ISKUSTVA U PRIMJENI ISO STANDARDA ZA PROJEKTIRANJE I PROIZVODNJU SOFTVERA

Perino Krneta, prof. dr.sc. Mile Pavlić, mr.sc. Sanja Čandrlić

SAŽETAK

Ova rad pokušava kroz zahtjev norme ISO 9001:2000, realizacija proizvoda, dati detaljan opis procedure za proizvodnju softvera. Standardizacija ove procedure u obliku kako je prikazana u radu posljedica je primjene ISO sustava upravljanja kvalitetom za područje projektiranja, razvoja, proizvodnje i održavanja softvera. Pored toga u radu se navodi i kratki opis ispunjenja svih zahtjeva norme koji su obuhvaćeni zahtjevom realizacija proizvoda.

ABSTRACT

This paper attempts to come up with a detailed software development procedure, using the ISO 9001:2000 standard as a guideline. The standardization of this procedure, as shown in the paper, results from the use of the ISO quality management system in the field of software design, development and maintenance. Apart from this, the paper provides a short description of the ISO standard requirements fulfilment, for section 7, product realization.

1. UVOD

Međunarodna organizacija za normizaciju (ISO - *International Standards Organization*) svjetsko je udruženje nacionalnih tijela za normizaciju, koje je službeno počelo djelovati 1947. godine, a danas okuplja 148 zemalja. Glavno tajništvo nalazi se u Ženevi. Temeljni joj je zadatak olakšavanje međunarodne razmjene roba i usluga, unapređivanje suradnje u području intelektualnog rada, znanosti, tehnologije i ekonomije. Rezultat rada organizacije je više od 11.000 izdanih međunarodnih normi od kojih su najpoznatije ISO 9000, ISO 14000 i SI sustav mjernih jedinica.

Međunarodne norme obično pripremaju tehnički odbori ISO-a. Svako tijelo član zainteresirano za područje za koje je osnovan tehnički odbor ima pravo biti zastupljeno u tome odboru. Međunarodne organizacije, vladine i nevladine, povezane s ISO-om također sudjeluju u tome radu.

Nacrti međunarodnih normi koje prihvate tehnički odbori šalju se tijelima članovima na glasovanje. Da bi nacrti bili objavljeni kao međunarodne norme potrebno je njihovo odobravanje od najmanje 75% tijela članova koja glasaju.

Normu ISO 9001:2000 CEN (Europski obor za normizaciju) je prihvatio 15. prosinca 2000. godine.

Prihvaćanje sustava upravljanja kvalitetom treba biti strateška odluka organizacije. Na uspostavu i primjenu sustava upravljanja kvalitetom organizacije utječu razne potrebe, konkretni ciljevi, ponuđeni proizvodi, uspostavljeni procesi te veličina i ustrojstvo organizacije.

Tvrtka RIS se na primjenu norme odlučila s ciljem pružanja vrhunske kvalitete naših proizvoda i usluga, te dostizanja unutarnje i vanjske kvalitete poslovanja. Proces certifikacije je uspješno završen u lipnju 2005. godine. i tvrtka RIS je dobila ISO certifikat za **projektiranje, programiranje i održavanje informacijskih sustava i aplikacija.**

Tijekom procesa certifikacije potrebno je ispuniti sve zahtjeve norme koji su definirani točkama 4. do 8. i temelje se na principima procesnog pristupa sustavu, to su:

4. Sustav upravljanja kvalitetom
5. Odgovornost uprave
6. Upravljanje resursima
7. Realizacija proizvoda
8. Mjerenje, analiza i poboljšavanje

Poseban je napor uložen u ispunjavanje zahtjeva 7. Realizacija proizvoda, odnosno u usavršavanje optimizaciju i procesa proizvodnje softvera, čemu je uglavnom i posvećen ovaj rad. Kao osnova za ispunjenje zahtjeva norme u dijelu realizacije proizvoda, odnosno projektiranja, razvoja i proizvodnje softvera korištena je metodika MIRIS. Metodika MIRIS (skraćeno od hrvatskog **M**etod**I**ka za **R**azvoj **I**nformacijskog **S**ustava) je skup metoda i uputa čiji je ukupni cilj projektirati i izgraditi informacijski sustav (IS). Oblikovana je od 1984. godine a objavljena 1995. godine (Pavlić, 1995).

Metodika propisuje faze razvoja i aktivnosti pojedine faze do potrebne razine detalja. Nadalje metodika definira veze između pojedinih aktivnosti i propisuje slijed izvođenja aktivnosti. Ona određuje kada započeti i kada završiti s aktivnošću te koji podaci (informacije, modeli) jesu ulazi u aktivnost a koji su rezultat rada aktivnosti. Za svaku aktivnost je određena metoda kojom se aktivnost izvodi te propisuje kvaliteta izlazne dokumentacije.

Metodike su složene i mogu se promatrati s različitih stajališta. Sa stajališta strukture procesa projektiranja informacijskih sustava metodiku definiramo kao skup aktivnosti modeliranja grupiranih u faze pri čemu se aktivnosti jedne faze izvode uglavnom oko jednog kohezivnog modela.

Metodike sa metodološkog stajališta promatramo kao skup metoda koje dovode do modela sustava.

2. REALIZACIJA PROIZVODA

Proces proizvodnje softvera o koji ćemo detaljnije objasniti u slijedećem poglavlju sadržan je unutar zahtjeva norme koji se odnosi na realizaciju proizvoda. Zahtjev realizacija proizvoda sastoji se od niz zahtjeva koje je potrebno ispuniti. Kao one najvažnije istaknuli bi:

- Procesi usmjereni na kupca

- Projektiranje i razvoj
- Proizvodnja i pružanje usluga

Prije nego krenemo na detaljno objašnjavanje procesa proizvodnje softvera sadržanog u zahtjevu 7.5 Proizvodnja i pružanje usluga recimo nešto i o prethodna dva ne manje važna zahtjeva.

