Upravljanje matičnih podatkov kot osnova pri vpeljavi storitveno usmerjene arhitekture

Marcel Križevnik, Matjaž B. Jurič
Univerza v Ljubljani, Fakulteta za računalništvo in informatiko, Tržaška 25, 1000 Ljubljana
marcel.krizevnik@fri.uni-lj.si; matjaz.juric@fri.uni-lj.si

Izvleček

Pri vpeljavi storitveno usmerjene arhitekture (SOA – Service Oriented Acrhitecture) se organizacije soočajo s številnimi težavami. Eden izmed glavnih vzrokov za neuspeh projektov SOA je slaba kakovost podatkov, saj so nekateri ključni šifranti (npr. stranke, dobavitelji, produkti) pogosto podvojeni, nekonsistentni, neažurni in nepopolni. Tovrstni problemi lahko ogrozijo uspešnost vpeljave SOA, zato se v zadnjem času vedno več organizacij odloča za vpeljavo celovitega pristopa k upravljanju matičnih podatkov (MDM – Master Data Management), ki predstavlja najnovejši pristop k podatkovni integraciji ter uspešno odpravlja naštete težave. Izkušnje kažejo, da se MDM in SOA odlično dopolnjujeta, saj po eni strani MDM z izboljšanjem kakovosti podatkov poenostavi vpeljavo SOA, po drugi strani pa SOA zagotavlja komponente, ki olajšajo implementacijo storitvenega sloja MDM, tako da je smiselno združiti oba projekta. V prispevku bomo predstavili napreden arhitekturni model SOA-MDM, ki združuje prednosti obeh pristopov in večja raven učinkovitosti in fleksibilnosti celotne arhitekture.
Ključne besede: SOA, storitveno usmerjena arhitektura, MDM, upravljanje matičnih podatkov.

Abstract

Master Data Management as a Basis for Adoption of Service Oriented Architecture

Companies often face a number of difficulties when adopting SOA (Service Oriented Architecture). One of the most common problems is bad data quality as a result of redundancy, inconsistency, inaccuracy and incompleteness of important business data (such as customers, suppliers and products). Such problems may jeopardize the success of adopting SOA, so recently an increasing number of organizations have decided to introduce Master Data Management (MDM), which represents the latest approach to data integration and successfully eliminates the issues listed. Similarly as MDM simplifies the adoption of SOA, also SOA provides some components which facilitate the implementation of the MDM services layer, so it makes sense to combine both projects. In this article, we introduce an advanced SOA-MDM architectural model which combines the advantages of both approaches and thus improves levels of efficiency and flexibility of the whole architecture.
Key words: SOA, Service Oriented Architecture, MDM, Master Data Management.

1 UVOD

Pri vpeljavi storitveno usmerjene arhitekture (SOA – Service Oriented Architecture) [9, 10] organizacije pogosto zanemarijo pomen kakovosti podatkov in izpustijo izvedbo podatkovne integracije. Posledica tega so lahko precejšnje težave pri integraciji aplikacij in implementaciji poslovnih procesov. Če vsaka aplikacija uporablja svoj podatkovni vir, se lahko zgodi, da se posamezna stranka nahaja v več šifrantih, v katerih so strukture podatkov različne, zapisi pa so pogosto nekonsistentni, podvojeni, zastareli in nepopolni [6]. Zadnje raziskave kažejo, da se pričakovana stopnja podatkovnih napak v nekaterih organizacijah povzpne do 30 odstotkov [7]. Še slabši so rezultati druge raziskave [16], v kateri je kar 83 odstotkov sodelujočih organizacij trpelo hude posledice zaradi slabe kakovosti ključnih podatkov. Posledice slabe kakovosti podatkov se čutijo na vseh ravneh organizacije in se odražajo v nepravilnem izvajanju kritičnih poslovnih procesov, zmanjšanem vpogledu v izvajanje poslovnih aktivnosti, netočnem delovanju sistemov za podporo odločanju ter visoki porabi sredstev za odkrivanje in popravljanje netočnih podatkov [1]. Omenjene težave lahko ogrozijo uspešnost vpeljave SOA in učinkovitost organizacije kot celote, zato je pri načrtovanju strategije SOA kot enega izmed začetnih korakov smiselno vključiti tudi izvedbo integracije in konsolidacije vseh ključnih šifrantov.

V prispevku bomo kot rešitev za odpravo opisanih težav predstavili celovit pristop k upravljanju matičnih podatkov (MDM – Master Data Management) [6, 14], ki se dopolnjuje s SOA in zagotavlja dolgoročno kakovost podatkov. MDM predvideva izolacijo ključnih (matičnih) podatkov iz obstoječih podatkovnih virov v centraliziran repozitorij ter vzpostavitev storitvenega sloja, ki omogoča učinkovito uporabo in upravljanje teh podatkov.

V prispevku bomo najprej izpostavili pomen zagotavljanja visoke kakovosti podatkov v organizaciji. V razdelku 3 bomo predstavili osnovne koncepte upravljanja matičnih podatkov (MDM). Nato bomo v razdelku 4 opisali prednosti in slabosti mogočih arhitekturnih stilov MDM ter kako izbrati najbolj ustrezno arhitekturo MDM. V razdelku 5 bomo opisali, kako je mogoče z uporabo principov SOA implementirati učinkovit in fleksibilen sloj matičnih podatkovnih storitev, ki pred poslovnim nivojem skriva vso nizkonivojsko kompleksnost podatkovnega nivoja. V razdelku 6 bomo predstavili predlagani postopek vpeljave MDM, ki povzema dobre prakse in omogoča organizacijam, da se izognejo nekaterim pogostim napakam. V razdelku 7 bomo predstavili sorodne raziskave ter na koncu v razdelku 8 podali sklep prispevka.

