STRATEŠKO PLANIRANJE INFORMACIJSKOG SUSTAVA - PREGLED MODELA, METODA I KRITERIJA IZBORA

Schönberger, Hana

Sveučilište Josipa Jurja Strossmayera u Osijeku
Ekonomski fakultet u Osijeku
Sveučilišni prijediplomski studij Poslovna informatika

Hana Schönberger

STRATEŠKO PLANIRANJE INFORMACIJSKOG SUSTAVA- PREGLED MODELA, METODA I KRITERIJA IZBORA

Završni rad

Osijek, 2023.

Sveučilište Josipa Jurja Strossmayera u Osijeku
Ekonomski fakultet u Osijeku
Sveučilišni prijediplomski studij Poslovna informatika

Hana Schönberger

STRATEŠKO PLANIRANJE INFORMACIJSKOG SUSTAVA - PREGLED MODELA, METODA I KRITERIJA IZBORA

Završni rad

Kolegij: Oblikovanje i implementacija IS-a
JMBAG: 0010230821
e-mail: hschonberger@efos.hr

Mentor: doc.dr.sc. Dario Šebalj

Osijek, 2023.

Josip Juraj Strossmayer University of Osijek
Faculty of Economics and Business in Osijek
Undergraduate Study Business Informatics

Hana Schönberger

STRATEGIC PLANNING OF INFORMATION SYSTEM - AN OVERVIEW OF MODELS, METHODS AND SELECTION CRITERIA

Final paper

Osijek, 2023.

IZJAVA

O AKADEMSKOJ ČESTITOSTI, PRAVU PRIJENOSA INTELEKTUALNOG VLASNIŠTVA, SUGLASNOSTI ZA OBJAVU U INSTITUCIJSKIM REPOZITORIJIMA I ISTOVJETNOSTI DIGITALNE I TISKANE VERZIJE RADA

1. Kojom izjavljujem i svojim potpisom potvrđujem da je završni (navesti vrstu rada: završni / diplomski / specijalistički / doktorski) rad isključivo rezultat osobnoga rada koji se temelji na mojim istraživanjima i oslanja se na objavljenu literaturu. Potvrđujem poštivanje nepovredivosti autorstva te točno citiranje radova drugih autora i referiranje na njih.
2. Kojom izjavljujem da je Ekonomski fakultet u Osijeku, bez naknade u vremenski i teritorijalno neograničenom opsegu, nositelj svih prava intelektualnoga vlasništva u odnosu na navedeni rad pod licencom *Creative Commons Imenovanje – Nekomercijalno – Dijeli pod istim uvjetima 3.0 Hrvatska.*
3. Kojom izjavljujem da sam suglasan/suglasna da se trajno pohrani i objavi moj rad u institucijskom digitalnom repozitoriju Ekonomskoga fakulteta u Osijeku, repozitoriju Sveučilišta Josipa Jurja Strossmayera u Osijeku te javno dostupnom repozitoriju Nacionalne i sveučilišne knjižnice u Zagrebu (u skladu s odredbama Zakona o visokom obrazovanju i znanstvenoj djelatnosti, NN 119/2022).
4. izjavljujem da sam autor/autorica predanog rada i da je sadržaj predane elektroničke datoteke u potpunosti istovjetan sa dovršenom tiskanom verzijom rada predanom u svrhu obrane istog.

Ime i prezime studenta/studentice: Hana Schönberger

JMBAG: 0010230821

OIB: 34224606726

e-mail za kontakt: hana.sch11@gmail.com

Naziv studija: Poslovna informatika

Naslov rada: Strateško planiranje informacijskog sustava – pregled modela, metoda i kriterija izbora

Mentor/mentorica rada: doc.dr.sc. Dario Šebalj

U Osijeku, 2023. godine

Potpis Hana Schönberger

Strateško planiranje informacijskog sustava – pregled modela, metoda i kriterija izbora

SAŽETAK

Razvojem interneta i potrebom za ključnim informacijama, informacijski sustavi postaju nezaobilazan dio organizacija u 21. stoljeću te je samim time njihovo strateško planiranje i izgradnja dalekosežan proces. U dosadašnjoj je praksi uočeno kako informacijski sustav pomaže djelatnicima organizacije u gotovo svim segmentima poslovanja, od osnovnih i specifičnih funkcija, pomoćnih, upravljačkih do funkcija analize i unaprjeđenja tj. reorganizacije. Organizacije imaju potrebu za izgradnjom različitih informacijskih sustava s različitim funkcionalnostima te izgradnja i razvoj istih često zahtijevaju individualan pristup. Izgradnja informacijskog sustava iziskuje značajna financijska, vremenska i tehnička sredstva, stoga je organizacijama u cilju sistematski i kvalitetno odraditi strateško planiranje. Kako bi informacijski sustav postigao svoj puni potencijal i svrhu, potrebno je izgraditi ga prema već utvrđenim smjernicama informacijskih stručnjaka. U tome su procesu uključene razne sfere informacijskih i ostalih znanosti, a uočeno je nekoliko istih obrazaca i pristupa koje sve organizacije slijede. Kroz povijest su se razvile modeli i metode, a njihov odabir i implementacija ovisi o kriterijima koje organizacija odabere. U sferi IT-a se pojavljuju različiti modeli od kojih su neki derivacije prethodnih modela, a predstavljaju potpuno novi pristup. Unatoč desetinama modela koji se pojavljuju u praksi, bitno je odabrati optimalan kako bi se postigao puni potencijal informacijskog sustava. Metode pružaju uvid u alate koji se koriste prilikom izgradnje IS-a te su kompleksnije. Kriteriji izbora nabrojani u ovom radu mogu poslužiti organizacijama prilikom odabirala prikladnog modela i metode, ali organizacije mogu i samostalno odrediti kriterije izbora.

Ključne riječi: informacijski sustav, model, agilni model, vodopadni model, metoda, SWOT, kriterij izbora

Strategic Planning of Information System - An Overview of Models, Methods and Selection Criteria

ABSTRACT

With the development of the Internet and the need for key information, information systems are becoming an indispensable part of organizations in the 21st century, and thus their strategic planning and construction is a far-reaching process. In practice so far, it has been observed that the information system helps the organization's employees in almost all business segments, from basic and specific functions, auxiliary, management functions to analysis and improvement functions, i.e. reorganization. Organizations need to build different information systems with different functionalities, and their construction and development often require an individual approach. The construction of an information system requires significant financial, time and technical resources, so it is the goal of organizations to carry out strategic planning in a systematic and high-quality way. In order for the information system to achieve its full potential and purpose, it is necessary to build it according to the already established guidelines of information experts. Various spheres of information and other sciences are involved in this process, and several of the same patterns and approaches that all organizations follow have been observed. Throughout history, models and methods have been developed, and their selection and implementation depends on the criteria chosen by the organization. Different models are emerging in the IT sphere, some of which are derivatives of previous models, and represent a completely new approach. Despite dozens of models that appear in practice, it is essential to choose the optimal one in order to achieve the full potential of the information system. The methods provide insight into the tools used when building IS and are more complex. The selection criteria listed in this paper can serve organizations when choosing a suitable model and method, but organizations can also independently determine the selection criteria.

Keywords: information system, model, agile model, waterfall model, method, SWOT, selection criteria

Sadržaj

1. Uvod

Informacijski sustavi sastavni su dio organizacija od 60-tih godina prošlog stoljeća. Danas, u vrijeme digitalizacije i ubrzanog poslovnog okruženja, strateško planiranje informacijskih sustava od ključne je važnosti za organizacije. Informacijski sustavi, koji obuhvaćaju hardver, softver, podatke i procese, postaju sastavni dio funkcioniranja poduzeća u mnogim industrijama. Problematika razvoja informacijskih sustava očituje se u nedovoljnoj informiranosti i nesmotrenom shvaćanju važnosti IS-a za organizaciju i poslovanje.

U prvome su dijelu rada prikazani teorijska podloga i prethodna istraživanja te je u tom poglavlju definiran informacijski sustav zajedno sa strukturom, funkcijama, arhitekturom. Nakon teorijske podloge, u radu će se analizirati modeli i metode koji se koriste prilikom planiranja izgradnje IS-a. Modeli i metode izgradnje informacijskog sustava često se doimaju kompleksnim i apstraktnim, a to otežava njihovo razumijevanje i iskorištavanje punog potencijala kojeg nude. Nerijetko, organizacije nisu upoznate sa svim modelima i metodama koje postoje na tržištu. Modeli i metode prikazani su u drugom potpoglavlju rada. Nadalje, prilikom izrade informacijskog sustava poduzeća nisu sigurna koje kriterije razmotriti i staviti u fokus. U trećem dijelu rada navode se mogući kriteriji izbora odgovarajuće metode i modela IS-a.