2.1 Procesi usmjereni na kupca

Kod procesa usmjerenih na kupca između ostalih Organizacija (tvrtka) mora odrediti zahtjeve koje postavlja kupac, zahtjeve zakona i propisa koji se odnose na proizvod. Svi ti zahtjevi moraju biti ocijenjeni i to ocjenjivanje mora biti provedeno prije nego tvrtka preuzme obvezu isporuke proizvoda kupcu, odnosno prije potpisivanja ugovora. Pored toga tvrtka mora primijeniti učinkovite postupke komunikacije s kupcima obzirom na informacije o proizvodu, postupanje s upitima, ugovorima ili narudžbama, uključujući dodatke, kao i povratne obavijesti od kupca, uključujući njegove reklamacije.

U tvrtci RIS kod realizacije proizvoda razlikujemo dvije skupine proizvoda:

- softver za poznatog kupca – naručitelja (specijalizirani softver prilagođen konkretnom kupcu)
- univerzalni softver za šire tržište koji nema konkretnog naručitelja (npr. za ustanove privatne zdravstvene njege, univerzalni financijsko računovodstveni informacijski sustav, ...)

Kod softvera za poznatog kupca zahtjevi se definiraju ugovorom. Prije potpisa ugovora Uprava analizira izvedivost, potpunost i smislenost zahtjeva korisnika i po potrebi usklađuje s korisnicima. Tek nakon provjere od strane Uprave, pristupa se potpisivanju ugovora kojim se i formalno potvrđuje pozitivna ocjena istog. Djelatnici RIS-a pri realizaciji proizvoda uočavaju moguće potrebne dodatne zahtjeve koji nisu prepoznati od strane korisnika i nisu definirani ugovorom, pa se tada, u slučaju da se mogu realizirati bez utjecaja na rokove, razmatraju s korisnikom te se u slučaju suglasnosti (aneks ugovora), pristupa izradi istih. Pri promjeni zakonskih propisa, korisnik daje zahtjev za izmjenom, koji se realizira prema ugovoru. Ugovorom je definiran razvoj, proizvodnja i održavanje proizvoda.

Za razliku od poznatog kupca zahtjevi za univerzalni softver se prikupljaju analizom poslovanja tvrtci za koje je softver namijenjen, putem stručne literature, korištenjem dosadašnjeg iskustva, analizom zakonskih propise. Nakon izrade takvog softvere pristupa se procesu prodaje i sklapa se ugovor sa novim kupcem u kome mogu biti izraženi i eventualni zahtjevi za prilagodbom načinu poslovanja konkretnom korisniku.

Svi upiti, narudžbe te eventualne reklamacije hijerarhijski su raspoređeni od najnižeg nivoa ka višem. Sve se odvija preko odgovarajućeg obrasca za zahtjeve korisnika, tako da su izbjegnuti svi eventualni nesporazumi. Ono što prođe na viši nivo, usvaja se i pristupa se razmatranju istog (to prelazi u radni nalog). Ukoliko korisnik ima nekih problema, može i direktno komunicirati s djelatnicima RIS-a za savladavanje istog, a u slučaju nemogućnosti rješenja, upućuje ga se na pismeno upućivanje zahtjeva putem za to predviđenog softvera.

2.2 Projektiranje i razvoj

Zahtjevi projektiranja i razvoja obuhvaćaju: planiranje projektiranja i razvoja, ulazne podatke za projektiranje i razvoj, rezultate projektiranja i razvoja, ocjenu projektiranja i razvoja, ovjeru/verifikaciju projektiranja i razvoja, utvrđivanje prihvatljivosti projektiranja i razvoja, upravljanje promjenama u projektiranju i razvoju.

Za svaki od ovih zahtjeva dati ćemo kratak opis na koji način su oni ispunjeni u tvrtci RIS.

planiranje projektiranja i razvoja

Faze projektiranja i razvoja novih proizvoda, te sve potrebne provjere i testiranja u svakoj pojedinoj fazi, su definirane metodikom MIRIS. U fazi planiranja Uprava određuje voditelja projekta kao i članove tima svojom odlukom. Koordiniranje i praćenje razvoja projekta odvija se tjedno sa direktorima sektora koji po potrebi mogu korigirati planove i potrebne resurse.

ulazni podaci za projektiranje i razvoj

Ulazni podaci za projektiranje i razvoj čine zahtjevi od korisnika, zakonski propisi, zahtjevi koji se eventualno uoče kao i informacije iz prijašnjih projekata.

Za svaki projekt Uprava provjerava i ocjenjuje te ulazne podatke, te ih potvrđuje svojim potpisom na ugovoru za poznatog korisnika, odnosno planom razvoja proizvoda za nepoznatog korisnika.

rezultati projektiranja i razvoja

Rezultati projektiranja i razvoja su Glavni projekt i Izvedbeni projekt što proizlazi iz metodike MIRIS. Rezultati su takvi da ispunjavaju sve zahtjeve dobivene kako ulazne podatke za projektiranje i razvoj, oni su potpuni i daju sve informacije za proizvodnju programskog proizvoda.

U njima su detaljno specificirani svi zahtjevi (Glavni projekt – analiza procesa, Izvedbeni projekt – modeliranje podataka) što u potpunosti zadovoljava kriterije prihvatljivosti softvera u konačnici.

ocjena projektiranja i razvoja

Po završetku projektiranja i razvoja softverskog proizvoda, održava se sastanak projektnog tima na čelu s voditeljem projekta i direktorom sektora. Vrši se ocjena uspješnosti provedenog razvoja u skladu s metodikom MIRIS i ukoliko se uoče određeni nedostaci, o tome se vodi zapisnik kojim je obuhvaćen plan potrebnih aktivnosti, rokovi i odgovorne osobe. Ukoliko je sve u redu, priprema se inicijalna "TODO" lista, formalno se odobrava projektna dokumentacija i prelazi se na proizvodnju.

ovjera/verifikacija projektiranja i razvoja

Direktor sektora zajedno s voditeljem projekta kontinuirano tijekom provedbe projektiranja i razvoja (putem tjednih sastanaka) nadzire nastajanje i usklađenost projektne dokumentacije s obzirom na postavljene ulazne zahtjeve korisnika u skladu s metodikom MIRIS.

