7. PRIMENJENA INFORMATIKA U OBLASTI ZŽS

7.1. Informatika u zaštiti životne sredine

Tokom proteklih nekoliko decenija ubrzani razvoj informacionih tehnologija (IT) omogućio je interdisciplinarno povezivanje istraživača u oblastima zaštite životne sredine i softverskog inženjerstva. Na osnovima tog interdisciplinarnog povezivanja došlo je do razvoja nove naučne discipline, ekološke informatike (engl. *Environmental Informatics*), koja obuhvata istraživanja u širokom dijapazonu oblasti, kao što su primena klasičnih informacionih sistema i baza podataka u oblasti zaštite životne sredine, geografski informacioni sistemi (GIS), menadžerski informacioni sistemi u zaštiti životne sredine, akvizicija podataka, modelovanje i simulacija, vizualizacija i veštačka inteligencija (engl. *Artificial Intelligence*, AI). Ova disciplina se nalazi između softverskog inženjerstva i inženjerstva zaštite životne sredine, i doprinosi novim saznanjima u oblasti zaštite životne sredine primenom softverskih tehnika.

Informacioni sistemi koji se primenjuju u zaštiti životne sredine imaju višestruke funkcije, uključujući monitoring i kontrolu, upravljanje i analizu prikupljenih podataka, kao i u oblastima planiranja i podrške odlučivanju. Primena informacionih tehnika igra ključnu ulogu u planiranju, predviđanjima, monitoringu i kontroli procesa koji se odvijaju u životnoj sredini, a dovela je i do značajnih otkrića, koja su imala odjeka i u širokoj zajednici, npr. omogućila su sagledavanje antropogenog uticaja na klimatske promene. Paralelno sa primenom informacionih sistema, različite institucije, organizacije i vlade širom sveta počele su da zauzimaju proaktivan stav u kontroli zagađenja prirode uvođenjem odgovarajuće zakonske regulative i ekološkim standardima, na primer ISO 14001, Evropski

EMAS standard, američki Nacionalni akt o politici zaštite životne sredine (*National Environmental Policy Act*, NEPA) i Kjoto Protokol.

Informacione i komunikacione tehnologije (engl. *Information and Communication Technology*, ICT) koje se koriste u istraživanjima i monitoringu u oblasti zaštite životne sredine, kao i prilikom donošenja odluka i obaveštavanja javnosti, mogu se podeliti u četiri velike grupe: (1) elektronika i mikrosistemi, (2) informacioni sistemi i softver, (3) mediji i sadržaj i (4) komunikacione tehnologije i mreže (Tabela 7.1). Predmet daljeg razmatranja biće prevashodno druga od navedenih grupa.

Tabela 7.1. Klasifikacija ICT sredstava

Katerorija	Sredstva
Elektronika i mikrosistemi	Automatizacija, Robotika, Senzori, Sistemi za kontrolu i monitoring
Informacioni sistemi i softver	Baze podataka, Upravljanje podacima/znanjem/procesima, Analitika i obrada podataka, Simulacija, Veštačka inteligencija, Podrška odlučivanju
Mediji i sadržaj	Izdavaštvo i diseminacija informacija, Izbor/semantika/statistika informacija, Vizualizacija, Geografski informacioni sistemi (GIS)
Komunikacione tehnologije i mreže	Audiovizuelna oprema i komunikacione tehnologija, Širokopojasni Internet, Internet i Web servisi, Mobilne komunikacije, Mrežne tehnologije, Portalne tehnologije, Satelitski sistemi/pozicioniranje/komunikacije