Cilj je ovoga rada analizirati najčešće modele, metode i kriterije izbora s kojima se organizacije susreću prilikom strateškog planiranja izgradnje informacijskog sustava.

1. Informacijski sustav

Kako bi se objasnio pojam informacijskog sustava, najprije treba objasniti pojmove poput sustava, podataka informacija, znanja i mudrosti.

Pavlić i sur. (2014) definiraju sustav kao općeniti skup elemenata koji primaju ulaze iz okoline i međudjelovanjem njihovih podsustava pretvaraju te ulaze u određene izlaze te tako ostvaruju funkciju cjeline sustava.

Bellinger i sur. (2004), prema Ackoffu (1989), interpretiraju značenje podatka, informacije, znanja, razumijevanja i mudrosti. Simbol koji sam po sebi nema značenje i može postojati u više oblika nazivaju podatak. Taj obrađeni podatak s točno određenim značenjem i odgovorima na određena pitanja poput „tko“, „što“ „gdje“ i „kada“ predstavlja informaciju. Znanje predstavlja primjenu svih podataka i informacija te pruža odgovor na pitanje „kako“. Kako bi se znanje preuzelo i koristilo, važno je odgovoriti na pitanje „zašto“ i taj odgovor onda predstavlja razumijevanje novog znanja iz onog starog. Mudrost predstavlja vrednovanje znanja.

„Informacijski sustav (IS) je međusobno povezan skup komponenti koji se koriste za prikupljanje, pohranu, obradu i prijenos podataka i digitalnih informacija. U svojoj srži, to je zbirka hardvera, softvera, podataka, ljudi i procesa koji rade zajedno kako bi transformirali sirove podatke u korisne informacije. IS podržava niz poslovnih ciljeva kao što su poboljšana usluga korisnicima ili povećana učinkovitost.“ (TechTarget, 2023).

1.1. Struktura informacijskog sustava

Šimović (2009:14) navodi da za uspješno obavljanje svih aktivnosti – prikupljanja, pohranjivanja, obrade i dostavljanja podataka i informacija korisnicima - suvremeno, znanstveno projektirani i izgrađeni informacijski sustav treba biti sinteza pet neophodnih komponenata:

- **hardvera** koji predstavlja materijalnu osnovicu informacijskog sustava. Tu se ubrajaju elektroničko računalo, ulazno-izlazni uređaji, dio uređaja i sredstava za komuniciranje i prenošenje podataka na daljinu, a koji je neposredni dio računala te ostalu računalnu opremu za obradu podataka;

- **softvera** koji obuhvaća nematerijalne elemente informacijskog sustava: programe, uvježbanost i metode vezane za organizaciju, upravljanje obrađivanje i korištenje rezultata obrade podataka i informacija. Software predstavlja skup aplikacijskih programa koji kontroliraju i koordiniraju hardverske komponente (GeeksforGeeks, n.d.a);
- **netware-a** koji označava mrežnu osnovu informacijskog sustava. Tu ubrajamo različitu komunikacijsku hardversko-softversku opremu, mrežne ulazno-izlazne uređaje i sredstva za komuniciranje i prenošenje podataka na daljinu koji nisu neposredni dio računala te ostalu opremu za olakšavanje daljinske obrade podataka;
- **lifeware-a** koji predstavlja ekipu stručnjaka – organizatora elektroničke obrade podataka, sustavnih analitičara, programera, operatera te korisnika informacijskog sustava tijekom određenog životnog ciklusa informacijskog sustava
- **orgware-a** koji obuhvaća organizacijske postupke, metode i načine usklađivanja i povezivanja triju komponenata u skladnu, funkcionalnu, ekonomičnu i djelotvornu cjelinu.

1.2. Funkcija informacijskog sustava

„Pod funkcijom sustava razumijevamo svrhu postojanja sustava, ulogu koju sustav ima u svojoj okolini i način ostvarivanja svrhe. Iz jasno se definirane funkcije sustava utvrđuju njegove komponente i njihovi međuodnosi (struktura). Primjeri su funkcije sustava nabava robe u tvornici, ponašanje slušnih organa na zvuk, operacije dijelova električnoga zvona pri uključenju, način djelovanja biljke (i to funkcioniranje korijena, lista, cvijeta, stabla) tako da funkcije dijelova budu podređene rastu i razvoju biljke kao cjeline (Pavlić i sur., 2014:11).“

Opća se raščlamba funkcija u organizacijskome sustavu može kategorizirati u četiri grupe i to (Pavlić i sur., 2014:11):

A. osnovne-specifične funkcije (zavisne od grane djelatnosti i opsega primarnih funkcija, tj. ima li poduzeće vlastiti razvoj ili radi po licenci, izvodi li testiranje ili održavanje svojih proizvoda - ideja opsega prema ISO 9000 seriji standarda 9001, 9002, 9003),
B. pomoćne funkcije (pravna, kadrovska, financijska, sigurnosna itd., bez obzira na djelatnost),
C. upravljačke (planiranje, praćenje, mjerenje, osiguranje kvalitete, rukovođenje) i
D. funkcije analize i unaprjeđenja tj. reorganizacije.

Pavlić (2011) navodi temeljne funkcije informacijskog sustava:

a) prikupljanje i upis podataka u bazu podataka - postiže se programima za prihvat podataka, pri čemu se podatci s ulaznih i izlaznih dokumenata upisuju u bazu podataka,
b) obrada podataka - ostvaruje se programima za automatizaciju procesa sustava, na način da se uzimaju podatci iz baze podataka i novi podatci iz sustava te se procesiraju podatci za bazu i sustav,
c) prikaz i diseminacija podataka iz baze podataka - ostvaruje se programima za izradu izvješća,
d) čuvanje (dokumentiranje, pohranjivanje) podataka – osigurava dostupnost i sigurnost podataka za buduće potrebe sustava.

1.3. Arhitektura informacijskog sustava

Za uspješno povezivanje strukture i funkcionalnosti, važno je razviti arhitekturu informacijskog sustava. „Softverska arhitektura definira temeljnu organizaciju sustava i jednostavnije definira strukturirano rješenje. Definira kako su sastavljene komponente softverskog sustava, njihov odnos i komunikaciju među njima. Služi kao nacrt za softversku aplikaciju i razvojnu osnovu za razvojni tim.“ (Jena, n.d.). Cilj arhitekture IS-a je analizirati, definirati i uskladiti razvoj informacijskih sustava na temelju poslovne strategije i procesa, a ovisno o tehnološkim inovacijama (Bardon, n.d.). Najčešće korišten arhitektonski obrazac je višeslojna arhitektura, koja ublažava složenost modernih aplikacija. Ova softverska arhitektura dijeli vašu aplikaciju na slojeve — neovisne dijelove sustava od kojih svaki ima svoje različite i specifične odgovornosti. Obično je svaki sloj zatvoren, dakle komuniciraju samo sa svojim susjednim slojevima (Wong, 2020) Najjednostavniji oblik višeslojne arhitekture je troslojna arhitektura koja se sastoji od sljedećih slojeva (Packt, 2018):

- Prezentacijski sloj - prvi i najviši sloj koji je prisutan u aplikaciji. Ova razina pruža prezentacijske usluge, odnosno prezentaciju sadržaja krajnjem korisniku kroz grafičko korisničko sučelje (GUI). Ovoj razini može se pristupiti putem bilo koje vrste klijentskog uređaja poput stolnog računala, prijenosnog računala, tableta, mobilnog telefona, tankog klijenta i tako dalje. Da bi se sadržaj prikazao korisniku, relevantne web stranice treba dohvatiti web preglednik ili druga prezentacijska komponenta koja

se izvodi na klijentskom uređaju. Za predstavljanje sadržaja bitno je da ova razina komunicira s ostalim razinama koje su prisutne prije nje,

- Aplikacijski sloj - srednji sloj arhitekture. Ovo je razina u kojoj se izvodi poslovna logika aplikacije. Poslovna logika je skup pravila koja su potrebna za pokretanje aplikacije u skladu sa smjernicama koje je postavila organizacija. Komponente ovog sloja obično se izvode na jednom ili više aplikacijskih poslužitelja,
- Podatkovni sloj - najniža razina ove arhitekture i uglavnom se bavi pohranjivanjem i dohvaćanjem aplikacijskih podataka. Podaci aplikacije obično se pohranjuju u poslužitelju baze podataka, poslužitelju datoteka ili bilo kojem drugom uređaju ili mediju koji podržava logiku pristupa podacima i pruža potrebne korake kako bi se osiguralo da su samo podaci izloženi bez pružanja ikakvog pristupa mehanizmima za pohranu i dohvaćanje podataka. To radi podatkovni sloj pružanjem API-ja aplikacijskom sloju. Pružanje ovog API-ja osigurava potpunu transparentnost podatkovnih operacija koje se izvode na ovoj razini bez utjecaja na aplikacijsku razinu. Na primjer, ažuriranja ili nadogradnje sustava u ovoj razini ne utječu na razinu aplikacije ove arhitekture.