U slučaju uočavanja određenih nesukladnosti radi se nova verzija "TODO" liste, koju vodi voditelj projekta.

utvrđivanje prihvatljivosti projektiranja

Na kraju projektiranja i razvoja softvera radi se završna validacija koju provode voditelj projekta i direktor sektora. U slučaju bilo kakvih nedostataka koji se mogu uočiti u toku validacije radi se zapisnik sa konkretnim aktivnostima, rokovima i nositeljima istih. U slučaju da je sve u redu, direktor sektora ovjerava projektnu dokumentaciju.

upravljanje promjenama u projektiranju i razvoju

Do promjena u projektiranju i razvoju može doći prilikom zakonskih promjena, naknadnih zahtjeva korisnika ili uočenih mogućih logičkih potrebnih promjena tokom razvoja. Sve eventualne promjene na razvoju novog proizvoda, voditelj projekta usaglašava s projektnim

Tablica: Aktivnosti proizvodnje softvera

G1: PLANIRANJE PROIZVODNJE
Planiranje aktivnosti proizvodnje SW
Određivanje izvršitelja za pojedine zadatke i određivanje rokova
Određivanje i kreiranje produkcijske, testne i razvojne okoline
G2: OBLIKOVANJE BAZE PODATAKA
Prevođenje logičkog modela podataka u fizički model sheme baze podataka
Kreiranje razvojne okoline za svakog pojedinog programera
Punjenje sheme baze podataka u razvojnoj okoline iz postojeće produkcijske BP
Dodavanje novih koncepata iz modela podataka u razvojnu shemu BP (tip entiteta, ...)
Kreacija razvojne baze podataka
Inicijalno punjenja testne baze podataka
G3: RAZVOJ PROGRAMSKOG PROIZVODA
Izrada glavnog izbornika (aplikacijskog stabla) ili dorada već postojećeg
Izrada ekrana za pregled redaka svake tablice po jednom ili više ključeva
Izrada ekrana za operacije nad jednim retkom tablice (unos, izmjena, brisanje i pregled)
Izrada programskih modula različitih vrsta i namjena kao: obračuna, procedura, funkcija kontrola, look-up-ova nad tablicama (s prvim testiranjem modula)
Izrada izvještaja (s prvim testiranjem modula)
G4: TESTIRANJE U TESTNOJ OKOLINI
Prijenos razvijenih programskih modula u testno okruženje
Spajanje novih modula s postojećim
Back-up verzija softvera
Testiranje prototipa softvera nad testnom bazom podataka
Ažuriranje planova proizvodnje softvera
Izvođenje prema potrebi aktivnosti iz ranijih grupa aktivnosti i ponovno testiranje
G5: TESTIRANJE I ISPRAVLJANJE U RADNOJ OKOLINI
Prijenos razvijenih programskih modula u radno okruženje
Spajanje novih modula s postojećim
Back-up verzija softvera
Punjenje baze podataka
Testiranje prototipa softvera nad produkcijskom bazom podataka od strane programera
Ažuriranje planova proizvodnje softvera
Izvođenje prema potrebi aktivnosti iz ranijih grupa aktivnosti i ponovno testiranje
G6: TESTIRANJE OD STRANE KORISNIKA
Prezentacija softvera korisniku
Testiranje od strane korisnika
Izrada popisa primjedbi korisnika
Ažuriranje planova proizvodnje softvera
Izvođenje prema potrebi aktivnosti iz ranijih grupa aktivnosti i ponovno testiranje
Izrada zapisnika o testiranju i prihvaćanju faze uvođenja

timom i direktorom sektora, te dogovara potrebne radnje i za to odgovorne osobe.

3. PROIZVODNJA I PRUŽANJE USLUGA - PROIZVODNJA SOFTVERA

U ovom poglavlju ćemo se pozabaviti ispunjenjem zahtjeva norme koji se odnosi na proizvodnju i pružanje usluga, odnosno detaljno ćemo opisati proces proizvodnje softvera unutar tvrtke RIS, koji pored projektiranja čini temelj našeg poslovanja.

Proizvodnja softvera započinje nakon izrade projekta IS uzimajući u obzir postojeće programske proizvode i bazu podataka a ima za cilj reorganizirati bazu podataka IS i kreirati potreban novi programski proizvod odnosno izmijenjen postojeći programski proizvod.

Proizvodnja softvera se sastoji od niza aktivnosti. Aktivnosti se mogu svrstati u grupe (oznaka G u tablici) prema vrsti posla uz koji su vezane.

Opišimo pojedine aktivnosti proizvodnje softvera detaljnije.

3.1. Planiranje proizvodnje

Proizvodnju softvera u softverskoj tvornici izvodi tim. Voditelj projekta je odgovoran za uspjeh tima. Njegova prva zadaća je planiranje aktivnosti u vremenu i podjela posla po ljudima.

Glavni izvori znanja od kuda polazi jesu:

- Odabrana metodika proizvodnje softvera,
- Projekt novog i starog IS,
- Shema baze podataka i sama postojeća baza podataka,
- Postojeći programski proizvodi,
- Programski alati za razvoj softvera, CASE alati, RAD, SUBP, ...,
- Operacijski sustavi i mreža računala,
- IT resursi.

U softverskoj tvornici je uglavnom već odabran niz standarda kojeg se razvojni tim treba pridržavati i to: baza podataka, SUBP, CASE alat, metodika, metode projektiranja, standard programiranja i dr.

Ukoliko se formira nova softverska kuća, onda je potrebno definirati sve ove elemente, ili ih prenijeti iz neke slične razvojne okoline. Ovo je izuzetno rizično, jer loš odabir alata i nedovoljna vještina u korištenju alata od strane programera može uzrokovati kašnjenje i propast projekta a ponekad i cijele softverske tvornice.