2 KAKOVOST PODATKOV V SOA

Pojem kakovost podatkov je zelo širok in predstavlja pričakovano raven kakovosti podatkov, ki organizaciji omogoča učinkovito izvajanje poslovnih aktivnosti. Preden se lotimo odprave problema slabe kakovosti podatkov, moramo definirati in razumeti zahteve (dimenzije) kakovosti podatkov [8, 14, 18]:

- **unikatnost**: zahteva, da so entitete zajete in predstavljene unikatno, kar pomeni, da nobena entiteta iz realnega sveta ne sme biti podvojena in da obstaja enolični identifikator za dostop do posamezne entitete;
- **točnost**: stopnja točnosti, s katero entitete v podatkovnem viru predstavljajo objekte iz realnega okolja. Točnost je v praksi zelo težko spremljati, saj potrebujemo za primerjavo podatkovni vir, za katerega smo prepričani, da je točen in ažuren, saj se lahko podatki v realnem svetu spreminjajo zelo hitro;
- **konsistentnost**: zahteva, da so podatki v eni zbirki podatkov konsistentni s podatki v drugi podatkovni zbirki. Podvajanje podatkov je sicer v nasprotju z zahtevo po unikatnosti, vendar se temu v realnem primeru včasih ne moremo izogniti, saj vedno ni mogoče izvesti popolne integracije in konsolidacije podatkov. Konsistentnost najpogosteje zagotavljamo z redno sinhronizacijo podatkovnih virov;
- **popolnost**: stopnja popolnosti se nanaša na delež od nič različnih (not null) vrednosti. Pri preverjanju popolnosti je treba podatke razdeliti v tri skupine. Prvo skupino predstavljajo obvezni podatki; to so podatki, ki morajo v vsakem trenutku imeti vpisano vrednost. Primer je, recimo, podatek o imenu osebe. V drugo skupino spadajo opcijski podatki. V zadnjo skupino pa spadajo podatki, ki so za nekatere zapise relevantni, za druge pa ne. Podatek o moči v šifrantu artiklov npr. ni relevanten za objekt klobuk;
- **dostopnost**: časovna pričakovanja v zvezi z dostopnostjo informacij. Merimo jo lahko v času, ki preteče med tem, ko je podana zahteva po podatku, in ko je podatek dejansko na voljo. Za lažje spremljanje dostopnosti priporočamo definiranje dogovora o ravni storitev (SLA – Service Level Agreement);
- **ažurnost**: stopnja ažurnosti podatkov v podatkovnem viru glede na realno stanje, saj se lahko podatki v realnem svetu spreminjajo zelo hitro. Ažurnost lahko merimo kot funkcijo pričakovane frekvence spreminjanja podatkov. Primer je pričakovana pogostost spreminjanja telefonskih številk;
- **skladnost s formatom**: vsak podatkovni model definira nabor pravil, ki definirajo predstavitev in način shranjevanja podatkov. Ta pravila določajo podatkovni tip podatkov, maksimalno dovoljeno dolžino ipd.;
- **referenčna integriteta**: pravila za ohranjanje integritete in konsistentnosti podatkov v podatkovnem viru, ki npr. definirajo, da je zaloga vrednosti tujih ključev enaka uniji vseh primarnih ključev za posamezno tabelo v podatkovni bazi.

Večina organizacij se sooča s težavami pri doseganju teh zahtev, a le peščici jih uspe uspešno odpraviti. Samo vpeljava rešitev ERP (Enterprise Resource Planning) in CRM (Customer Relationship Management) običajno ne prinese želenih rezultatov, saj te rešitve ne zagotavljajo dolgoročne kakovosti podatkov, poleg tega ne omogočajo integracije vseh ključnih šifrantov. Organizacije želijo predvsem nov pristop, ne pa le še enega šifranta, ki ga je treba vzdrževati. Iz omenjenih razlogov se v zadnjem času vedno

bolj uveljavlja pristop MDM, ki se odlično dopolnjuje s SOA in zagotavlja dolgoročno kakovost podatkov.

3 UPRAVLJANJE MATIČNIH PODATKOV

Podatke v organizaciji delimo v tri skupine; to so:

- **transakcijski podatki**, ki se uporabljajo pri vsakodnevnem izvajanju aplikacij, imajo izrazito časovno dimenzijo ter predstavljajo rezultat izvedbe posamezne transakcije. Primeri transakcijskih podatkov: naročila, računi, plačila itn. Za razliko od matičnih podatkov količina transakcijskih podatkov raste skoraj premo sorazmerno s poslovno aktivnostjo;
- **analitični podatki**, ki podpirajo sprejemanje poslovnih odločitev. Nahajajo se v podatkovnih skladiščih in se uporabljajo pri podatkovnem rudarjenju. Analitični podatki so shranjeni v velikih tabelah dejstev, ki jih obdajajo ključne dimenzije, kot so stranke, produkti, dobavitelji, računi ipd.;
- **matični podatki**, to so pomembni poslovni objekti, ki jih uporablja več informacijskih sistemov in predstavljajo ključ pri izvedbi transakcij in poslovnih procesov [2]. Ti podatki prav tako predstavljajo dimenzije v podatkovnem skladišču, na podlagi katerih se izvajajo analize za podporo poslovni inteligenci. Najpogostejši primeri matičnih podatkov so šifranti strank, dobaviteljev, produktov, zaposlenih, računov itn. Za uspešno delovanje organizacije je zagotavljanje kakovosti teh podatkov izredno pomembno. Posledice nezadostne kakovosti matičnih podatkov so lahko zelo hude in se izražajo v nepravilnem izvajanju poslovnih aktivnosti, netočnem delovanju sistemov za podporo odločanju, visoki porabi sredstev za odkrivanje in odpravljanje netočnih podatkov, otežen razvoju novih rešitev itn.

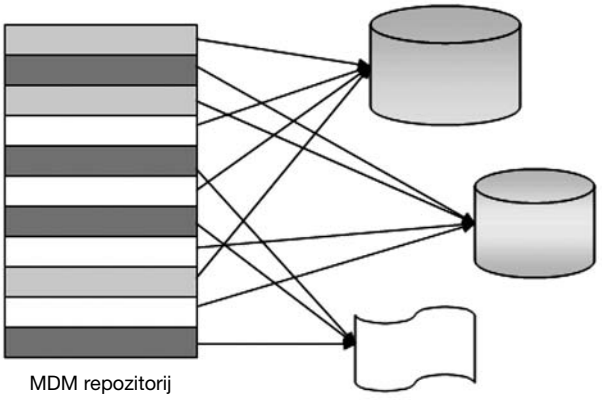
Upravljanje matičnih podatkov (MDM) [6, 14] je celovit pristop k izvedbi podatkovne integracije, ki združuje nabor procesov in orodij za konsistentno definiranje in upravljanje ključnih (matičnih) podatkov. Bistvo MDM je v izolaciji matičnih podatkov iz obstoječih aplikacij v centraliziran repozitorij, ki omogoča celovito upravljanje življenjskega cikla podatkov ter ponuja nabor podatkovnih storitev, s pomočjo katerih lahko do teh podatkov dostopajo vse aplikacije, ki operirajo s temi podatki. MDM se od drugih pristopov k podatkovni integraciji razlikuje predvsem po tem, da zahteva natančno definicijo odgovornosti, vlog in politik za vzpostavitev celovitega upravljanja podatkov (Data Governance), ki se ne zaključi s polnjenjem matičnega repozitorija, temveč predstavlja ponavljajočo se aktivnost v vsem življenjskem ciklu matičnih podatkov. MDM torej zagotavlja trajno kakovost podatkov ter s tem predstavlja idealno podlago za vpeljavo SOA.