Kompleksnije višeslojne arhitekture, također se sastoje od prezentacijskog, aplikacijskog i podatkovnog sloja, s time da je podatkovni sloj rastavljen na slojeve kao što su (Wong, 2020):

- Poslovni sloj - usklađuje radnje s poslovnim pravilima koja su definirana u poslovnim procesima
- Sloj postojanosti - obrađuje primljene zahtjeve i sadrži logiku koja određuje kako se podaci trebaju mijenjati u bazi podataka.

2. Modeli planiranja IS-a

U ovom poglavlju pažnja je posvećena samim modelima planiranja i na kojim principima ti modeli rade. Modeli razvoja opće su ideje o mogućim putovima razvoja informacijskog sustava (Pavlić, 2011:119). Modeli se u literaturi dijele na tradicionalne i agilne te će se njihove definicije i principi detaljno razraditi u sljedećim potpoglavljima. Obuhvaćeni modeli su agilno programiranje, ekstremno programiranje, model vodopada, evolucijski model, spiralni model te iterativni model. Svaki od njih biti će detaljno opisan i uspoređen s ostalima. Navedene su prednosti i mane modela te koliko su primjenjivi u današnje vrijeme.

2.1. Agilno programiranje

Prema Enciklopedija.hr (n.d.) agilnost (lat. *agilitas*) predstavlja pokretljivost i brzinu te se sintagma agilna metodologija proučava u kontekstu fleksibilnog i brzog razvoja softverskih aplikacija.

U prijašnjoj se praksi koristio model vodopada, koji je u svojoj osnovi rigidan s puno dokumentacije. Ovaj je model bio moguć sve do trenutka kada su se na tržištu počele razvijati internetske i mobilne aplikacije, koje zahtijevaju drugačiji pristup. Struktura razvojnog tima postaje manja, često okupljena u manjim organizacijama sa slabijom informatičkom podrškom. Promjene na tržištu, veliki konkurentski i vremenski pritisci dovode do potrebe za pronalaskom drugačijeg modela razvoja softvera. Potaknuti promjenama, 2001. godine skupina iskusnih programera počinje prakticirati softverski razvoj poznatiji pod nazivom agilno programiranje (Sacolick, 2022).

Agilni model predstavlja pristup razvoja softvera temeljen na iterativnom razvoju u kojem je projekt podijeljen na manje dijelove te se izbjegava dugoročno planiranje. U pravilu se opseg i zahtjevi projekta definiraju na početku razvojne faze. Broj ponavljanja, trajanje i opseg svake iteracije jasno su određeni unaprijed. U ovome je modelu, svaka iteracija mali vremenski "okvir" koji traje od jednog do četiri tjedna (Javatpoint, n.d.).

Kako bi agilni model bio što preciznije definiran, jasniji i optimalno implementiran u praksi, stvoren je agilni manifest. Navedeni akt predstavlja konsenzus 4 temeljnih vrijednosti i 12 principa agilnog programiranja.

Četiri temeljne vrijednosti agilnog modela, pokazuju da se u ovome modelu više cijeni i kao prioriteti se izdvajaju (Beck, i dr., 2001):

1. Pojedinci i interakcije ispred procesa i alata
2. Radni softver ispred opsežne dokumentacije
3. Suradnja s klijentima ispred pregovora o ugovoru
4. Reagiranje na promjenu umjesto slijeđenja plana

Iz temeljnih vrijednosti agilnog modela, proizlazi dvanaest sljedećih principa (Agilni manifest, n.d.):

1. Naš je najvažniji prioritet zadovoljstvo klijenta kroz ranu i neprekinutu isporuku softvera koji nosi vrijednost.
2. Promjenjivi su zahtjevi dobrodošli, čak i u kasnijoj fazi razvoja. Agilni procesi koriste promjene u svrhu poboljšanja klijentove konkurentske prednosti
3. Česta isporuka uporabljivog softvera, u vremenskim razmacima od nekoliko tjedana do nekoliko mjeseci, preferirajući kraće razmake.
4. Poslovni ljudi i razvojni programeri moraju svakodnevno surađivati, tijekom cjelokupnog trajanja projekta
5. Razvoj projekta postiže se oslanjanjem na motivirane pojedince. Pruža im se okruženje i potrebna podrška te im se s povjerenjem prepušta posao.
6. Razgovor uživo predstavlja najefektivniji i najefikasniji način prijenosa informacija razvojnom timu i unutar samog tima
7. Uporabljiv softver je temeljno mjerilo napretka
8. Agilni procesi podržavaju održivi razvoj. Pokrovitelji, razvojni tim i korisnici trebaju imati mogućnost održavanja jednakog tempa rada neograničeno dugo.
9. Kontinuirani naglasak na tehničkoj izvrsnosti i dobrom dizajnu poboljšava agilnost.
10. Jednostavnost – umijeće povećanja količine posla kojeg nije potrebno raditi– je od presudne važnosti.
11. Samoorganizirajući timovi stvaraju najbolju programsku arhitekturu, zahtjeve i dizajn
12. U redovnim intervalima, razvojni tim razmišlja o tome kako postati učinkovitiji te u skladu s time prilagođava svoje ponašanje.

Hamilton (2023) navodi prednosti agilne metodologije:

- agilni je model usmjeren na klijenta. Drugim riječima, klijent je kontinuirano uključen tijekom svake faze izrade softverske aplikacije.
- agilni model svojim pristupom razvoja softvera osigurava održavanje kvalitete razvoja

- proces se u potpunosti temelji na inkrementalnom napretku, što znači da klijent i tim znaju što je dovršeno. Ovakvim se pristupom smanjuje rizik u procesu razvoja.

Hamilton (2023) navodi nedostatke agilne metodologije:

- agilni model nije pogodan za male razvojne projekte.
- trošak implementacije agilne metode je veći u usporedbi s drugim razvojnim modelima.
- projekt može lako skrenuti s puta ako voditelju projekta nije jasno kakav ishod želi

Iako je u svojoj osnovi agilni model fleksibilan i sklon promjenama, ipak postoje faze kojih se razvojni tim pridržava prilikom agilnog programiranja. Prema Javatpoint (n.d.) agilni model podijeljen je u sljedeće faze

1. Prikupljanje zahtjeva
2. Dizajniranje zahtjeva
3. Konstrukcija/iteracija
4. Ispitivanje/osiguranje kvalitete
5. Implementacija
6. Povratne informacije

Prikupljanje zahtjeva

U ovoj se fazi definiraju zahtjevi te se na temelju prikupljenih informacija procjenjuje tehnička i ekonomska izvedivost projekta (Javatpoint, n.d.). Zahtjevi se dijele na (Indeed, 2023):

- funkcionalne – objedinjuju procese i informacije potrebne za funkcioniranje sustava
- nefunkcionalne – stavljaju u fokus operativne i tehničke aspekte poput enkripcije, sigurnosti i oporavka od katastrofe.

Kako bi se navedeni zahtjevi prikupili i ispravno tumačili preporučuje se koristiti sljedeće provjerene metode prikupljanja zahtjeva (Indeed, 2023):

- intervju jedan na jedan
- skupni intervju
- brainstorming

- fokusne grupe
- istraživanje

Dizajniranje zahtjeva

Nakon prikupljanja zahtjeva, sljedeći je korak dizajniranje zahtjeva, najčešće pomoću UML dijagrama (Javatpoint, n.d.a). Jedinstveni jezik za modeliranje (engl. *Unified Modeling Language* – UML) je jezik za modeliranje opće namjene. Glavni cilj UML-a je definirati standardni način za vizualizaciju načina na koji je sustav dizajniran. Nalikuje nacrtima koji se koriste u drugim područjima inženjerstva (GeeksForGeeks, 2023).

Konstrukcija/iteracija

Nakon definiranja zahtjeva, razvojni tim počinje raditi na softveru (Javatpoint, n.d.). Razvojni timovi često koriste integrirane razvojne okoline koje potpomažu razvoj softvera. Općenito, integrirana razvojna okolina predstavlja radnu površinu utemeljenu na grafičkom korisničkom sučelju (GUI) dizajnirana za podršku programeru u izradi softverskih aplikacija s integriranim okruženjem u kombinaciji sa svim potrebnim dostupnim značajkama. Najčešće značajke su mogućnost otklanjanja pogrešaka, kontrole verzija i pregledavanja strukture podataka (Rouse, 2017).