Tablica: TODO lista projekta «Osnovna sredstva»

Opis aktivnosti	Analiza i Dizajn	Baza	Forma	Brow-se	Report	Kodi-ranje	SQl procedure	Test Unos	Ukupno
Zajednička analiza, dizajn, baza	0,250	0,250							0,500
Izrada šifarnika 'Tip lokacije'		0,125	0,125	0,125	0,250				0,625
Izrada šifarnika 'Lokacija'		0,125	0,125	0,125	0,250				0,625
Spajanje 'Lokacije' s 'Mjestom troška'		0,125				0,250			0,375
Izrada šifarnika 'Tip prometa OSA'		0,125	0,125	0,125	0,250				0,625
Izrada šifarnika 'Konto'		0,125	0,125	0,125	0,250				0,625
Izrada šifarnika 'Mjesto troška'		0,125	0,125	0,125	0,250				0,625
Izrada šifarnika 'Tip OSA'		0,125	0,125	0,125	0,250				0,625
Izrada šifarnika 'Grupa OSA'		0,125	0,125	0,125	0,250				0,625
Izrada šifarnika 'Godina'		0,125	0,125	0,125					0,375
Izrada šifarnika 'Amortizacijska grupa'		0,125	0,125	0,125	0,250				0,625
Izrada šifarnika 'Amortizacijska grupa u godini'		0,125	0,250	0,250	0,250	0,500			1,375
Pregled i ažuriranje osnovnih sredstava		0,250	0,250	0,250	0,250	0,500			1,500
Promet osnovnih sredstava		0,500	0,500	0,500		1,000			2,500
Autorizacija korisnika		0,500	0,125	0,125		1,000			1,750
Ažuriranje nabavne vrijednosti osnovnog sredstva			0,500				1,000		1,500
Ažuriranje naziva osnovnog sredstva			0,125				0,125		0,250
Ažuriranje nomenkalture			0,500				1,000		1,500
Izmjena konta			0,125				1,000		1,125
Prijenos zaduženja na novu lokaciju			0,125				0,500		0,625
Otpis osnovnog sredstva			0,125				1,000		1,125
Mjesečni obračun amortizacije			0,500				5,000		5,500
Godišnji obračun amortizacije			0,500				5,000		5,500
Kartica osnovnog sredstva			1,000		1,000	1,000	1,000		4,000
Izdvajanje osnovnih sredstava po lokaciji			0,125		2,000	1,000		0,500	3,625
Izdvajanje osnovnih sredstava po kontu			0,125		2,000	1,000		0,500	3,625
Izdvajanje osnovnih sredstava po datumu nabave			0,125		2,000	1,000		0,500	3,625
Izdvajanje osnovnih sredstava po amortizacijskoj grupi			0,125			0,125			0,250
Lista za inventuru			0,125		2,000	2,000		0,500	4,625
Plan otpisa osnovnih sredstava (mjesečni)			0,500		2,000	2,000	3,000	1,000	8,500
Plan otpisa osnovnih sredstava (godišnji)			0,500		2,000	2,000	3,000	1,000	8,500
Konverzija podataka iz stare aplikacije osnovnih sredstava	1,000						3,000		4,000
Testiranje sustava									5,000
UKUPNO	1,250	2,875	7,375	2,250	15,500	13,375	24,625	4,000	76,250

Stoga se češće softverske tvornice razvijaju tako da grupa programera napusti neku postojeću tvrtku i prenese znanja i tehnologiju proizvodnje softvera u novu tvrtku.

Voditelj projekta proizvodnje softvera organizira rad tima. U proizvodnji softvera, kao i u drugim proizvodnjama potrebno je praviti planove proizvodnje. Postoje brojne metode za planiranje i vođenje projekata. Većina tih metoda je primjenjiva u softverskoj industriji. Relativno jednostavna i dobra metoda planiranja je pomoći tzv. TODO lista ili analitičkih radnih lista. TODO lista je popis planiranih aktivnosti koje treba izvesti na projektu (vidi tablicu). U prvom stupcu Opis aktivnosti se navode planirane aktivnosti. Kako projekt napreduje tako se dodaju nove aktivnosti u trenutku kada se otkrije da nešto treba napraviti a nije na popisu. Za svaku od aktivnosti se procjenjuje koliko joj treba vremena rada po pojedinim vrstama rada na projektu.

Vrste rada na projektu razvoja IS jesu: analiza i dizajn (odnosno projektiranje), organiziranje baze podataka, izrada ekranskih prikaza kroz koji će se pristupati podacima jednog retka tablice tzv. «Forma» i dr. U ćelijama na presijeku stupca i retka nalaze se podaci o vremenu rada u čovjek/danima. Tako za aktivnost Izrade šifarnika Konta treba 0,625 čovjek dana ili 5 sati. Ukupno za projekt Osnovnih sredstava treba preko 76 dana rada. Na projektu može raditi jedan ili više ljudi pri čemu oni mogu raditi kontinuirano ili isprekidano tako da u međuvremenu rade i neke druge projekte. Datum početka i datum završetka projekta u funkciji su količine ljudi i njihove opterećenosti drugim poslovima. Nakon toga softver je proizveden, ali ne i uveden. Uvođenje zahtijeva posebno planiranje.

U TODO listi se odrede nositelji i odgovornost iza izvršenje posla. Svaki individualni programer točno zna gdje se njegov modul uklapa u okviru cjelokupne aplikacije. Jedan od iskusnijih programera proglašava se vodećim programerom na projektu. Kod manjih timova to je i sam voditelj projekta. Cijela softverska tvornica treba imati jednog odličnog programera zaduženog za kontrolu i standardizaciju programiranja. Njegova je uloga da timu i programerima u raznim timovima pomogne ukoliko naiđu na nepremostive probleme razvoja softvera ili napiše takve programe koji automatiziraju proces proizvodnje. Taj programer radi na razvoju vlastitih CASE alata.

Da bi razvojni tim mogao započeti s radom potrebno im je osigurati uvjete za rad. Najvažniji element jesu tri razine okoline u kojoj će nastajati i koristiti se rezultati softverske proizvodnje i to:

- Razvojna okolina,
- Testna okolina,
- Produkcijska okolina.

Razvojna okolina je radno mjesto programere na kojem programer slobodno i samostalno mijenja bilo koju programsku komponentu ili podatak u bazi podataka a da pri tom ne može narušiti integritet radne baze podataka i radnih programskih proizvoda kako korisnika tako i drugih programera. Dakle, to je potpuno odvojen i organiziran sustav proizvodnje koji ima maksimalan stupanj slobode i minimalan stupanj rizika. Sva stanja ili samo željena stanje razvojne okoline se čuvaju, kako bi se u slučaju potrebe mogao programer vratiti na ranije postavke.

