kreirana** (tj. kasnije promene u modelu ne uzrokuju promene u izvršavanju).



Arhitektura sistema za upravljanje poslovnim tokom

Opšta arhitektura sistema za upravljanje poslovnim tokovima :

1. Podsistem za modelovanje: Obezbeđuje alata za modelovanje tehničkih aspekata poslovnog toka. Za svaku aktivnost koju u operativnom poslovnom procesu realizuje softver se daje detaljna specifikacija izvršnog okruženja.
2. Repozitorijum modela : Čuvanje i pristup svim kreiranim modelima.
3. Radno okruženje: Workflow engine, jesu platforme na kojima se izvršavaju poslovni tokovi definisani modelima.

Ukoliko se radi i o interaktivnim poslovnim tokovima neophodno je obezbediti i :

4. Podsistem za vezu sa drugim aplikacijama
5. Grafički interfejs za komunikaciju sa korisnicima

*Workflow Menagement Coalition (**WfMC**) :*

Definiše **model arhitekture** koji bi trebalo da implementiraju sistemi za upravljanje poslovnim tokovima tj. ono što sveobuhvatni sistem za upravljanje poslovnim tokom treba da obezbedi korisnicima. Standardizacija interfejsa kako bi se omogućila saradnja raznorodnih sistema je veoma poželjna. Delovi:

1. **Platforma za izvršavanje** : Centralno mesto modela. To je ono što predstavlja workflow engine.
2. **Alati za definiciju procesa** : Alati za modelovanje.
3. **Klijentske aplikacije** : Interfejs ka korisnicima.
4. **Pozivane aplikacije** : Sistem za upravljanje poslovnim tokovima može da koristi druge aplikacija za izvršavanje određenih aktivnosti.
5. **Drugi sistemi za izvršavanje poslovnih tokova** : Prilikom izvršavanja poslovnih aktivnosti sistem može da komunicira sa drugim sličnim sistemima.
6. **Podsistem za administraciju i nadzor** : Omogućuje obavljanje kontrolnih i upravljačkih funkcija nad sistemom.

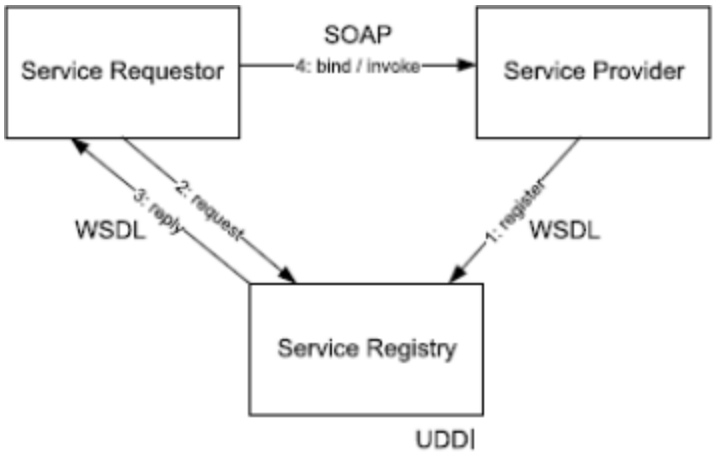
Dinamička adaptacija : Dinamičko prilagođavanje instanci poslovnog procesa novim zahtevima koji nisu bili predviđeni u trenutku modelovanja. Nije deo tradicionalnog modelovanja, slabo je podržano u sistemima za upr. posl. tokovima. Osnovni problemi :

- Kako je dizajnirati i implementirati?
- Kako je kontrolisati i koji kriterijumi korektnosti su primenljivi?
- Koji je opseg adaptacije tj. koje instance koje se trenutno izvršavaju treba prilagoditi novom modelu?
- Kome je dozvoljeno da vrši prilagođavanja i pod kojim uslovima?

Rešenja : Na organizacionom nivou mora biti definisano ko i u kom opsegu može menjati model poslovnih aktivnosti. **Instanciranje** i **izvršavanje** instance zahtevaju drugačiji pristup. Prema tome, koristi se pristup sličan **interpeterskim jezicima** (model se analizira i interpretira u momenti izvršavanja). Primenom ovakvog pristupa **nema više striktne podele** na *build i run time* i promene na modelu se mogu primenjivati i u vreme izvršavanja. Ukoliko se promena odnosi na deo procesa koji je **zadata instanca već obavila** na nju su onda **promene nerelevantne**. U različitim momentima instanca može biti pod kontrolom više različitih modela, ali u jednom trenutku je pod kontrolom samo jednog modela.

Web servisi i tehnologije web servisa : Predstavljaju trenutno stanje implementacije servisno orijentisanih tehnologija.

Web servisi su **potpune**, **samoopisujuće**, **modularne aplikacije** koje mogu biti **publikovane**, **locirane** i **pokrenute** preko web-a. Funkcije koje obavljaju mogu biti u opsegu od jednostavnih operacija do komplikovanih poslovnih procesa. Nakon publikacije, ostale aplikacije mogu da ih pronađu i koriste. Zasnivaju se na opštevažećim standardima.



Osnovne komponente W3C arhitekture

WSDL : jeste fizički i logički ugovor. Logički je jer definiše javno dostupne interfejse, nezavisne od implementacije, formata poruka i transportnog protokola. Fizički je jer definiše i detalje upotrebe servisa i za jedan servis ih može biti vezano više.

Kompozicija web servisa : Omogućuje **labavo** povezivanje aplikacija tj. omogućuje da se nove aplikacije/servisi razvijaju na osnovu postojećih. Opisuje veze između postojećih web servisa. Rekurzivan je koncept.

WS-BPEL : Standardni jezik za **opis kompozicije web servisa**. Omogućuje opis **apstraktnih**(spolja vidljivo ponašanje) i **konkretnih** (sadrže informacije neophodne za izvršavanje web servisa) **procesa**.