Testiranje

U ovoj fazi tim za osiguranje kvalitete ispituje performanse proizvoda i traži grešku. U agilnom modelu se testiranje provodi istovremeno s razvojem softvera. Pristupi testiranju se mogu definirati u sljedeće dvije kategorije (Pitter, n.d.):

- ručno - provodi se osobno, klikom na aplikaciju ili interakcijom sa softverom i API-jima s odgovarajućim alatom.
- automatsko - izvodi stroj koji izvršava test skriptu koja je unaprijed napisana.

Implementacija

U ovoj fazi tim izdaje proizvod za radnu okolinu korisnika. Implementacija informacijskog sustava predstavlja stavljanje informacijskog sustava u korištenje. U novijoj se praksi koristi pristup kontinuirane integracije. Kontinuirana integracija (CI) je praksa automatizacije integracije promjena koda od više suradnika u jedan softverski projekt (Atlassian, n.d.).

Povratne informacije

Nakon puštanja proizvoda, posljednji korak je povratna informacija. Ovaj korak se izvodi za vrijeme razvoja informacijskog sustava i nakon implementacije informacijskog sustava.

2.2. Ekstremno programiranje

Agilni model razrađen u prošlom potpoglavlju, predstavlja temeljnu odrednicu agilnog programiranja. Ekstremno programiranje (XP) spada u domenu agilnih modela, uz precizniju definiciju i dodane značajke. Andres i Beck (2012) smatraju da je XP stil razvoja softvera usmjeren na izvrsnu primjenu programskih tehnika, jasnu komunikaciju i timski rad što omogućuje postizanje ciljeva koji su u prijašnjoj praksi bili nezamislivi. XP uključuje:

- način razvoja softvera temeljen na vrijednostima komunikacije, povratnih informacija, jednostavnosti, hrabrosti i poštovanja.
- skup praksi koje su se pokazale korisnima u poboljšanju razvoja softvera. Prakse se međusobno nadopunjuju, pojačavajući svoje učinke.
- skup komplementarnih principa, intelektualnih tehnika za interpretiranje što se pokazalo korisnim u situacijama kada ne postoji trenutna praksa određeni problem.
- zajednicu ljudi okupljenih oko istih vrijednosti i praksi.

2.3. Model vodopada

Model vodopada stekao je popularnost 1970-ih i 1980-ih, posebno u velikim i složenim projektima koji su zahtijevali visoku pouzdanost i kvalitetu, kao što su obrambeni, zrakoplovni i državni sustavi. Ovaj je model predstavljao je standardizirani model razvoja softvera u 20. stoljeću, sve do pojave agilnog programiranja (Sacolick, 2022).

“Kaskadni ili klasični model (vodopadni, engl. *Waterfall* model) razvoja slijedno je izvođenje aktivnosti i napredovanje iz faze u fazu. U njemu nisu dopuštene naknadne promjene rezultata prethodnih faza. Prikladan je za velike projekte (investicije) i za dobro definirano okruženje, gdje postoje propisane procedure ručne obradbe informacija ili računalni sustav, a rezultat je poznat iz niza sličnih ranije uspješnih projekata.„ (Pavlić, 2011:120).

Unatoč činjenici da je model vodopada zastarjelog karaktera uz sve manju primjenu, odlikuju ga prednosti koje se i danas mogu prevagnuti prilikom izbora optimalnog modela za planiranje IS-a. Model vodopada je vrlo jednostavan i razumljiv te je pogodan za programere početnike.

Karakterizira ga jednostavnost upravljanja projektima zbog rigidnosti modela. Štoviše, svaka faza ima specifične rezultate i pojedinačni proces pregleda. Model vodopada značajno štedi vrijeme u svim fazama koje se obrađuju i dovršavaju u određenom trenutku. Zahtjevi su precizno shvaćeni/definirani u ovome modelu, a samim time projekt ima unaprijed definiranu putanju s minimalnim odstupanjima (Hamilton, 2022).

Iako je uporaba ovog modela bila široko zastupljena i prihvatljiva u sferi informacijskih znanosti, razvojem informatike te sve većom potrebom za diverzifikacijom sustava, počinju se nadzirati nedostaci. Jedan od nedostataka je taj što se može koristiti isključivo u situacijama kada su zahtjevi precizni i dostupni unaprijed, što je u današnjoj praksi rijetka situacija. Ovaj model nije primjenjiv na projekte koji zahtijevaju kontinuirano održavanje, a navedeno je neizbježno u vremenu narušene informacijske sigurnosti. Ukoliko je aplikacija u fazi testiranja, nije preporučljivo vraćati se i raditi bilo kakve izmjene i dopune u softveru te se ovo smatra najvećim nedostatkom ovog modela. Ne postoji mogućnost razvoja radne verzije softvera dok se ne završi posljednja faza ciklusa, što je nepraktično u softverima koji zahtijevaju brzu isporuku radne verzije. Povratne informacije klijenata se ne mogu uključiti u faze razvoja te se na taj način (Hamilton, 2022).

Prilikom analiziranja agilnog modela i modela vodopada uočavaju se razlike koje mogu biti od presudne važnosti prilikom izbora odgovarajućeg modela. Tablica 1. prikazuje usporedbu modela i njihove razlike.

Tablica 1. Usporedba agilnog modela i modela vodopada

Agilni model	**Model vodopada**
Životni ciklus razvoja projekta podijeljen je u sprintove	Životni ciklus razvoja projekta podijeljen je u faze
Slijedi inkrementalni pristup	Slijedi sekvencijalni pristup
Model je fleksibilan	Model je strukturiran i rigidan
Razvoj softvera može biti podijeljen na više projekata	Razvoj softvera odvija se isključivo kao jedan projekt
Omogućuje promjene u zahtjevima razvoja projekata	Nema mogućnosti promjene u zahtjevima razvoja projekata

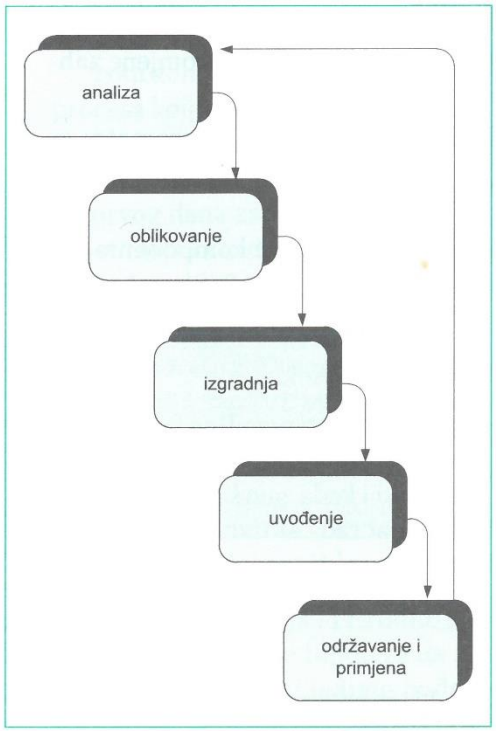
Planiranje, razvoj, izrada prototipova i druge faze razvoja softvera odvijaju se više puta i iterativno	Sve faze razvoja projekta poput projektiranja, razvoja, testiranja itd. dovršene su jednom u modelu
Testni plan se pregledava nakon svakog sprinta	Testni plan se rijetko pregledava u testnoj fazi
Model je pogodan za projekte u kojima se zahtjevi mijenjaju	Model je pogodan za projekte koji imaju određene zahtjeve i promjene koje se uopće ne očekuju
Testiranje se provodi paralelno s razvojem softvera	Faza „Testiranje“ dolazi nakon faze „Razvoj“
U centru je pozornosti način razmišljanja o proizvodu, fokus je na zahtjevima kupaca	U centru je pozornosti način razmišljanja o projektu, fokus je na ostvarenju projekta
Visok stupanj koordinacije i sinkronizacije tima	Ograničena koordinacija i sinkronizacija tima je
Vlasnik proizvoda s timom priprema zahtjeve gotovo svaki dan tijekom projekta.	Poslovna analiza priprema zahtjeve prije početka projekta.
Testni tim može bez problema sudjelovati u promjeni zahtjeva.	Testnom timu teško je pokrenuti bilo kakvu promjenu zahtjeva.
Opis detalja projekta može se promijeniti bilo kada tijekom SDLC procesa.	Detaljan opis treba implementirati vodopadni pristup razvoju softvera.
Članovi Agile tima su međusobno zamjenjivi, kao rezultat toga, rade brže. Također, nema potrebe za voditeljima projekta jer projektima upravlja cijeli tim.	U modelu vodopada proces je uvijek jednostavan pa voditelj projekta igra ključnu ulogu tijekom svake faze SDLC-a.

Izvor: Izrada autora, prema Hamilton (2023)

Kao što je otprije rečeno model vodopada je rigidan s točno utvrđenim procedurama. Pavlić (2011) navodi sljedeće procedure:

1. Analiza
2. Oblikovanje
3. Izgradnja
4. Uvođenje
5. Održavanje i primjena.

Na slici 1 prikazane su faze vodopadnog modela.