3.2 Oblikovanje baze podataka

Velike baze podataka mogu imati i preko stotinu tablica a jako veliki sustav i preko tisuću tablica, od kojih su neke i preko 1.000.000 redaka u tablici. Takvim velikim bazama podataka manipuliraju specijalisti za upravljanje bazom podataka. Razvojni tim treba koristi njihova znanja i držati se uputa kod izgradnje softvera, kako bi se u fazi uvođenja i testiranja nad produkcijskom bazom podataka lakše i s manje napora uveo novi programski proizvod.

Projektanti u modelu podataka tijekom snimanja korisničkih zahtjeva ne vode računa o postojećoj bazi i njenim postavkama. Model podataka, npr. Dijagram entiteta i veza, se analizira i prevodi u relacijsku shemu baze podataka tako da se poštuje postojeća struktura i standardi u organizaciji baze podataka. Pored toga tim može zaključiti da će logički model baze podataka mijenjati tako da neke tipove entiteta realizira u jednoj ili više tablica iz opravdanih razloga, a da pritom osigura sve korisničke zahtjeve.

Definiranu strukturu sheme baze podataka može potpuno ili djelomično unijeti DBA ili programer.

Kreiranje razvojne okoline za svakog pojedinog programera treba biti napravljeno ako nije napravljeno ranije. U razvojnu okoline se sada treba naknadno dodati sheme baze podataka na kojoj će programer razvijati softver, a koja odgovara shemi postojeće produkcijske baze podataka s dodanim novim tablicama i svim karakteristikama projektiranog modela podataka.

Razvojni programer može samostalno dodati nove koncepte modela podataka u razvojnu shemu baze podataka ako to nije ranije učinjeno ili ako otkrije da je projekt u tom dijelu nedorečen ili manjkav.

Na osnovi nove sheme kreira se skup praznih tablica razvojne baze podataka a potom izvrši inicijalno punjenja testne baze podataka (konverzije tablica iz produkcijske baze, ručni upis).

3.3 Razvoj programskog proizvoda

Programiranje je vrlo stara informatička djelatnost i obiluje nizom karakteristika. Danas se programeri uglavnom služe dobrim razvojnim alatima koji im olakšavaju programiranja (editori za brzi pisanje koda) i koji im automatski programiraju dijelove jednostavnijih programskih modula (CASE alati).

Neovisno u kom razvojnom alatu je izvedeno, programiranje aplikacija ima slična obilježja.

Struktura programskog proizvoda

svaki aplikacijski programski proizvod sastoji se od slijedećih dijelova: programski modul za pristup aplikacijskom sustavu, izbornik, programski moduli za pristup podacima u bazi podataka, različiti programski moduli koji operiraju nad podacima, izvještaji.

Aktivnosti koje treba izvesti programerski tim da bi proizveo potrebne programe jesu:

- Izrada programa za prijavu korisnika (uglavnom postoji kao zajednički modul za većinu aplikacija)
- Izrada glavnog izbornika
- Izrada ekrana za pregled redaka svake tablice
- Izrada ekrana za operacije nad jednim retkom tablice
- Izrada programskih modula različitih vrsta
- Izrada izvještaja

Izrada glavnog izbornika odnosno aplikacijskog stabla je dio programa koji se prikazuje korisniku nakon poziva aplikacije odmah po prijavi na računalo. Ako postoji glavna aplikacija a naša nova aplikacija postaje jedna funkcija glavne aplikacije tada se vrši dorada već postojećeg izbornika i u njega dodaje nova grana.

Pregled grupa podataka u svakoj tablicu u bazi podataka se treba moći pristupiti iz aplikacije. Pristup se vrši kroz

ekran za pregled više redaka odabrane tablice odjednom tzv. «BROWS». Ekrani mogu biti složeni u više razina (TAB-ova) tako da podacima u tablici pristupaju po jednom ili više ključeva. Proizvodnja programskih modula ide od lakših prema težim modulima i to tako da se prvo grade procedure ovim redom: šifarnici, jak tip entiteta, slab tip entiteta, agregirani tip entiteta. Izrada programa podrazumijeva da će programer cijelo vrijeme testirati dobiveni proizvod. Završetak ove aktivnosti znači i završetak testiranja. To je prvo testiranje pojedinog modula.

Podacima u tablici se treba omogućiti pristup i to jedan po jedan redak. Pristup se vrši kroz ekran za pregled jednog redaka odabrane tablice tzv. «FORM». Ekran za operacije nad jednim retkom tablice omogućuju korisniku unos, izmjenu, brisanje i pregled podataka. Izrada tog modula uključuje njegovo testiranje kao i tim povezano testiranje modula BROWS.

Aplikacije se razlikuju među sobom i pored programskih modula za pristup tablicama u bazi imaju više ili manje različitih procedura za obradu podataka (obračuna, procedura, funkcija kontrola, look-up-ova nad tablicama). Primjer takvih programskih modula za projekt osnovna sredstva jest: Otpis osnovnog sredstva, Mjesečni obračun amortizacije, Izdvajanje osnovnih sredstava po lokaciji, Konverzija podataka. Ovi programski moduli se uklapaju u aplikacijsku cjelinu tako da se za njih otvara nova grana u aplikaciji, ili dodaje «Gumb» na postojeće programske module za ekrane tipa «Browse» ili «Form» ili se upoće ne povezuje s aplikacijskim stablom već postoji odvojeno i koristi jednokaratno kao npr. Konverzija podataka iz stare aplikacije osnovnih sredstava.

Uobičajeno je da korisnici traže izradu brojnih izvještaja iz baze podataka na papirnati ili drugi medij.

Cilj proizvodnje softvera je napraviti proizvod koji će zadovoljiti zahtjeve korisnika ali istovremeno osigurati lakoću održavanje programa i to onim programerima koji nisu razvijali programski proizvod. Za to postići važna je upotreba istog CASE alata u jednoj softverskoj tvornici. Pored toga treba definirati standarde programiranja i obavezno dokumentiranje unutar programa kao sto je maksimalna upotreba komentara. Uz CASE alata nije potrebno pisati priručnik koji bi brzo vodio programera do modula koji treba održavanje. CASE alati sadrže dokumentaciju o programskom proizvodu kao što su: prikazi hijerarhije programskih modula, dijagrame logike programa, ulazno/izlazne tablice iz modula, prikaz izvještaja i dr.