4 ARHITEKTURNI STILI MDM

Obstajajo trije glavni arhitekturni pristopi k MDM [14]: navidezni register, transakcijsko vozlišče ter hibridni model. Cilj vseh arhitektur MDM je podpora transparentnosti in skupni uporabi unikatnih predstavitev matičnih podatkov.

4.1 Navidezni register

Arhitekturni stil navidezni register (slika 1) za delovanje uporablja majhen indeks matičnih podatkov, ki vsebuje le minimalen nabor enoličnih identifikatorjev. Register torej vsebuje le kazalnike na dejanske zapise, ki pa se še vedno nahajajo na strani izvornih aplikacijskih podatkovnih virov.



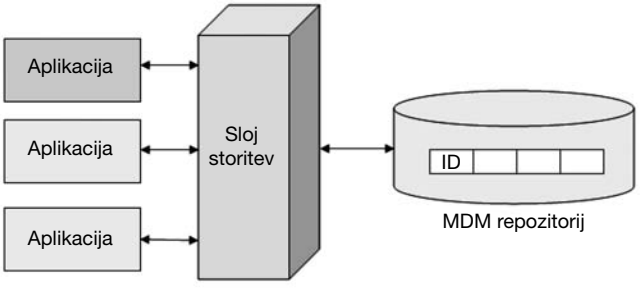
Slika 1 [14]: **MDM arhitekturni stil navidezni register**

Arhitekturni stil navidezni register torej ne predvideva polne integracije podatkov, saj se matični podatki ne nahajajo v centralnem repozitoriju in tako ne moremo govoriti o zlatem zapisu. O konceptu zlatega zapisa lahko govorimo le v primeru, ko so podatki konsolidirani in integrirani ter se nahajajo le v matičnem repozitoriju, ki tako za podatke predstavlja edino verzijo resnice (single version of truth). Prednost uporabe tega arhitekturnega stila je predvsem dejstvo, da obstoječih aplikacij ni treba prilagajati, da bi podatke pridobivale iz centralnega registra

s pomočjo matičnih podatkovnih storitev. Register implementiramo tako, da za vsako entiteto iz realnega sveta v repozitoriju obstaja kvečjemu en zapis, ki vsebuje stolpce s primarnimi ključi, ki kažejo na zapise v podatkovni virih. Če posamezen podatkovni vir ne vsebuje določenega zapisa, je vrednost ključa enaka vrednosti null. Glavna slabost tega pristopa so kompleksna SQL-povpraševanja, ki lahko vključujejo več podatkovnih virov in vračajo podvojene zapise. Ker se matični podatki nahajajo v lokalnih podatkovnih virih, sta dostopnost in ažurnost podatkov visoki. Zaradi dejstva, da se za posamezen objekt iz realnega okolja v podatkovnih virih lahko nahaja več zapisov, pa so stopnje koherentnosti, konsistentnosti in determinizma nizke. Arhitekturni stil navidezni register po navadi predstavlja prvo stopnjo v evoluciji rešitve MDM in ne sme predstavljati končnega cilja, ki je izolacija matičnih podatkov v centralni repozitorij MDM.

4.2 Transakcijsko vozlišče

Transakcijsko vozlišče (slika 2) je celovit repozitorij, ki se uporablja za upravljanje vseh aspektov matičnih podatkov. Vsi matični podatki se nahajajo le v tem repozitorju in se ne replicirajo v druge podatkovne vire, zato je treba ustrezno prilagoditi aplikacije za manipuliranje s temi podatki. Za dostop do matičnih podatkov in upravljanje njihovega življenjskega cikla aplikacije uporabljajo matične podatkovne storitve.

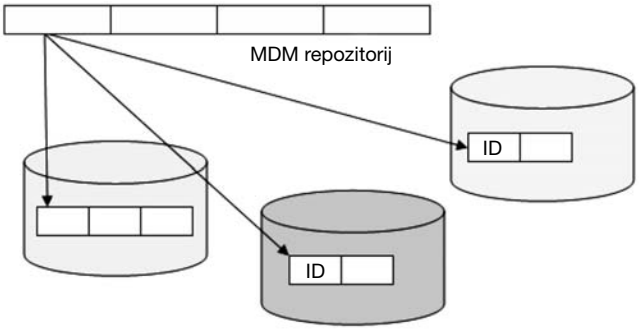


Slika 2 [14]: **MDM arhitekturni stil transakcijsko vozlišče**

Transakcijsko vozlišče funkcionira kot edina verzija matičnih podatkov, pri čemer so vsi atributi podatkov del matičnega repozitorija. Za vsako entiteto iz realnega sveta obstaja v matičnem repozitoriju kvečjemu en zapis, zato lahko govorimo o konceptu zlatega zapisa. Stopnje dostopnosti, ažurnosti, determinizma in konsistence so visoke. Kljub vsem naštetim prednostim pa ima ta arhitekturni stil tudi svoje pomanjkljivosti. Prvo težavo predstavlja dejstvo, da obstoječih aplikacij ni mogoče vedno prilagoditi, saj se lahko zgodi, da nimamo na voljo izvorne kode. Tudi če jo imamo, je lahko prilagajanje aplikacij zelo boleče in zahteva veliko časa in truda. Prav tako se lahko zgodi, da repozitorij MDM postane glavno ozko grlo v arhitekturi.

4.3 Hibridni model

Hibridni model (slika 3) predstavlja kompromis med prej predstavljenima arhitekturnima stiloma in lahko poleg enoličnih identifikatorjev za posamezen zapis vsebuje tudi nabor konsolidiranih skupnih podatkovnih atributov; te podatke imenujemo tudi skupni osnovni matični objekti.