Slika 1. Vodopadni model

Izvor: Pavlić (2011:120)

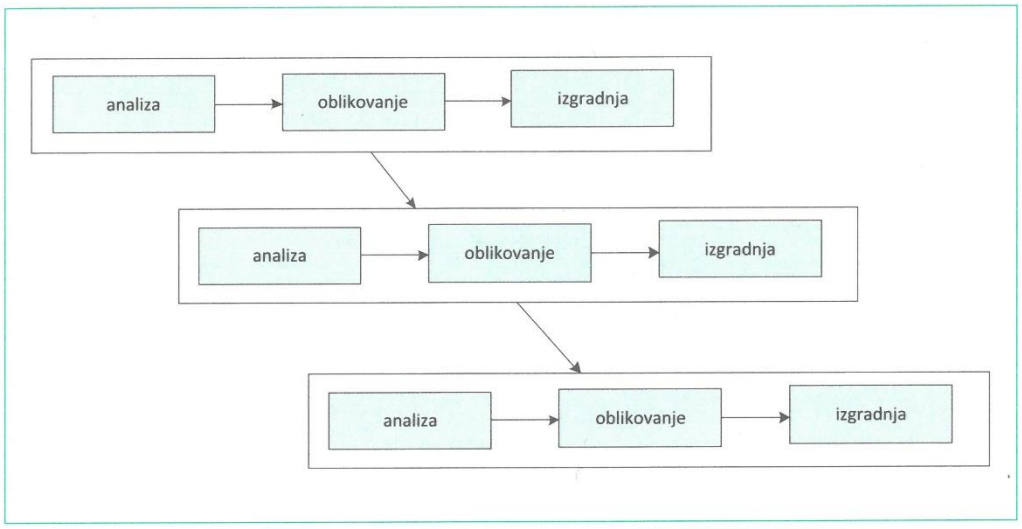
2.4. Evolucijski model

"Evolucijski je pristup niz paralelnih (ili s malim pomakom u fazi sljedova aktivnosti, takvih da svaki pojedini slijed vodi prema proizvodu koji se isporučuje korisniku." (Pavlić, 2011:124).

Evolucijski model prema Pavliću (2011) objedinjuje sljedeće procedure:

1. Analiza
2. Oblikovanje
3. Izgradnja

Slijed procedura evolucijskog modela prikazan je na sljedećoj slici.



Slika 2. Evolucijski model

Izvor: Pavlić (2011:125)

Evolucijski model se rijetko koristi, ali postoje situacije u kojima je njegova uporaba prihvatljiva, kao što je objektno orijentiran razvoj i u projektima pogodnim za inkrementalnu implementaciju (GeeksForGeeks, n.d.b).

2.5. Spiralni model

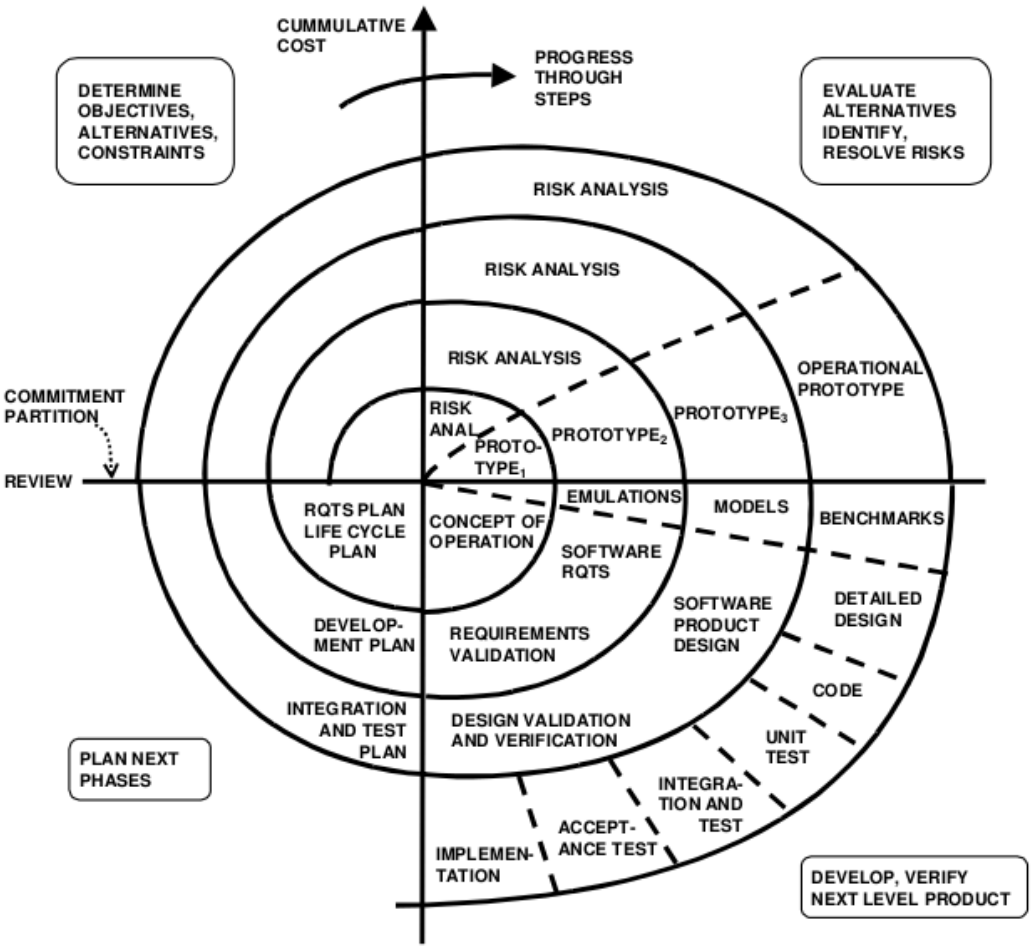
Razumijevanjem modela vodopada te uočavanjem njegovih nedostataka, javlja se potreba za razvojem novog modela. Barry Boehm 1986. godine razvija koncept spiralnog modela, koji je u svojoj srži spoj agilnog modela i modela vodopada (Kanjilal, 2023).

"Spiralni je model prijedlog odvijanja faza jedna za drugom u krug." (STSC, 1999). Jedan krug predočuje jednu fazu. Krug je izrezan u isječke, a svaki dio kruga čini jednu apstraktnu vrstu posla koja se izvodi u svakoj fazi. Svaki krug (svaka faza) ima sve apstraktne aktivnosti za jednu vrstu cilja koji želimo postići u projektu tom fazom. Tako prvi krug može označivati izradbu studije izvodljivosti, drugi krug može biti definicija korisničkih zahtjeva, treći krug može biti dizajn sustava, četvrti implementacija, peti uvođenje i integracija, a šesti održavanje. Krugovi u tom modelu predočuju faze ranijih modela. Svaki se krug sastoji od apstraktnih aktivnosti:

1. planiranje sljedeće faze,
2. određivanje ciljeva, alternativa i ograničenja,
3. procjena alternativa, analiza rizika,

4. izradba prototipa,
5. simulacija, modeliranje, mjerenje,
6. razvoj, verifikacija sljedeće razine proizvoda.

Te se aktivnosti provode u svakoj fazi. One su dodatak tog modela i svojevrstan metamodel oblikovanja aktivnosti svake pojedinačne faze (Pavlić, 2011:125). Navedene faze i aktivnosti najjednostavnije je predočiti dolje navedenom slikom.



Slika 3. Spiralni model

Izvor: Gurendo (2015)

2.6. Iterativni model

“U iterativnom modelu, iterativni proces počinje jednostavnom implementacijom malog skupa softverskih zahtjeva i iterativno unapređuje verzije koje se razvijaju sve dok cijeli sustav nije implementiran i spreman za implementaciju.

Iterativni model životnog ciklusa ne pokušava započeti s potpunom specifikacijom zahtjeva. Umjesto toga, razvoj počinje određivanjem i implementacijom samo dijela softvera, koji se zatim pregledava kako bi se identificirali daljnji zahtjevi. Ovaj se proces zatim ponavlja, stvarajući novu verziju softvera na kraju svake iteracije modela." (Tutorialspoint, n.d.).