Testiranje

Testiranje se pojavljuje tijekom cijelog razvojnog ciklusa i ima niz svojih obilježja. Testiranje je opisano normom od strane IEEE (IEEEstd 829-1998 for software test development).

Aktivnosti testiranja jesu:

- plan testiranja i upravljanje testiranjem
- analiza i dizajn testova
- obavljanje testova
- analiza rezultata i izvješće o testiranju
- završetak testiranja

Svaka od aktivnosti ima odgovarajuće dokumente, sudionike i rezultate testiranja. Konačno testiranje završava zbirnim izvještajem o testiranju.

Pravilo je da svatko testira svoj kod, a integraciju komponenata i testiranje sustava provodi jedna osoba iz tima «tester» ili posebna osoba zadužena za to, a kod većih projekata to može biti i tim za testiranje.

Tester pregledava i odobrava plan:

- Funkcionalnosti koje će se testirati, a koje ne
- Aktivnosti testiranja
- Odgovorne osobe za svaku aktivnost
- Rizici tijekom testiranja

Tester priprema testne slučajeve (ulazni podaci, akcija ili događaj koji testiramo i očekivani rezultat), izvodi i bilježi rezultate i dokumentira greške (izvještaj o testiranju). Tester greške pohranjuje u arhivu i daje nalog za ispravljanje grešaka. Na kraju svakog ciklusa testiranja nastaje zbirni izvještaj testiranja. Ako nema grešaka vrši se dalje isporuka programskim

Ukoliko ne koristimo alate za upravljanje testiranjem i greškama dolazi do problema:

- Programeri ne dobivaju informaciju o grešci u vrijeme kad je nastala
- Sporo rješavanje programskih grešaka i problema od strane programskih inženjera
- Nedovoljna komunikacija između testera, arhitekata programskog sustava, projektanata IS i programera
- Teško mjerenje kvalitete programske opreme i generiranje izvješća
- Otežano praćenje statusa projekta i upravljanje projektom
- Znanje o problemima i prijedlozima poboljšanja često nije dostupno svim članovima projektnog tima

Porastom veličine i kompleksnosti projekta raste i potreba za alatom za testiranje, bilježenje i otklanjanje grešaka.

Programski alati kao baze podataka za bilježenje programskih grešaka jesu: Clear-quest, Atlassian JIRA, Bugzero, Problem Tracker, Serena Trucker. Besplatni open source alati za testiranje jesu :Bugzilla, Itracker, Track+, Mantis bug tracker, Scarab.

Postoje i lokalni alati razvijeni u softverskim tvornicama za testiranje i ispravljanje grešaka, kao i brojni programski alati sastavni dio CASE alata i programskih jezika.

Elementi testiranja

Testiranjem se bave mnogi autori i objavljene su brojne knjige na tu temu (Hetzel, 1984).

U praksi se često izvrši «grubo» testiranje a korisnika posjedne na testno okruženje i zamoli da on testira prototip. Prednosti tog pristupa su što programer ne troši vrijeme na testiranje već to radi korisnik i što se korisnik rano uključuje u učenje nove aplikacije.

Nedostatak je to što korisnik može steći krivu sliku o projektu jer ne očekuje prototip već gotov program. Dobrim pristupom ovo se može otkloniti. Nedostatak je što korisnik ne poznaje tehnologiju testiranja te ne može otkriti sve pogreške.

Elementi koji se mogu testirati jesu:

1. Funkcionalnost softvera

Funkcionira li sve što je zamišljeno ispravno, prihvaća li sustav ulazne podatke, jesu li ispravni izlazi iz sustava, rade li obrade podataka pouzdano, da li nešto nedostaje ili je sustav cjelovit.

2. Unutarnja kvalitete softvera

Jesu li procedure napisane efikasno i kratko i da li se izvršavaju u minimalnim vremenima, je li kod dobro strukturiran kako bi se moduli mogli lako testirati i revidirati, je li program dokumentiran.

3. Održavanje softvera

Može li se sustav prilagoditi potrebama i zahtjevima različitih korisnika i poslovnih situacija u budućnosti, te

može li se lako održavati, mogu li programski moduli biti ponovno korišteni i u drugim aplikacijama,

4. Korištenje aplikacije

Mogu li korisnici brzo naučiti koristiti sustav? Radi li program zadovoljavajuće brzo? Pruža li sustav korisnicima primjereno i poučno vodstvo za pronalaženje i ispravljanje grešaka?

Je li sustav dizajniran tako da nije stalni uzrok muke korisnika, odnosno je li korisnik zadovoljan novom aplikacijom?

Principi testiranja

Po Hetzel (1984) postoje 4 principa testiranja:

- Potpuno testiranje nije moguće
- Testiranje je kreativno i zahtjevno
- Testiranje je bazirano na riziku (vjerojatnosti)
- Testiranje zahtijeva nezavisnost

Ako želite testirati točnost svake IF....THEN....ELSE naredbe, u najjednostavnijem slučaju gdje svaka naredba ima samo dvije mogućnosti (točno ili netočno), u programu sa 10 naredbi treba izvesti 1024 testa a u programu sa 15 naredbi potrebno je obaviti 32 768 testova. Broj testova eksponencijalno raste s povećanjem kompleksnosti sustava. Rijetko imate mogućnosti testirati svaku mogućnost sustava.

Pronađena greška u jednom od izlaznih podataka može korak po korak «prema unazad» dovesti do pravog izvora problema. Većina pronađenih grešaka su simptomi problema. Pronalaženje pravog problema uključuje kreativno i detektivsko razmišljanje i djelovanje.

Kompletno testiranje nije ostvarivo. Odabiru se programski moduli koji imaju najveću vjerojatnost za prekid. Pored njih treba otkriti i one programske module koji imaju najveći rizik gubljenja podataka kada se prekid pojavi. Te rizične module treba u potpunosti testirati.

Moguće je da manje riskantni moduli uopće ne budu testirani, jer su resursi za testiranje potrošeni.

Osoba koja pri testiranju otkrije greške u projektu ili programiranju neće biti popularna među svojim kolegama. Osoba koja je programirala module programa ne bi trebala nikad testirati iste module, odnosno to bi trebao iskusniji programer. U praksi često to rade mlađi programeri.