7.2. Osnove informacionih sistema

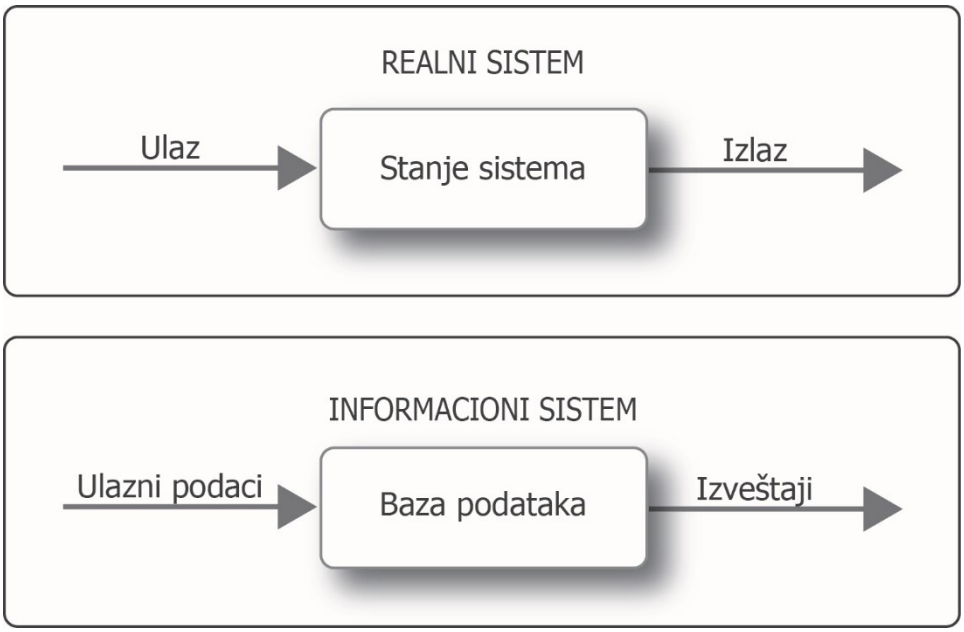
Uopšteno govoreći, informacioni sistemi imaju za cilj da se preko njih sagleda stanje određenog realnog sistema, da bi se na osnovu toga donele odluke, koje će taj realni sistem dovesti do ispunjenja njegove svrhe. Obuhvat, veličina i priroda posmatranog realnog sistema, koji se generalno definiše kao skup objekata i njihovih međusobnih veza, može biti veoma različita. Objekti koji sačinjavaju realni sistem mogu uključivati fizičke objekte, kao što su proizvodna postrojenja i oprema, osobine iz okruženja uključujući i životnu sredinu, razne vrste procesa i događaja, ali i dokumente i njihove delove, apstraktne koncepte, pa čak i elemente softvera i njegovog interfejsa prema korisnicima ili drugim programima. Karakteristike objekata relevantne sa stanovišta realnog sistema se nazivaju atributi.

Za svaki realni sistem postoje granice, koje definišu koji objekti pripadaju sistemu, a sve van toga se naziva okolina sistema. Dejstvo okoline na sistem predstavlja ulaz u sistem, a dejstvo sistema na okolinu predstavlja izlaz sistema. Ulazi u sistem menjaju stanje sistema, a promena stanja sistema se reflektuje na izlaz preko izlazne transformacije. Stanje sistema predstavlja njegovu fundamentalnu karakteristiku, pošta ga sačinjava skup objekata, njihovih međusobnih veza i skup vrednosti atributa objekata u nekom trenutku vremena i omogućava nam da, poznavajući prošla i sadašnje stanje, odredimo kakvi će biti budući izlazi pod dejstvom poznatih ulaza.

Iako su informacioni sistemi (IS) postojali još od početka ljudske civilizacije i od svog nastanka koristili različita tehnička sredstva, kao na primer pismo, oni su doživeli posebno snažan i brz razvoj kada su se kao njihova tehnička sredstva pojavili računarski sistemi. Dakle, iako IS teorijski mogu biti realizovani upotrebom bilo kog sredstva, danas se podrazumeva njihova realizacija putem računarskih sistema, koji, zahvaljujući sistemima unutrašnjeg zapisivanja podataka i naredbi za rad sa njima, omogućavaju automatizovanu obradu podataka. Nadalje će biti reč isključivo o ovakvim sistemima.