Alshamrani i sur. (2015) smatraju da se iterativni model treba koristiti u sljedećim situacijama:

- u nisko do srednje rizičnim projektima
- u projektima dugotrajnog razvoja
- u projektima u kojima se koristi nova tehnologija
- u situacijama kada je razvoj projekta odjednom rizičan

Alshamrani i sur. (2015) navode sljedeće prednosti iterativnog modela:

- razvoj visoko rizičnih komponenata u ranim fazama izgradnje sustava
- rizik se raspoređuje na manje komponente umjesto na jednu veliku cjelinu
- korisnici u ranim fazama dobiju funkcionalnost te su u mogućosti odgovorit na svaku sljedeću nadopunu
- svako izdanje donosi operativni proizvod
- početna isporuka proizvoda je brža
- smanjuje se rizik od kvara i nastanka promjena

Nedostaci iterativnog modela su sljedeći (Alshamrani i sur., 2015):

- model zahtijeva precizno planiranje i dizajn
- zahtijeva rano definiranje cijelog i potpuno funkcionalnog sustava kako bi se funkcionalni sustav razvio u budućim iteracijama
- model ne dopušta ponavljanja unutar svakog inkrementa

3. Metode planiranja IS-a

Poput modela iz prijašnjeg poglavlja postoje i metode planiranja informacijskog sustava. Za razliku od modela, koji informacijski sustav razrađuju isključivo kao softversko rješenje, metode objedinjuju povezanost softvera i organizacije. Pažnja je posvećena metodama poput SWOT koja je jedna od prvih metoda, zatim jedna od najzastupljenijih SSADM te ARIS, MIRIS i Case method. Sve će metode biti objašnjene, uz prikaz njihovih glavnih karakteristika, kao i prednosti i mana te će biti uspoređene jedna s drugom.

3.1. SWOT

Ubrzanim razvojem informacijskih znanosti 60-tih godina prošlog stoljeća te globalne promjene u strukturama organizacija, došlo je do potrebe za pronalaskom optimalnih metodologija za planiranje IS-a. SWOT metodologija pojavljuje se kao odgovor za navedene probleme, nudeći istovremeno dugoročnu viziju i individualan pristup.

„SWOT matricom sučeljavaju se postojeće i buduće vanjske prijetnje i prilike s postojećim i budućim unutarnjim slabostima i snagama organizacije.“ (Panian i sur., 2010:219). SWOT predstavlja engleski akronim za snage (engl. *strenghts*), slabosti (engl. *weaknesses*), prilike (engl. *opportunities*) i prijetnje (engl. *threats*).

Snage i slabosti se odnose na čimbenike unutar organizacije i IS-a (British Library, n.d.). Snage, kao unutarnji čimbenik, predstavljaju inicijative koji pokazuju dobre rezultate te njihovo ispitivanje pomaže u razumijevanju komponenti koje funkcioniraju. Nakon razmatranja već funkcionalnih stavki, poželjno je osvrnuti se na stavke koje trebaju poboljšati učinkovitost. Slabosti se u SWOT-u odnose na interne inicijative koje nisu uspješne te identificiranje unutarnjih slabosti predstavlja početnu točku za poboljšanje tih projekata (Raeburn, 2022).

Prilike i prijetnje se odnose na vanjske čimbenike koji mogu imati utjecati na poslovanje. (British Library, n.d.). Prilike u SWOT-u proizlaze iz identificiranih postojećih snaga i slabosti, zajedno sa svim vanjskim inicijativama koje jačaju konkurentski položaj (Raeburn, 2022).

3.2. SSADM

Jedna od najzastupljenijih metoda planiranja informacijskog sustava je SSADM (engl. *Structured Systems Analysis and Design Method*). Pavlić (2011, citirano prema Longworth i Nicholls, 1989) zastupa mišljenje da "metodika SSADM povezuje velik broj (različit od verzije do verzije) iskušanih i provjerenih metoda na kompatibilan način, što osigurava dobru dokumentaciju".

"Osnovni princip SSADM metodologije jest pripadnost sustava korisniku pa je uloga korisnika u razvojnom procesu esencijalna. Korisnik je konzultiran prigodom prikupljanja činjenica, za potvrdu potpunosti, kao i za ispravnost niza produkata na različitim razinama razvoja SSADAM-a (Pavlić, 2011:285).

Rouse (2011) navodi tehnike koje se koriste u SSADM-u, a to su modeliranje logičkih podataka, modeliranje protoka podataka i modeliranje ponašanja entiteta. Modeliranje logičkih podataka odnosi se na proces identificiranja, modeliranja i dokumentiranja podataka kao dio prikupljanja zahtjeva sustava. Modeliranje protoka podataka obuhvaća praćenje protoka podataka u informacijskom sustavu uz precizno analiziranje procesa, pohrane podataka, vanjskih entiteta kretanja podataka. Modeliranje ponašanja entiteta predstavlja identificiranje i dokumentiranje događaja koji imaju utjecaj na svaki entitet i slijed u kojem se navedeni događaji odvijaju.

3.3. ARIS

August-Wilhelm Scheer 1990-ih predstavlja metodu ARIS, engleski akronim od *Architecture of Integrated Information Systems* (Rouse, 2015).

Sheer (1999) smatra da metoda ARIS primarno pomaže u uočavanju širokog spektra deskriptivnih aspekata poslovnih procesa, dodjeljivanja metoda njima, određivanju bilo kojeg preklapanja metoda i identificiranju nezauzetih polja opisa. ARIS pruža prednosti za rješavanje administrativnih ili organizacijskih problema u poslovanju, kao i za računalno potpomognuto inženjerstvo informacijskih sustava.

Koncept ARIS-a dizajniran je tako da uključuje integrirane metode i alate (ARIS *Toolset*) kako bi se iskoristili ovi pristupi, ubrzala implementacija i smanjio napor implementacije standardnog softvera (Sheer, 1999).

ARIS funkcionalnost vidljiva je u sljedećim značajkama (Sheer, 1999):

- sadrži okvir (arhitekturu) za potpuni opis standardnih softverskih rješenja,
- integrira najprikladnije metode za modeliranje informacijskih sustava u arhitekturu, razvija metode za opisivanje poslovnih procesa i događaja,
- omogućuje referentne modele kao alate za administraciju znanja i iskustva aplikacija, za modeliranje i analizu zahtjeva sustava, te za osiguravanje korisniku prilagođene navigacije unutar samog modela.
- koristeći standardna softverska rješenja, ARIS objedinjuje arhitekturu za upravljanje poslovnim procesima. Koristeći sustave tijeka rada povezan je sa softverskim "blokovima" (poslovnim objektima). ARIS pruža okvir za opisivanje sklapanja softverskih komponenti, što rezultira poslovnim informacijskim sustavom koji je idealan za konfiguriranje sustava tijeka rada i određivanje parametara za aplikacije.

3.4. MIRIS

Specijalizirana metodologija MIRIS (skraćeno od Metodologija za Razvoj Informacijskog Sustava) skup je metoda i uputa čiji je ukupni cilj projektirati i izgraditi informacijski sustav. Njezino je oblikovanje započelo 1984. godine, a objavljena je 1995. godine (Pavlić, 1995). Poznata je kao metodika MIRIS. Ta specijalizirana metodologija propisuje faze razvoja i aktivnosti pojedine faze do potrebne razine detalja informacijskih sustava.

Osnovna hipoteza oko koje je MIRIS oblikovan jest „dekompozicija sustava”, a glasi (Pavlić, 2011):

- životni ciklus projektiranja podijeliti u tri faze
- u prvoj fazi apstraktno modelirati cijeli sustav i u tom modelu sustav podijeliti u podsustave.
- u drugoj fazi modelirati podsustav i propisati odgovarajuću metodu za modeliranje procesa
- u trećoj fazi modelirati podatke relevantnih procesa i definirati arhitekturu aplikacije prema modelu procesa i modelu podataka.

Predloženi životni ciklus metodologije MIRIS je „V“ model, i to tako da su na lijevoj strani faze kojima se dekomponira poslovni sustav (dok se kod uobičajenog “V” modela dekomponira proces modeliranja sustava), a na desnoj se strani predlaže linearan razvoj i

uvođenje aplikacija neovisno o složenosti (dok se kod "V" modela izvodi testiranje i okrupnjivanje modula u veće proizvode) (Pavlić, 2011).

MIRIS se koristi trima osnovnim metodama: metodom za modeliranje podataka, metodom za modeliranje procesa i metodom za modeliranje aplikacija. Metode su slične mnogobrojnim metodama u drugim metodologijama.

Faze životnih ciklusa grupirane su u dvije skupine faza: logičko oblikovanje (projektiranje IS) i fizičko oblikovanje (izgradnja IS). Svaka skupina faza ima tri faze. Faze se dalje dijele u aktivnosti (Pavlić, 2011:286).

3.5. CASE METHOD

"Metode imaju odgovarajući softver koji omogućuje unos ulaznih podataka u bazu znanja o sustavu i kreaciju izlaznih modela. Takve softverske proizvode nazivaju se CASE (engl. *Computer-Aided Systems Engineering*) alatima (CASE sustavima). CASE znači računalno potpomognuto inženjerstvo sustava." (Pavlić, 2011:273). Tijekom 1990-ih, „CASE alati" postaju dio softverskog leksikona, a velike tvrtke poput IBM-a koristile su ove vrste alata za pomoć u stvaranju softvera (GeeksForGeeks, n.d.a).