Upravljanje testiranjem

Općenito postoje dvije strategije testiranja, od glavnog izbornika do najnižih modula ili obrnuto.

Testiranje treba planirati. Plan sadrži popis modula za testiranje s datumom testiranja i osobom koja će testiranje izvoditi.

Svako testiranje programskog modula ili aplikacije treba biti dokumentirano. Dokumentacija sadrži podatke o: Procedure koje se testiraju, datum testa, osoba koja je testiranje izvela, uspješnost testa, komentari u slučaju da nije testiranje uspjelo, datum kada treba ponovo testirati sustav.

Osoba koja testira može imati neke «izmišljene» podatke za testiranje ili može upotrijebiti podatke iz stvarnog svijeta koji su dobiveni od korisnika. Moguće je uspoređivanje novog sustava sa trenutnim izlazom sustava.

3.4 Testiranje u testnoj okolini

Kada pojedinačni programer završi razvoj i testiranje grupe modula na razvojnoj okolini onda iste module prenosi u testnu okolinu. Testna okolina je serversko okruženje na kojem se dalje nastavlja testiranje i razvoj cjeline programskog sustava, na kome se spajaju procedure svih programera u jedan programski sustav i pojavljuju konflikti integracije. I u ovom okruženju se nastavlja ispravljanje grešaka i razvoj sve dok programski proizvod ne bude spreman za primjenu kod korisnika.

Kad jedan programer prenese programske module u testno okruženje i poveže ih s ostatkom programskog proizvoda pojavljuju se konflikti spajanja. Programer može koristiti alate za automatsko spajanje i ispravljanje grešaka ili ručno jednu po jednu grešku otkriti i ispraviti.

Poželjno je prije spajanja napraviti back-up programskog sustava u slučaju gubljanja programskih modula ili potrebe za vraćanjem na prethodno stanje. Nakon otklanjanja svih konflikata programski sustav je poželjno ponovo back-upirati. Sve verzije programskog proizvoda je poželjno čuvati. Nakon uspješnog spajanja programskih modula jednog programera u cjeloviti programski proizvod potrebno je testirati programski proizvod kao cjelinu. Ovo zovemo testiranje prototipa aplikacije. Aplikacija kao programski proizvod koji zadovoljava neku funkcionalnu cjelinu i korisničke zahtjeve sada ima obliku proizvoda koji će biti isporučen korisniku. Podaci u testnoj bazi su takvi da se mogu izvesti svi planirani testovi i načiniti potrebne provjere prije prijenosa prototipa aplikacije u produkciju.

Otkriju li se greške i manjkavosti koje traže dodatni razvoj, ažuriraju se planovi proizvodnje kroz «TODO liste». Poslovi se dodjeljuju programerima i izvodi razvoj prema ranije opisanim procedurama.

3.5 Testiranje i ispravljanje u radnoj okolini

Kada je razvoj u testnoj okolini završen, prototip programskog proizvoda se prenosi u radnu okolinu. Radna okolina je serverski sustav koji poslužuje korisnike za svakodnevno obavljanje svojih poslova (produkcijsko okruženje). U radnoj okolinu se nalazi baza podataka tvrtke koja je slika poslovanja i koja je kontinuirano dostupna korisnicima. U radnoj okolini se ne mogu raditi eksperimenti s podacima i programima jer bi to moglo ugroziti poslovanje.

Stoga se u radnu okolinu prenose samo oni programski moduli koji si prošli testiranje i čije funkcioniranje ne može ugroziti IS, odnosno poslovni sustav. Prijenos je sličan prijenosu s razvojne u testnu okolinu. Spajaju se novi programski moduli s postojećim. Back-upira se stanje softvera prije i poslije spajanja. Baza podataka se posebno back-upirea svojim procedurama i štiti pri bilo kakvim izmjenama.

Zbog novih funkcionalnosti i prelaska na novi programski proizvod potrebno je ponekad napuniti bazu podataka novim tablicama, kako bi se provelo realno testiranje.

Prototip softvera na radnoj okolini je potrebno testirati i to u vrijeme da se ne ometa dnevna produkcija. To se izvodi u popodnevnim satima ili vikendom. U slučaju grešaka sustav se može zaustaviti i vratiti na raniju verziju, kako programa tako i baze podataka.

Ako testiranje pokaže da je potrebno izvršiti daljnje razvojne aktivnosti, ažuriraju se planovi razvoja i pristupa svim potrebnim ranije navedenim aktivnostima.

3.6 Testiranje od strane korisnika

Prototipski pristup je zasebna klasa metodika razvoja IS. Ovaj pristup koriste različite metodike a podrazumijeva izgradnju prototipa programskog proizvoda i njegovo podvrgavanje testiranju kako u programerskom laboratoriju tako i od strane budućih korisnika.

Ovdje pod testiranje od strane korisnika podrazumijevamo da je prototip aplikacije gotov, barem tester i programer misle da je gotov i treba ga testirati od strane korisnika. To je skup aktivnosti od trenutka kada je softver testiran od strane informatičara i s proizvodom krećemo ka korisnicima. Aktivnosti koje ovo testiranje uključuje su:

- Prezentacija softvera korisniku
- Aktivno korištenje programa i testiranje svih varijanti od strane korisnika
- Izrada popisa primjedbi korisnika
- Ažuriranje planova proizvodnje softvera
- Izvođenje prema potrebi aktivnosti iz ranijih grupa aktivnosti i ponovno testiranje
- Izrada zapisnika o testiranju i prihvaćanju faze uvođenja

Ova faza započinje obukom korisnika, ako do sada korisnik nije vidio i koristio programski proizvod.

Korisnik koji ima iskustvo s ranijim projektima ili sklonost testiranju, te koji je prošao obuku za testiranje, brže, lakše i potpunije obavlja posao testiranja. U toj fazi korisnik je sklon pokušaju rušenja programa unošenjem nemogućih kombinacija podataka. Sve to pridonosi poboljšanjima sustava. Pretpostavka je da programski sustav obavlja temeljnu funkcionalnost za koju je proizveden. Ako ne onda je to veliki propust ranijih faza. U toj fazi nastaje niz primjedbi koje pripadaju kategoriji ograničenja i poboljšavanja sustava.