Kod IS zasnovanih na korišćenju računara mogu se uočiti dve odvojene faze. U prvoj, ili razvojnoj fazi, sva pravila i transformacije nad podacima, kao i sama struktura podataka, kojima se modeluje realni sistem, pretvaraju se u računarski čitljiv vid, odnosno izvršne programe, koji se potom smeštaju unutar računarskog sistema. Ovi programi sadrže strukture podataka i algoritme koji definišu način obrade podataka u budućem informacionom sistemu, ali još uvek ne i same podatke koji opisuju stanje realnog sistema. Tek u drugoj, ili eksploatacionoj fazi informacionog sistema, izvršavanjem programa na procesoru računara, podaci iz realnog sistema se unose, skladište, obrađuju i po potrebi prikazuju.

Pošto IS predstavlja model odgovarajućeg realnog sistema zasnovan na informacijama, odnosno podacima, posebno važan deo, a može se reći i osnovu IS čini baza podataka, kao skup računarskih programa za upravljanje, čuvanje i obradu podataka. U fazi razvoja IS, baza podataka mora biti tako projektovana, odnosno strukturirana, da dobro odražava objekte u realnom sistemu, njihove atribute i međusobne veze, jer će samo na taj način IS moći verno modelovati stanje realnog sistema. Kao i kod realnog sistema, i IS ima ulaz, koji čine ulazni podaci, koji prolaze kroz programe za održavanje baze podataka, kao i izlaz, koji sačinjavaju izlazni podaci dobijeni posredstvom programa za izveštavanje, slika 7.1.



Slika 7.1. Odnos realnog i informacionog sistema

Adekvatno projektovanje baze podataka je dakle ključno za uspešno funkcionisanje IS: ukoliko baza podataka predstavlja dobar model realnog sistema i ukoliko programi za održavanje dobro modeluju dejstvo ulaza na stanje realnog sistema, onda će se iz IS moći dobiti bilo koje informacije potrebne za upravljanje realnim sistemom. Intelektualni alati kojima se to postiže, odnosno kojima se modeluje sistem kao skup objekata, njihovih atributa i međusobnih veza, nazivaju se modeli podataka.

Postoji veći broj modela podataka, od kojih je danas najšire primenjivan Model Objekti-Veze (engl. *Entity Relationship Model*, ER). Detaljan prikaz modela podataka izlazi iz okvira ove knjige i može se naći u literaturi, na primer u knjizi Lazarevića i saradnika, datoj u referencama na kraju ovog poglavlja. Važno je međutim napomenuti da poznavanje modela podataka i posebno ER modelovanja može biti od velike pomoći i inženjerima i ekspertima koji nisu specijalisti za informatiku u definisanju IS, zahteva koji se pred IS postavljaju, očekivanih rezultata, kao i u praćenju njihovog razvoja i eksploatacije.

7.2.1. Tehničko-ekonomske specifičnosti informacionih sistema

U prethodnih nekoliko decenija došlo je do velike ekspanzije informacionih tehnologija (IT), kroz eksponencijalni rast snage i kapaciteta hardvera, koji je pratio takođe eksponencijalni rast broja korisnika informacionih tehnologija i Interneta. Zbog širokog uticaja na privredu i društvo u celini, ovaj razvoj označen je kao treća tehnološka ili digitalna revolucija (posle prve, karakterisane početnom mehanizacijom i parnom mašinom, i druge, obeležene masovnom proizvodnjom i električnom strujom), a danas je u nastanku četvrta, koja se zasniva na kibernetsko-fizičkim sistemima, i u kojoj se može očekivati dalji eksponencijalni rast broja korisnika IT, s obzirom da će IT početi sve više da aktivno koriste mašine, pa čak i proizvodi.

Nažalost, pri razvoju i uvođenju IS, stvari su se razvijale drugom, značajno sporijom dinamikom, pri čemu su uočena dva važna problema.

Prvi problem poznat je pod imenom "kriza softvera" i ukazuje na nemogućnost profesionalaca koji se bave razvojem softvera da pravovremeno i u potpunosti iskoriste novonarasle hardverske potencijale. Iako je činjenica da se na globalnom nivou povećava ukupno raspoloživa količina softvera, to je pre svega posledica rasta broja programera u svetu, dakle ekstenzivno, pošto je prosečan godišnji porast produktivnosti u proizvodnji softvera u ukupnom proteklom periodu iznosio svega nekoliko procenata. Situacija se samo pogoršala razvojem Interneta i na njemu zasnovane digitalne ekonomije, koja je značajno povećala tražnju za softverom, a najočitija posledica opisane krize jeste dugotrajni manjak stručnjaka u oblasti softvera na celokupnom svetskom tržištu.