CASE alati klasificiraju se u tri klase (Pavlić, 2011:272):

- Upper CASE - pomažu u procesu projektiranja i dokumentiranje modela informacijskog sustava.
- Lower CASE - podržavaju izgradnju i stvaranje softvera (automatsko programiranje)
- Full CASE - podržavaju cjelokupni razvojni proces i dizajn te izgradnju i upravljanje procesom razvoja

Pavlić (2011) smatra da je jedan od ciljeva ove metodologije povećanje produktivnosti razvoja informacijskog sustava te da se navedeni ciljevi u fazi dizajna postižu implementacijom metodologije, a u fazi izgradnje uporabom softverskih alata koji automatski generiraju izvorni kod. U slučaju kada CASE alat ne generira izvršni kod, obično ne povećava produktivnost razvojnog ciklusa u fazi izgradnje informacijskog sustava. Full CASE alat (generator izvornog koda) povećava produktivnost te je poželjan kao alat za izgradnju informacijskog sustava.

4. Kriteriji izbora odgovarajućeg modela i m izgradnje IS-a

Nakon definiranih i analiziranih modela i metoda, sljedeći je korak odabir prikladnih kriterija za planiranje informacijskog sustava. Kako bi se izabrali najbolji model i metoda za izgradnju nekog informacijskog sustava moraju se poštovati i zadovoljiti određeni kriteriji koje postavlja organizacija. Kroz literaturu i praksu razmatraju se sljedeći kriteriji (fullscale, 2023):

Projektni zahtjevi i složenost – prvi korak za razvijanje uspješnog informacijskog sustava predstavlja razumijevanje prirode i opsega projekta. Jedan od prioritetnih kriterija bi trebala biti sama veličina i složenost informacijskog sustava jer oni igraju ključnu ulogu u postavljanju funkcionalnosti i strukturi samog sustava. Složenost informacijskog sustava ima bitan utjecaj na hardverske i softverske komponente arhitekture.

Veličina tima i iskustvo – u praksi predstavljaju glavni faktor. Veličina tima igra ulogu u raspoređivanju resursa, podjeli zadataka, hijerarhiji i odgovornosti. S druge strane, iskustvo je ključno u odabiru pristupa i pogleda na razvoj softvera. Programeri s višegodišnjim iskustvom bolje procjenjuju vremenske rokove, rizike te se lakše snalaze u rutinskim zadatcima.

Uključivanje kupaca i povratne informacije - razmatra razinu uključenosti korisnika i povratne informacije potrebne tijekom procesa razvoja. Ovim se kriterijem stavlja naglasak na razumijevanje i uvažavanje potreba na tržištu.

Vremenska ograničenja i rokovi - uveliko utječu na izbor metodologije. Vrijeme ovisi o čimbenicima poput organizacije tima, stupnja znanja pojedinih osoba u timu, poteškoće u implementaciji sustava i mnogim drugim. Kako ne bi došlo do konflikata i greške u komunikaciji i razilaženju u očekivanom trajanju projekta, svaka bi organizacija na samom početku projekta trebala postaviti vremenski rok do kojeg softver, odnosno informacijski sustav treba biti isporučen, ali u isto vrijeme trebaju biti svjesne da je prekoračenje vremenskog roka relativno često u praksi. Stoga je u ovom aspektu potrebno poslušati zaposlenike pri njihovoj procjeni koliko je taj rok moguće ispoštovati kako bi odmah u početku bilo jasno može li se projekt isporučiti do očekivanog datuma ili ne.

Tolerancija na rizik - procjena tolerancije organizacije na rizik i potencijalni učinak neuspjeha projekta. Rizici izgradnje informacijskog sustava i sigurnost predstavljaju važan kriterij u doba internetske nesigurnosti, neovlaštenih pristupa sustavima i informacijama te

ostalim aspektima računalnog kriminala. Razumijevanjem rizika mogu razviti strategije za smanjenje rizika i planove za nepredviđene situacije kako bi se minimizirao njihov utjecaj na projekt. Ovaj proaktivni pristup pomaže u upravljanju potencijalnim preprekama i osigurava fluidniji proces razvoja (Khakhkhar, 2023). S druge strane, neuspjeh projekta odražava u vidu smanjenih financijskih resursa i demotiviranosti tima. što može dodatno otežati razvoj softvera.

5. Rasprava

Za izgradnju složenih informacijskih sustava preporučuje se koristiti agilni model jer ima veću sklonost dodavanja novih funkcija i promjena od modela vodopada koji je nepromjenjivog karaktera. Agilni model također ima mogućnost rastavljanja projekta na sprintove te se tako složeni projekt može uspješno riješiti. Navedeni stav potrebno je proučiti i iz perspektive projektnih zahtjeva. Ukoliko su zahtjevi jasno i precizno definirani s stabilnim i dostupnim informacijama moguće je koristiti agilni i vodopadni model. Vodopadni model je u ovoj situaciji prikladan jer zahtijeva dobro utvrđene zahtjeve i dokumentaciju. U situaciji kada zahtjevi nisu precizno definirani, izvori podataka i informacija su ograničeni preporuka je koristiti agilni model odnosno ekstremno programiranje zbog sklonosti promjenama. Kada su informacije izrazito ograničene te se dolazi do njih, preporuka je također koristit agilni model ili ekstremno programiranje.

Nadalje, agilni model i ekstremno programiranje pogodni su u projektima koje zahtijevaju visok stupanj koordinacije, sinkronizacije i dobre komunikacije. Također, u ovim se modelima naglašava potreba inovativnosti i fleksibilnosti članova. S druge strane, model vodopada ima strožu podjelu zadataka, manje inovativnosti među članovima tima i slabiju sinkronizaciju i koordinaciju. Iterativni i spiralni model imaju veću razinu inovativnosti i fleksibilnosti članova u odnosu na model vodopada, ali manju u odnosu na agilni model i ekstremno programiranje. SWOT matrica i SSADM metoda MIRIS pogodne je za timski rad i pronalaženje zajedničkih rješenja, za razliku od CASE METHOD i ARIS koje imaju utvrđene principe i korake.

Naglasak na povratnim informacijama i uključenost kupaca se očituje u agilnom modelu i ekstremnom programiranju i iterativnom modelu. Vodopadni model je orijentiran na ispunjavanje ciljeva bez razmišljanja o svim potrebama krajnjih korisnika. SWOT i SSADM predstavljaju pogodne metode kada su želje i potrebe kupaca na prvom mjestu.

Vremensko ograničenje i prekoračenje istog može biti problem u agilnom modelu i ekstremnom programiranju jer u situacijama kada zahtjevi nisu dovoljno dobro definirani projekt lagano izgubi putanju. Vodopadni i spiralni model zbog svojih već utvrđenih principa teže ispunjavanju vremenskih rokova. CASE METHOD zbog dobro definiranih faza i mogućnosti automatskog programiranja značajno štedi vrijeme, a samim time i resurse.

Zbog svojih zastarjelih principa i rigidnosti preporuka izbjegavati model vodopada, evolucijski, spiralni i iterativni model, pogotovo kada su sigurnosni zahtjevi i rizici visoki.

Agilni model i ekstremno programiranje mogu se rabiti u visokorizičnim projektima, upravo zbog mogućnosti nadogradnje sustava i povećanja zahtjeva. SSADM i ARIS metode zbog široke rasprostranjenosti nude odgovarajuća sigurnosna rješenja.

6. Zaključak

Informacijski sustav omogućuje organizaciji stratešku prednost u konkurentskoj okolini, utjecajem na operativnu efikasnost poslovanja i pokretanjem inovativnosti. Shvaćanjem važnosti informacijskog sustava za organizaciju, stavlja se fokus na strateško planiranje informacijskog sustava. Prije razmatranja strateškog planiranja izgradnje informacijskog sustava, najprije je potrebno definirati informacijski sustav i pripadajuće dijelove sustava. Za strateško planiranje informacijskog sustava koriste se modeli, metode i kriteriji izbora. Modeli opisani u ovome radu su: agilni model, ekstremno programiranje, vodopadni model, evolucijski, spiralni te iterativni model. Svaki od navedenih modela ima različiti pristup razvoju i implementaciji sustava te prednosti i nedostatke. Metode koje se koriste u svrhu strateškog planiranja su SWOT, SSADM, ARIS, MIRIS i CASE METHOD. Metode se međusobno više razlikuju nego modeli te je stoga izbor metode kompliciraniji. Kako bi organizacije odabrale optimalni model i metode trebaju postaviti kriterije izbora, od kojih se ističu: projektni zahtjevi i složenost, veličina tima i iskustvo, uključivanje kupaca i povratne informacije, vremenska ograničenja i tolerancija na rizik. Kriteriji su izbora često suprotnog smjera, stoga informiranost o kriterijima uvelike smanjuje mogućnost odabira pogrešnog modela i metode.