Rezultat testiranja korisnika je zapisnik s listom primjedbi koje će programski tim analizirati i napraviti plan daljnjih aktivnosti na projektu. Ako korisnik više nema primjedbi na programski proizvod izrađuje se zapisnik u kome to piše, te pristupa planiranju faze uvođenja programsko proizvoda u organizacijski sustav.

4. ZAKLJUČAK

Primjena ISO sustava u svakoj tvrtci, a posebno u softverskoj je složen proces koji obuhvaća kompletno poslovanje jedne tvrtke. U taj proces su direktno ili indirektno involvirani svi zaposlenici tvrtke, što dodatno komplicira cijeli proces jer pored „redovnog" posla to predstavlja „dodatni" posao za zaposlenike koji dodatno otežava ispunjenje preuzetih obveza u svakodnevnom poslovanju. Međutim na kraju cijelog puta možemo reći da se sve to skupa isplati iz nekoliko razloga.

Prvo, ISO sustav podrazumijeva standardizaciju procesa, što znači da se sva znanja, u ovom slučaju o projektiranju, razvoju i proizvodnji koja su bila zapisana u knjigama, glavama projektanata i programera standardiziraju. Drugo, tijekom postupka certifikacije ustvari se vrši redizajn poslovnih procesa, što podrazumijeva optimizacije poslovanja, odnosno smanjenje troškova. Treće, i ono najvažnije postupak certifikacije je jedan podrazumijeva kontinuirano poboljšavanja i usavršavanje procesa, u našem slučaju projektiranja i proizvodnje softvera. Procedura za proizvodnju softvera koju smo opisali u ovom radu vrijedi za sada, da li će ovako izgledati i iduće godine teško je reći, jedno je sigurno, stalno će se poboljšavati sa svrhom što kvalitetnije i brže proizvodnje softvera, kako bi u potpunosti udovoljili zahtjevima svih naših korisnika.

Literatura:

1 Pavlić, M., Ivašić, M., Zamlić, I., Methodology MIRIS, Proceedings of the eight Electrotechnical and Computer Science Conference ERK'99", 23.09. - 25.09. 1999., Slovenian Section IEEE, Portorož, Slovenija, 1999. str. 309-312.

2 Pavlić, M., Razvoj informacijskih sustava - projektiranje, praktična iskustva, metodologija, Znak, Zagreb, 1996

3 Sommerville, I.: Software Engineering, Fifth Edition, Addison-Wesley, Harlow, 1995., p. 742

4 Inmon., W. H., (1986), «Information Systems Architecture – a System Developers Primer», Prentice-Hall, NJ, 1986, ISBN 0-13-464694-0

5 Kovačić, A., Vintar, M., (1994), «Načrtovanje in gradnja informacijskih sistemov», DZS, Ljubljana, 1994., ISBN 86-341-1179-2

6 Pavlić, M., (1995), "Tehnologija projektiranja informacijskih sustava", CASE 7, 7. savjetovanje o metodama i alatima za projektiranje IS, Opatija, 5.6.1995.-9.6.1995., str. 1-13, ISBN 953-6129-04-3.

7 Varga. M., (1996), «Faze razvoja informacijskog sustava», Infotrend, Zagreb, 1996., vol. 43, br. 2. str. 54-57.

8 Simon. J.C., (2001), «Introduction to Information Systems», John Wiley $ Sons, NY, 2001., ISBN 0-471-39390-8.

9 Državni zavod za normizaciju i mjeriteljstvo (2002), Hrvatska Norma

Podaci o autorima:

Perino Krneta
Ris d.o.o.
Pilepčić 10
51215 Kastav
tel: 051/687-500
fax: 051/687-501
e-mail: perino.krneta@ris.hr

prof. dr.sc. Mile Pavlić
Ris d.o.o.
Pilepčić 10
51215 Kastav
tel: 051/687-500
fax: 051/687-501
e-mail: mile.pavlic@ris.hr

Prof. dr. sc. Mile Pavlić bavi se metodologijom projektiranja i razvojem informacijskih sustava. Direktor je tvrtke RIS osnovane 1993. za razvoj informacijskih sustava. Aktivno izučava metode za projektiranje informacijskih sustava. Predavač predmeta o projektiranju informacijskih sustava na Odsjeku za informatiku na Filozofskom fakultetu u Rijeci. Bio je pročelnik Odsjeka od 1995. do 1998. Aktivno sudjeluje na razvoju niza projekata u privredi (MOHV, HRT, CROATIA i dr.). Stalni predavač metoda za projektiranje IS od 1986. za projektante. Član programskog odbora za organiziranje CASE - alati i metode savjetovanja. Završio studij Fizika s matematikom na Pedagoškom fakultetu u Rijeci 1980. godine. Na poslovima analitičara/programera u ERC-u, radio od rujna 1982. godine u brodogradilištu 3. Maj u Rijeci. Prvi informacijski sustav o praćenju stanja skladišta završio 1983. godine. Radio na nizu projekata, bilo na projektiranju bilo na fizičkoj realizaciji, kao voditelj, projektant, analitičar ili programer. Održavao predavanja o: arhitekturi operativnih sustava računara, radu prevoditelja, bazi podataka, ekspertnim sustavima, DSS, modeliranju procesa i podataka, organizaciji informatike, informacijskim sustavima i dr. Radio na radnom mjestu projektanta i direktora INFO centra u RiAdria banci d.d. Rijeka. Biran za područje informacijskih znanosti u znanstvenoistraživačko zvanje znanstvenog asistenta 17. ožujka 1992. a u zvanje znanstvenog suradnika 9. studenog 1994. te u docenta 1997. godine. Objavio niz znanstvenih i stručnih radova. Magistrirao na usporedbi raznih metoda za modeliranje podataka, a doktorirao na temi izučavanja procesa praktične primjene informatičkog inženjeringa u poduzećima.

mr.sc. Sanja Čandrlić
Sveučilište u Rijeci, Filozofski fakultet u Rijeci
Omladinska 14
51000 Rijeka
tel: 051 345 050
fax: 051 345 207
e-mail: sanjac@ffri.hr