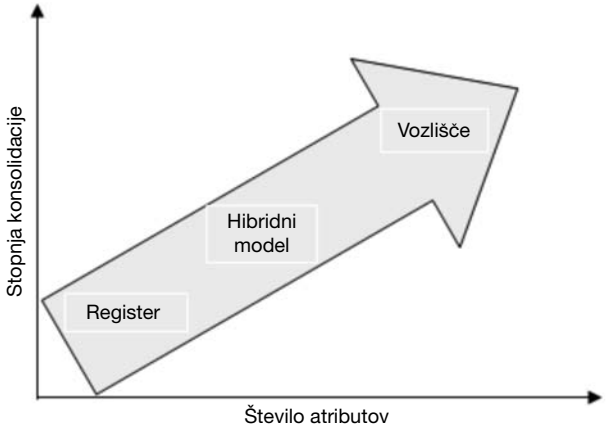


Slika 3 [14]: **MDM arhitekturni stil hibridni model**

Ker se nekateri atributi nahajajo v matičnem repozitoriju, drugi pa v posameznih aplikacijskih virih, je izvedba sinhronizacije še posebno zahtevna, saj ne želimo ogroziti konsistentnosti med podatki v matičnem repozitoriju in podatki v lokalnih podatkovnih virih. Spremembe nad matičnimi objekti se morajo zato propagirati med vsemi lokalnimi kopijami. Izvedba paketne sinhronizacije ob določenih časovnih terminih najpogosteje ne predstavlja ustrezne rešitve, saj ne želimo ogroziti ažurnosti in konsistentnosti podatkov. Vsako spremembo nad matičnimi podatki želimo torej takoj propagirati do lokalnih kopij. Tukaj pa se pojavi problem težavnosti implementacije matičnih podatkovnih storitev, saj je treba ob spreminjanju ali branju matičnega podatka dostopati do več podatkovnih virov in hkrati skriti vso kompleksnost pred poslovnim nivojem. Tukaj nam je lahko v precejšnjo pomoč SOA, kar bomo podrobneje razložili v razdelku 5.

4.4 Izbira ustreznega arhitekturnega stila

Izbira ustreznega arhitekturnega stila je vse prej kot preprosta. Na prvi pogled se zdi najboljša izbira transakcijsko vozlišče, saj zagotavlja visoko ažurnost in konsistentnost, prav tako pa zagotavlja preprosto implementacijo matičnih podatkovnih storitev. Vendar pa ima tudi ta arhitekturni stil svoje slabosti. Pri izbiri ustrezne arhitekture je treba upoštevati te dejavnike: število atributov, zahtevano stopnjo konsolidacije ter sklopljenost aplikacij z matičnimi podatki. Register je najbolj primeren za okolja, v katerih je število atributov majhno in so aplikacije šibko sklopljene, zahteve po konsolidaciji pa niso previsoke. Prav tako ta pristop zahteva najmanj časa, truda in finančnih sredstev, zato pogosto predstavlja prvo stopnjo pri vpeljavi MDM. Ravno nasprotno velja za transakcijsko vozlišče. Hibridni model pa se, kot prikazuje slika 4, nahaja nekje vmes.



Slika 4: **Izbira ustreznega MDM arhitekturnega stila**

Našteti kriteriji pa seveda niso edino merilo pri odločanju o izbiri ustrezne arhitekture. Upoštevati je namreč treba tudi predvideno kompleksnost storitvenega sloja, mehanizme dostopa ter zahteve glede odzivnosti.

5 IZGRADNJA STORITVENEGA SLOJA MDM

Ko so podatki konsolidirani in je matični repozitorij napolnjen, vpeljava MDM še zdaleč ni končana. Zbrani podatki postanejo uporabni šele, ko sta mogoča njihovo upravljanje in uporaba. Ena izmed poglavitnih nalog MDM je torej vzpostavitev sloja matičnih podatkovnih storitev [3, 13, 14, 23], ki aplikacijam omogoča uporabo matičnih podatkov ter upravljanje njihovega življenjskega cikla. Te storitve lahko razdelimo v te skupine:

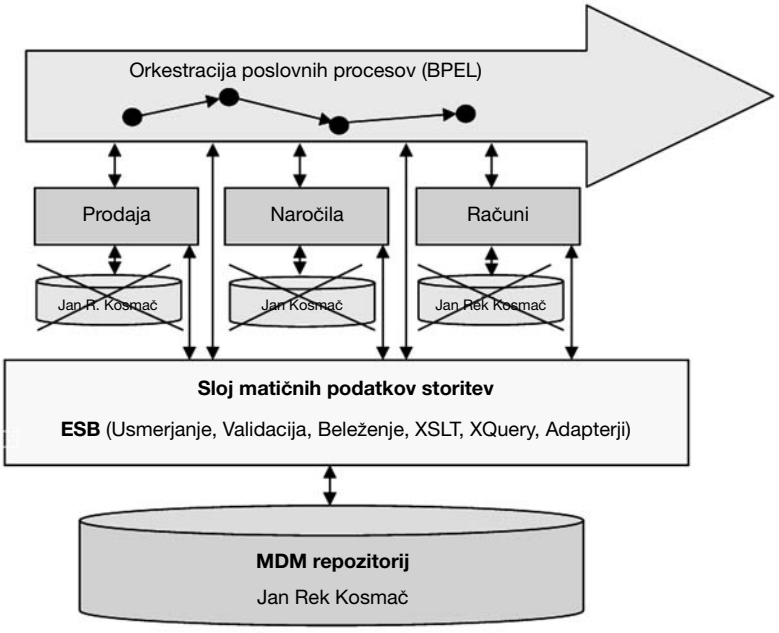
- **kreiranje** matičnega zapisa v centralnem repozitoriju. V primeru arhitekturnega stila navidezni register ali hibridni model vključuje tudi kreiranje zapisa v lokalnih podatkovnih virih. Vstavljanje novega matičnega zapisa lahko vključuje tudi preverjanje skladnosti z definiranimi politikami upravljanja matičnih podatkov. Tako se ob vstavljanju zapisa po navadi preveri kakovost vpisanega podatka ter izvrši preverjanje, če morebiti v matičnem repozitoriju že obstaja podatek za to entiteto. Prav tako se lahko izvrši postopek avtentikacije in avtorizacije (vse osebe nimajo pravice vstavljanja matičnih podatkov). Vsako vstavljanje zapisa se tudi zabeleži skupaj s časovno značko, kar omogoča sledenje sprememb;
- **dostop in uporaba:** branje in spreminjanje matičnih podatkov. V primeru transakcijskega vozlišča gre za preproste operacije branja in spreminjanja podatkov v matičnem repozitoriju. V primeru hibridnega modela ali navideznega registra pa so te operacije bolj kompleksne, saj v transakciji sodeluje več podatkovnih virov. Vsaka sprememba matičnega podatka se zabeleži in preveri se kakovost spremenjenih podatkov;
- **deaktivacija in umik:** ko matični podatek ni več aktualen, ga je treba deaktivirati in umakniti iz uporabe, vzrok za to pa podrobno dokumentirati. Deaktivacija matičnega podatka ne pomeni fizični izbris podatka, temveč ustrezno spremembo statusa;
- **vzdrževanje in popravljanje:** nabor storitev, ki omogoča paketno preverjanje kakovosti podatkov v matičnem repozitoriju in tako zagotavlja dolgotrajno kakovost podatkov. Preverjanje vključuje odpravo podvojenih zapisov, preverjanje konsistentnosti podatkov ipd.