Literatura

1. Ackoff, R. L. (1989). From Data to Wisdom. Journal of Applied Systems Analysis, Vol. 16.
2. Agilni manifest. (n.d.). Dostupno na: https://agilemanifesto.org/principles.html [Pristupljeno 10.7.2023.]
3. Alshamrani, A., Bahattab, A., Fulton, I. A. (2015). A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model. IJCSI International Journal of Computer Science Issues, Volume 12, Issue 1, No 1. Dostupno na: https://www.ijcsi.org/papers/IJCSI-12-1-1-106-111.pdf [Pristupljeno: 10.6.2023.]
4. Andres, C., Beck, K. (2012). Extreme Programming Explained (2. izdanje). Pearson Education. Dostupno na: https://ptgmedia.pearsoncmg.com/images/9780321278654/samplepages/9780321278654.pdf [Pristupljeno: 10.6.2023.]
5. Atlassian (n.d.). What is continous integration?. Dostupno na: https://www.atlassian.com/continuous-delivery/continuous-integration [Pristupljeno: 20.8.2023.]
6. Bardon, C. (n.d.). Information system architecture, from methodology to implementation tools. Blueway. Dostupno na: https://www.blueway.fr/en/blog/information-system-architecture-from-methodology-to-implementation-tools [Pristupljeno 20.6.2023.]
7. Beck, K., Beedle, M., Bennekum, A. v., Cockburn, A., Cunningham, W., Fowler, M., . . . Thomas, D. (2001). Agile Manifesto. Dostupno na https://agilemanifesto.org/ [Pristupljeno 11.7.2023.]
8. Bellinger G., Castro, D., Mills, A. (2004). Data, Information, Knowledge, and Wisdom. Dostupno na: https://www.systems-thinking.org/dikw/dikw.htm [Pristupljeno 20.6.2023]
9. British Library (n.d.). What is SWOT analysis?. Dostupno na: https://www.bl.uk/business-and-ip-centre/articles/what-is-swot-analysis [Pristupljeno 20.6.2023.]
10. Enciklopedija.hr (n.d.). Agilnost. Dostupno na: https://www.enciklopedija.hr/natuknica.aspx?id=784 [Pristupljeno 20.8.2023.]

11. fullscale (2023). SOFTWARE DEVELOPMENT METHODOLOGIES: CHOOSE AMONG THE TOP 10. Dostupno na: https://fullscale.io/blog/top-10-software-development-methodologies/ [Pristupljeno 4.9.2023.]
12. GeeksForGeeks (n.d.a). Components of Information System. Dostupno na: https://www.geeksforgeeks.org/components-of-information-system/#article-meta-div [Pristupljeno 20.8.2023.]
13. GeeksForGeeks (n.d.a). Computer Aided Software Engineering (CASE). Dostupno na: https://www.geeksforgeeks.org/computer-aided-software-engineering-case/ [Pristupljeno 20.8.2023.]
14. GeeksForGeeks (n.d.b). Software Engineering | Evolutionary Model. Dostupno na: https://www.geeksforgeeks.org/software-engineering-evolutionary-model/ [Pristupljeno 17.6.2023.]
15. GeeksForGeeks (2023). Unified Modeling Language (UML) | An Introduction. Dostupno na: https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/ [Pristupljeno 17.6.2023.]
16. Gurendo, D. (2015). Software Development Life Cycle (SDLC). Spiral Model. XB Sofware. Dostupno na: https://xbsoftware.com/blog/software-development-life-cycle-spiral-model [Pristupljeno: 10.6.2023.]
17. Hamilton, T. (2023). Agile vs Waterfall – Difference Between Methodologies. Guru99. Dostupno na: https://www.guru99.com/waterfall-vs-agile.html [Pristupljeno:11.6.2023]
18. Indeed (2023). 12 Techniques for Requirement Gathering. Dostupno na: https://www.indeed.com/career-advice/career-development/requirement-gathering-techniques [Pristupljeno 10.7.2023.]
19. Javatpoint (n.d.). Agile model. Dostupno na: https://www.javatpoint.com/software-engineering-agile-model [Pristupljeno: 10.6.2023.]
20. Jena, S. (n.d.). Fundamentals of Software Architecture. GeeksforGeeks. Dostupno na: https://www.geeksforgeeks.org/fundamentals-of-software-architecture/ [Pristupljeno 12.6.2023.]
21. Kanjilal, J. (2023). Overview of Spiral Software Development. Developer.com. Dostupno na: https://www.developer.com/project-management/spiral-software-development/ [Pristupljeno 12.6.2023.]
22. Khakhkhar K. (2023). A Guide to Choosing the Right Software Development Methodologies for Your Next Project. Dostupno na: A Guide to Choosing the Right

Software Development Methodologies for Your Next Project. Dostupno na: https://www.peerbits.com/blog/how-to-choose-software-development-methodology.html [Pristupljeno: 4.9.2023.]

23. Longworth, G., Nicholls, D. (1989). SSADM MANUAL Techniques and documentation, National Computing Centre, England.
24. Packt (2018). What is a multi layered software architecture? Dostupno na: https://hub.packtpub.com/what-is-multi-layered-software-architecture/ [Pristupljeno 11.7.2023.]
25. Panian, Ž., Ćurko, K., Bosilj, V., Čerić, V., Pejić, M., Požgaj, Ž., Spremić, M., Strugar, I., Varga, M. (2010). Poslovni informacijski sustavi.
26. Pavlić, M. (1995). *Tehnologija projektiranja informacijskih sustava – MIRIS, CASE*, Opatija
27. Pavlić, M. (2011). Informacijski sustavi. Zagreb: Školska knjiga.
28. Pavlić, M., Jakupović, A., Čandrlić, S. (2014). Modeliranje procesa. Rijeka: Odjel za informatiku Sveučilišta u Rijeci.
29. Pitter, S. (n.d.). The different types of software testing. Atlassian. Dostupno na: https://www.atlassian.com/continuous-delivery/software-testing/types-of-software-testing [Pristupljeno 11.7.2023.]
30. Raeburn, A. (2022). Extreme programming (XP) gets results, but is it right for you? Asana. Dostupno na: https://asana.com/resources/extreme-programming-xp [Pristupljeno:13.6.2023]
31. Rouse, M. (2011). Structured Systems Analysis And Design Method. Technopedia. Dostupno na: https://www.techopedia.com/definition/3983/structured-systems-analysis-and-design-method-ssadm [Pristupljeno:13.6.2023]
32. Rouse, M. (2015). Architecture of Integrated Information Systems. Technopedia. Dostupno na: https://www.techopedia.com/definition/30659/architecture-of-integrated-information-systems-aris [Pristupljeno 14.7.2023.]
33. Rouse, M. (2017). Integrated Development Environment. Dostupno na: https://www.techopedia.com/definition/26860/integrated-development-environment-ide [Pristupljeno 20.8.2023.]
34. Sacolick, I. (2022). A brief history of the agile methodology. InfoWorld. Dostupno na: https://www.infoworld.com/article/3655646/a-brief-history-of-the-agile-methodology.html [Pristupljeno 14.6.2023.]

35. STSC (1999). A Gentle Introduction To Software Engineering, United States Air Force, Utah
36. Sheer, A. W. (1999). ARIS - Business Process Frameworks (3. izdanje). Dostupno na: https://books.google.hr/books?hl=hr&lr=&id=u2qLJh-tamAC&oi=fnd&pg=PA1&dq=aris+methodology&ots=OCkboHWrGB&sig=EC_QrXE5RkoW_x6bfW92Eh42es8&redir_esc=y#v=onepage&q=aris%20methodology&f=false [Pristupljeno:17.6.2023]
37. Šimović, V. (2009). Uvod u informacijske sustave. Golden marketing – Tehnička knjiga.
38. Tutorialspoint (n.d.). SDLC - Iterative Model. Dostupno na: https://www.tutorialspoint.com/sdlc/sdlc_iterative_model.htm [Pristupljeno 15.6.2023]
39. TechTarget (2023). Information Systems (IS). TechTarget. Dostupno na: https://www.techtarget.com/whatis/definition/IS-information-system-or-information-services [Pristupljeno 13.7.2023.]
40. Wong, C. (2020). Multilayered Software Architecture. Medium. Dostupno na: https://medium.com/@e0324913/multilayered-software-architecture-1eaa97b8f49e [Pristupljeno 20.8.2023.]

Popis slika

Popis tablica