Glavni namen storitvenega sloja MDM je torej aplikacijam omogočiti čim bolj učinkovito in preprosto manipuliranje z matičnimi podatki. Kot smo že spoznali, je težavnost implementacije matičnih podatkovnih storitev v veliki meri odvisna od izbranega arhitekturnega stila. V primeru transakcijskega vozlišča je vstavljanje, branje in spreminjanje matičnih podatkov preprosto, saj se vsi podatki nahajajo znotraj matičnega repozitorija. Če uporabimo hibridni model, pa še zdaleč ni tako. Ker so v tem primeru matični podatki razpršeni v več podatkovnih virih,

je dostop do njih precej bolj zapleten. Ob vstavljanju zapisa se tako v matični repozitorij vstavi zapis z naborom skupnih atributov. Poleg tega se nov zapis vstavi tudi v vse lokalne podatkovne vire, pri čemer se v posamezni vir zapišejo le aplikacijsko specifični atributi. Pri tem je treba upoštevati, da so lahko podatkovni viri različnih tipov. Celotna operacija se mora izvršiti kot globalna transakcija, izvesti se mora tudi preverjanje kakovosti vnešenih podatkov in zabeležiti se mora sprememba. Preprosta operacija vstavljanja torej postane precej kompleksna. Podobno velja za branje podatkov. Implementacijo sloja podatkovnih storitev MDM pa lahko precej poenostavimo z uporabo pristopa SOA. V tem prispevku predlagamo napredni arhitekturni model SOA-MDM, ki omogoča izgradnjo učinkovitega in fleksibilnega sloja matičnih podatkovnih storitev. Model skriva vso kompleksnost podatkovnega sloja pred poslovnim nivojem in vpeljuje koncept t. i. virtualnega zlatega zapisa, ki je neodvisen od izbranega MDM arhitekturnega stila. Pri izgradnji podatkovnega sloja MDM predlagamo uporabo naslednjih komponent SOA, ki so sestavni del vsake celovite platforme SOA:

- **storitveno vodilo** (ESB – Enterprise Service Bus) [10] s svojimi zmožnostmi usmerjanja zahtev in pretvarjanja med različnimi podatkovnimi formati predstavlja komunikacijsko hrbtenico vsake zrele storitveno usmerjene arhitekture, saj odpravlja neposredne povezave med ponudniki in uporabniki storitev ter zagotavlja šibko sklopljenost, ki je ena izmed glavnih zahtev SOA. ESB nam tako omogoča implementacijo fleksibilnih matičnih podatkovnih storitev, pri čemer se zahteva za branje ali spreminjanje matičnega podatka preusmeri do matičnega repozitorija in v primeru hibridnega modela ali navideznega registra tudi do lokalnih podatkovnih virov. Podatke lahko po potrebi ustrezno pretvorimo, saj ESB podpira transformacije med podatkovnimi formati (uporaba XSLT, XQuery). S pomočjo ESB lahko preprosto implementiramo tudi beleženje vseh sprememb ter validacijo podatkov v primeru spreminjanja ali dodajanja zapisov;
- **sistem za upravljanje s poslovnimi pravili** (BRMS – Business Rules Management System) je sestavni del vsake zrele SOA-platforme in nam omogoča izolacijo poslovnih pravil zunaj kode aplikacij, kar omogoča bolj fleksibilno upravljanje poslovnih pravil. Pri implementaciji storitvenega sloja MDM lahko poslovna pravila uporabimo za validacijo kakovosti podatkov ter za preverjanje, kdo ima pravico branja ali spreminjanja matičnih podatkov. Recimo, da spletna trgovina svojim zvestim strankam dodeli status flzlati uporabnik«, kar jim prinaša dodatne ugodnosti v obliki popustov, in da je eden izmed pogojev za pridobitev tega statusa, da je uporabnik registriran že vsaj tri leta. Matična podatkovna storitev mora torej pred vsakim spreminjanjem statusa uporabnika izvesti ustrezno validacijo ter s klicem poslovnega pravila preveriti, ali uporabnik izpolnjuje zahtevana merila;
- **adapterji** omogočajo preprost dostop do različnih tipov podatkovnih virov (relacijske podatkovne baze, podatkovne baze XML, datoteke, sporočilne vrste) s pomočjo uporabe čarovnikov. Uporaba adapterjev nam lahko precej olajša delo z matičnimi podatki, še posebno ko se ti nahajajo v več podatkovnih virih, ki so lahko različnih tipov;
- **jezik BPEL** (Business Process Execution Language) [15] omogoča orkestracijo spletnih storitev v poslovne procese in je ena izmed najpomembnejših komponent vsake platforme SOA. Pri implementaciji matičnih podatkovnih storitev ga lahko uporabimo skupaj z ESB za definiranje kompleksnejših podatkovnih storitev, ki vključujejo tudi pošiljanje obvestil, uporabniške aktivnosti itn. BPEL lahko npr. uporabimo pri spreminjanju kritičnih matičnih podatkov, pri čemer je npr. za brisanje podatkov predvideno večnivojsko ročno potrjevanje za to zadolženih uporabnikov;
- **dogodkovni mehanizmi** (npr. EDA – Event Driven Architecture) [21] podpirajo šibko sklopljeno komunikacijo, ki sledi vzorcu objavi–naroči. V našem arhitekturnem modelu SOA-MDM predlagamo uporabo dogodkovnih mehanizmov za objavo dogodkov o spremembah nad matičnimi podatki v centralnem repozitoriju MDM in posredovanje teh dogodkov do za to zadolženih podatkovnih storitev, ki te spremembe propagirajo do lokalnih podatkovnih virov. Tako lahko zagotovimo visoko stopnjo konsistentosti med podatki v matičnem repozitoriju ter morebitnimi lokalnimi kopijami matičnih podatkov.

Predlagani arhitekturni model SOA-MDM prikazuje slika 5.



Slika 5: **Uvedba storitvenega sloja MDM s pomočjo SOA**

V primeru transakcijskega vozlišča so vsi matični podatki konsolidirani v repozitoriju MDM, v primeru hibridnega modela ali navideznega registra pa se matični podatki nahajajo tudi v lokalnih podatkovnih virih. Storitveni sloj MDM je implementiran s pomočjo pristopa SOA, ki predvideva uporabo storitvenega vodila, jezika BPEL, dogodkovnih mehanizmov, beleženja, uporabe poslovnih pravil, adapterjev itn. Obstoječe aplikacije so tako prilagojene, da do matičnih podatkov dostopajo prek sloja matičnih podatkovnih storitev. Sloj matičnih podatkovnih storitev skriva vso kompleksnost podatkovnega nivoja in zagotavlja pogled virtualnega zlatega zapisa. Rezultat je bistveno izboljšana fleksibilnost, saj imajo razvijalci enovit in konsistenten pogled na matične podatke in se jim ni treba ukvarjati s tem, kje so dejansko shranjeni ti podatki.

6 PREDLAGANI POSTOPEK VPELJAVE MDM

Postopek vpeljave MDM [14] je lahko zelo obsežen in dolgotrajen, zato se ga je treba lotiti premišljeno in z majhnimi koraki. To je še posebno pomembno v primeru, ko se ga lotevamo sočasno z vpeljavo storitveno usmerjene arhitekture. V nadaljevanju je navedenih šestnajst korakov, ki predstavljajo predlagani postopek prehoda in povzemajo številne dobre prakse.

1. Pridobitev podpore vodstva organizacije

MDM služi predvsem kot dobra podlaga za doseganje drugih ciljev, kot so npr. vpeljava sistema SOA ali ERP, izboljšano delovanje sistemov za poročanje, izboljšano obvladovanje napak, olajšano sprejemanje poslovnih odločitev, boljša poslovna produktivnost itn. V tej luči je treba projekt najprej predstaviti vodstvu organizacije, ki mora dati zeleno luč za začetek projekta in odobritev sredstev.

2. Razvoj osnovnega načrta vpeljave MDM

Projekt vpeljave MDM zahteva strateško planiranje s podrobnimi opisi zahtev za posamezne faze. Te zahteve kasneje služijo kot glavno merilo pri spremljanju uspešnosti postopka vpeljave. Primer takšnega načrta je lahko sestavljen iz teh faz: priprava, izvedba pilotnega POC (Proof of Concept) projekta, začetna namestitev, prva verzija, prehod itn. Kadar vpeljava MDM spada v okvir vpeljave SOA, je osnovni načrt MDM seveda del osnovnega načrta vpeljave SOA.

3. Določitev vlog in odgovornosti

Določiti je treba vloge ter pripadajoče odgovornosti za vse osebe, ki sodelujejo pri projektu. V vsakem trenutku mora vsaka oseba natančno poznati svoje zadolžitve. To je eden izmed ključnih korakov v postopku vpeljave MDM. Ena izmed najpomembnejših vlog je vloga skrbnika podatkov, ki je neposredno odgovoren za spremljanje in zagotavljanje kakovosti podatkov enega ali več šifrantov matičnih podatkov [11, 20].

4. Planiranje projekta

Ko je pripravljen osnovni načrt vpeljave ter so dodeljene odgovornosti, je treba pripraviti podroben načrt z natančno dodeljenimi odgovornostmi ter roki za končanje posameznih faz, kar predstavlja podlago za vse naslednje korake.

5. Analiza modelov poslovnih procesov

Poslovni procesi predstavljajo jedro vsake organizacije. Iz tega razloga je treba začeti pri njih, vedno ko želimo izboljšati delovanje organizacije. Tako lahko identificiramo aplikacije, ki sodelujejo pri izvedbi procesov, ter podatke, ki se prenašajo med posameznimi aktivnostmi. Rezultat analize je seznam modelov poslovnih procesov, ki uporabljajo funkcionalnosti, katerih podatkovne vire bi bilo smotrno prestaviti v matično okolje.

6. Identifikacija podatkovnih virov za integracijo

Korak predvideva podrobno preučitev zbranih procesnih modelov z namenom identifikacije ključnih šifrantov, ki predstavljajo potencialne kandidate za izvedbo podatkovne integracije.

7. Obvladovanje podatkov

Korak narekuje definiranje informacijskih politik, ki odsevajo poslovne potrebe in pričakovanja ter procese za spremljanje skladnosti s temi politikami.

8. Vpeljava upravljanja z metapodatki

Upravljanje metapodatkov zagotavlja podlago za vse nadaljnje aktivnosti. Korak predvideva vpeljavo metapodatkovnega repozitorija, ki omogoča kontrolo in spremljanje razvoja matičnega okolja, z vzdrževanjem tehle informacij: rezultati analize podatkov, razlaga posameznih podatkovnih tipov, seznam matičnih podatkovnih tipov, podatkovni modeli matičnih podatkov, opis interakcije med aplikacijami in matičnimi podatki, zahteve glede kakovosti podatkov, dostop do podatkov in njihovo upravljanje itn.

9. Analiza matičnih podatkov

Faza predstavlja pripravo na nadaljnji korak, tj. modeliranje matičnega repozitorija. Na tej stopnji je treba analizirati in konsolidirati različne predstavitve matičnih podatkov, ki se nahajajo v številnih aplikacijskih podatkovnih virih. Rezultati koraka: križno-sistemska podatkovna analiza, zaključeni metapodatkovni profili, poslovna pravila, preslikave in navodila za transformacijo podatkov.

10. Modeliranje matičnih podatkov

Izdelava podatkovnega modela matičnih podatkov za uporabo v matičnem repozitoriju. Rezultata modeliranja sta model za izvedbo konsolidacije ter model za trajno shranjevanje matičnih podatkov.

11. Upravljanje kakovosti podatkov

Eden izmed ključnih ciljev pri vpeljavi matičnega okolja je povečano zaupanje v kakovost podatkov. Za zagotavljanje tega je treba najprej definirati pričakovano raven kakovosti podatkov. Pri zagotavljanju kakovosti podatkov pa si je treba pomagati s posebnimi orodji za razčlenjevanje in standardizacijo podatkov. Pričakovani rezultati koraka: definirane poslovne zahteve glede orodij za izboljševanje kakovosti, formalizacija in implementacija tehnik za trajno zagotavljanje kakovosti podatkov, pridobitev in uporaba orodja za izboljšanje kakovosti podatkov.

12. Izbira arhitekture MDM

Mogoče arhitekture MDM so opisane v razdelku 4. V tem koraku gre torej za analizo sistema z namenom ugotovitve števila atributov ter stopnje sklopljenosti in zahtevane ravni konsolidacije. Ko izberemo najprimernejšo arhitekturo, je treba preučiti še vse rešitve na tržišču ter nato izbrati najprimernejšo.

13. Integracija in konsolidacija podatkov ter polnjenje matičnega okolja

Korak predvideva uporabo integracijskih in konsolidacijskih orodij ter polnjenje matičnega repozitorija, ki od sedaj naprej predstavlja edini vir matičnih podatkov.

14. Uvedba storitvenega sloja MDM

Storitveni sloj sestavlja nabor matičnih podatkovnih storitev, ki omogočajo uporabo in upravljanje matičnih podatkov. Načini implementacije so lahko precej različni, v večini primerov pa se kot najprimernejša izkaže uporaba pristopa SOA, ki je podrobneje opisan v razdelku 5.

15. Izdelava plana prehoda na MDM

Potrebno je preoblikovanje aplikacij, da začnejo namesto lokalnih podatkovnih virov uporabljati podatke iz matičnega repozitorija. Seveda je treba pred začetkom prehoda izdelati podroben plan, v katerem so postavljeni roki posameznih korakov ter identificirane ključne težave, do katerih lahko pride pri izvedbi prehoda.

16. Vzdrževanje MDM

Ko imamo postavljeno arhitekturo MDM in so matični podatki že v uporabi, je treba vseskozi spremljati in nadzirati kakovost podatkov, po potrebi dopolniti podatkovni model in storitveni sloj, zagotavljati redno sinhronizacijo podatkov, izdelovati poročila o kakovosti podatkov itn.

7 SORODNE RAZISKAVE

Številni avtorji [2, 7, 8, 18] so v svojih delih izpostavili pomen zagotavljanja kakovosti podatkov v organizaciji. Kot verjetno najboljšo rešitev za trajno izboljšanje kakovosti podatkov so mnogi predlagali uporabo pristopa MDM [1, 3, 4, 5, 11, 12, 14]. Avtorji v [19] kritično presojajo MDM in ugotavljajo, ali gre dejansko za nov pristop ali samo za že znan pristop v novi preobleki. Prispevek sklenejo z ugotovitvijo, da MDM predstavlja evolucijo obstoječih pristopov k podatkovni integraciji (ERP, CRM), ki se niso najbolje obnesli, in da bo treba na tem področju opraviti še precej raziskav. Cleven in Wortmann [5] predlagata ogrodje s štirimi mogočimi strategijami vpeljave MDM, vendar ne omenjata povezave med MDM in SOA. Na možnost uporabe MDM za zagotavljanje kakovosti podatkov v SOA so v svojih delih opozorili številni avtorji [1, 3, 4, 6, 13, 17, 23]. Wang [22] npr. smatra MDM kot najpomembnejšo strateško začetno točko pri vpeljavi SOA. Loshin [14] izpostavlja simbiotsko razmerje med MDM in SOA, pri čemer storitveno orientirani pristop k razvoju poslovnih rešitev temelji na obstoju repozitorija ključnih poslovnih podatkov in je uspeh vpeljave MDM odvisen od možnosti izgradnje fleksibilnega storitvenega sloja MDM. Kalagirou [12] razlaga, kako si SOA in MDM delita številne razvojne principe, kot so ponovna uporaba, abstrakcija, razvoj na podlagi dobro definiranih pogodb itn. Butler in Pollock [3, 4] omenjata možnost uporabe storitvenega vodila pri izgradnji storitvenega sloja MDM, vendar ne podajata konkretnejših arhitekturnih napotkov, ki bi bili primerljivi z modelom, predstavljenim v tem prispevku. Predlagani arhitekturni model SOA-MDM tako nadgrajuje prispevke omenjenih avtorjev, tako da definira konkretne predloge, kako s pomočjo principov in komponent SOA implementirati fleksibilen sloj matičnih podatkovnih storitev, ki skriva vso kompleksnost podatkovnega nivoja in bistveno poenostavi vpeljavo MDM, ki poteka kot eden izmed začetnih korakov vpeljave SOA.

8 SKLEP

V prispevku smo izpostavili pomen zagotavljanja kakovosti podatkov v storitveno usmerjeni arhitekturi. Identificirali smo ključne težave ter kot najboljšo rešitev za njihovo odpravo predstavili pristop celovitega upravljanja matičnih podatkov (MDM). Opisali smo mogoče arhitekturne stile MDM ter predstavili pomen izgradnje sloja matičnih podatkovnih storitev, ki zagotavlja učinkovito uporabo teh podatkov. SOA in MDM se torej odlično dopolnjujeta, zato je pri prehodu na SOA kot enega izmed začetnih korakov smiselno razmisliti tudi o izvedbi podatkovne integracije s pomočjo pristopa MDM. V prispevku smo predstavili napredni arhitekturni model SOA-MDM, ki omogoča izgradnjo fleksibilnega storitvenega sloja MDM, ki skriva kompleksnost podatkovnega nivoja in vpeljuje koncept virtualnega zlatega zapisa, ki bistveno poenostavi razvoj novih poslovnih rešitev. Najpomembnejše prednosti uporabe predlaganega pristopa so izboljšana kakovost podatkov, izboljšano upravljanje s strankami, konsistentno poročanje, izboljšano obvladovanje tveganj, lažja izdelava analiz in planiranje, olajšan razvoj novih aplikacij, večje zaupanje v rezultate sistemov za podporo odločanju, izboljšana operativna učinkovitost ter nižji stroški.

9 VIRI IN LITERATURA

[1] Andreescu, A., Mircea, M. (2008). Combining Actual Trends in Software Systems for Business Management. 9th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing (CompSysTech '08). Gabrovo, Bolgarija.

[2] Brunner, J.-S., Ma, L., Wang, C., Zhang, L., Wolfson, D. C., Pan, Y., Srinivas, K. (2007). Explorations in the use of semantic web technologies for product information management. Proceedings of the 16th International Conference on World Wide Web, Alberta, Canada.

[3] Butler, D. (2007). MDM as a foundation for SOA, Oracle. Dostopno na: http://www.oracle.com/master-data-management/mdm-foundation-for-soa-white-paper.pdf.

[4] Butler, D., Pollock, J. (2008). Data Management: The Missing Link In Your SOA Strategy. SOA Magazine.

[5] Cleven, A., Wortmann, F. (2010). Uncovering four strategies to approach master data management. 43rd Hawaii International Conference on System Sciences, 2010.

[6] Dreibelbis, A., Hechler, E., Milman, I., Oberhofer, M., Van Run, P., Wolfson, D. (2008). Enterprise Master Data Management: An SOA Approach to Managing Core Information, IBM Press.

[7] Fan, W., Geerts, F., Jia, X. (2008). A Revival of Integrity Constraints for Data Cleaning. Proceedings of the 34th International Conference on Very Large Data Bases, Auckland, New Zealand.

[8] Fox, C., Levitin, A., Redman, T. (1994). The notion of data quality and its quality dimensions, Information Processing & Management, Letnik 30, številka 1.

[9] Havey, M. (2008). The SOA Cookbook: Design Recipes for Building Better SOA Processes. Packt Publishing, Birmingham.

[10] Jennings, F., Jurič, M., Sarang P., Loganathan, R. (2007). SOA Approach to Integration, Packt Publishing, Birmingham.

[11] Joshi, A. MDM Governance: A Unified Team Approach. Cutter IT journal. Letnik 20, 2007. Str. 30–35.

[12] Kalogirou, J. Master Data Management Meets SOA. SOA World Magazine. April 2007.
[13] Križevnik, M., Jurič, M. (2010). Improved SOA Persistence Architectural Model. ACM SIGSOFT Software Engineering Notes, Letnik 35, številka 3.
[14] Loshin, D. (2008). Master Data Management, 1. izd., Morgan Kaufmann Publishers, Burligton.
[15] Mathew, B., Jurič, M., Sarang, P. (2006). Business Process Execution Language for Web Services 2nd Edition, Packt Publishing, Birmingham.
[16] Neil, S. (2007). A new structure for corporate data. Managing Automation.
[17] Newman, D. (2005). The Essential Building Blocks for Enterprise Information Management. Gartner Inc.
[18] Power, D. (2007). Clean, Accurate and Synchronized: Overcoming MDM Challenges. Information Management Special Reports, Februar 2007.
[19] Sammon, D., Frederic, A., Tadhg, N., Carlsson, S. (2010). Making Sense of the Master Data Management (MDM) concept: Old Wine in New Bottles or New Wine in Old Bottles? Proceeding of the 2010 conference on Bridging the Socio-technical Gep in Decision Support Systems: Challanges for the Next Decade. IOS Press Amsterdam.
[20] Snow, C. Embrace the role and value of master data management. Manufacturing Business Technology. Letnik 26, 2008. Str. 38–40.
[21] Taylor, H., Yochem, A., Phillips, L. (2009). Event-Driven Architecture: How SOA Enables the Real-Time Enterprise, Pearson Education.
[22] Wang, R. (2006). Eleven Entry Points To SOA For Packaged Applications. Forrester Research.
[23] Your SOA can be DOA without MDM (2007). TIBCO Software Inc. Dostopno na: http://www.information-management.com/white_papers/10001609-1.html

▪

Marcel Križevnik je mladi raziskovalec v laboratoriju za integracijo informacijskih sistemov na Fakulteti za računalništvo in informatiko v Ljubljani, kjer pripravlja doktorsko disertacijo. Raziskovalno dela predvsem na področjih storitveno usmerjenih arhitektur in računalništva v oblaku. Sodeluje v številnih raziskovalnih in aplikativnih projektih za industrijo. Je soavtor knjige WS-BPEL 2.0 for SOA Composite Applications with Oracle SOA Suite 11g (Packt Publishing). Udeležuje se številnih konferenc s področja informatike, na katerih predstavlja teme s področij upravljanja poslovnih procesov in storitveno usmerjenih arhitektur.

▪

Matjaž B. Jurič je redni profesor na Fakulteti za računalništvo in informatiko Univerze v Ljubljani, kjer vodi laboratorij za integracijo informacijskih sistemov. Ukvarja se z računalništvom v oblaku, storitvenimi arhitekturami, kompozicijo poslovnih procesov, integracijo, elektronskim poslovanjem, spletnimi storitvami in optimizacijo zmogljivosti. Je avtor oz. soavtor knjig WS-BPEL 2.0 for SOA Composite Applications with IBM WebSphere 7, WS-BPEL 2.0 for SOA Composite Applications with Oracle SOA Suite 11g, Oracle Fusion Middleware Patterns, Business Process Driven SOA, SOA Approach to Integration, Best Practices for SOA-based integration and composite applications development, Business Process Execution Language for Web Services (Packt Publishing), .NET Serialization Handbook, J2EE Design Patterns Applied, Professional J2EE EAI in Professional EJB (Wrox Press) in poglavij v knjigah More Java Gems (Cambridge University Press) ter Technology Supporting Business Solutions (Nova Science Publishers), objavljal je v revijah SOA-Web Services Journal, eAI Journal, Java Report, Java Developers Journal in na konferencah, kot so OOPSLA, Oracle Open World, Java Development, BEA Forum, Wrox Conferences idr. Sodeloval je pri številnih projektih doma in v tujini, med drugim tudi pri razvoju RMI-IIOP, sestavnega dela platforme Java 2 in je član BPEL Advisory Boarda. Je predsednik nacionalne komisije za inovacije. Leta 2007 mu je SOA World Journal podelil nagrado za najboljšo knjigo s področja SOA, leta 2010 pa je prejel nagrado za najodmevnejši znanstveni članek s področja storitev iz NMS. Ima naziva Java Champion in Oracle ACE Director.