Drugi problem se odnosi na mogućnost informatičkih i softverskih profesionalaca da razumeju realni sistem, koji bi trebalo da pretoče u korisnički softver, odnosno informacioni sistem. Naime, prema podacima Ministarstva odbrane SAD, još sredinom osamdesetih godina je samo 2% informacionih sistema odmah nakon isporuke softvera zadovoljilo stvarne potrebe korisnika, a još 3% razvijenog aplikativnog softvera moglo je biti korišćeno posle naknadne dorade. Svi ostali projekti se mogu smatrati manje ili više neuspešnim, pri čemu je 19% odbačeno kao neupotrebljivo ili je moralo biti u znatnoj meri preprojektovano, 29% ugovora nije dalo nikakav rezultat, a u čak 47% slučajeva softver je isporučen, ali nikad nije korišćen, pošto nije na zadovoljavajući način rešio realni problem. Situacija se donekle popravila razvojem informatike i softverskog inženjerstva u prethodne tri decenije, ali se još uvek procenat potpuno uspešnih projekata razvoja IS kreće oko 30%, odnosno varira između 25 i 34%, u zavisnosti od istraživanja.

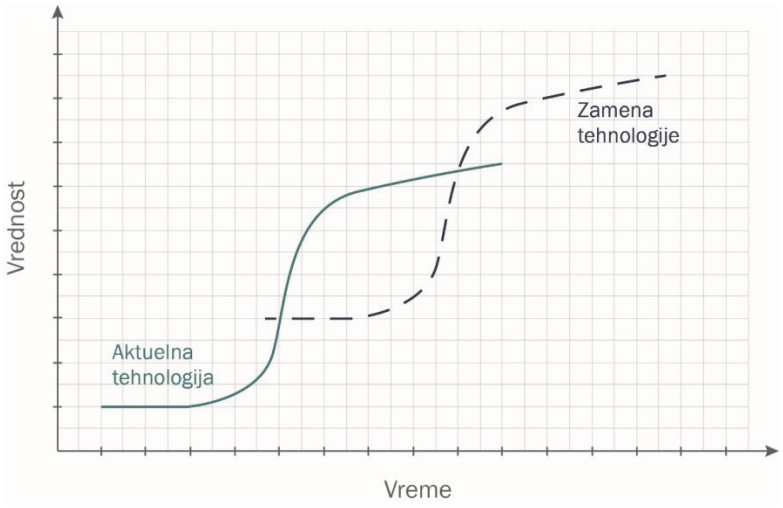
Uzroci ovakvog stanja su veoma brojni i složeni, ali može se reći da u najvažnije spadaju nedovoljno razumevanje životnog ciklusa softvera, posebno njegovih inicijalnih i kritičnih faza konceptualizacije i projektovanja, kao i nepoznavanje osnovnih tehno-ekonomskih specifičnosti IS i IT uopšte. Te zakonitosti, koje u najvećoj meri važe i za ostale tehnologije, a koje su od ključnog značaja za uspešno uvođenje informacionih sistema, uključujući i one u oblasti zaštite životne sredine, biće ukratko opisane u ovom poglavlju, dok će osnove životnog ciklusa IS biti date u narednom poglavlju.

Jedan od najosnovnijih koncepata kada je bilo koja tehnologija u pitanju, a time i IT i IS, jeste tehnološka S kriva, koja opisuje razvoj performansi određene tehnologije, prikazan na ordinati, u vremenu, prikazanom na apscisi. Alternativno, vreme može biti zamenjeno cenom, ukoliko se želi analizirati šta je moguće postići jednom određenom tehnologijom, u zavisnosti od raspoloživih sredstava. U oba slučaja, na ovoj tehnološkoj krivoj mogu se uočiti 3 područja, koja zajedno čine oblik latiničnog slova S, po kome je kriva i dobila ime (slike 7.2.).

U prvom, početnom području, koje se često aproksimira kvadratnom funkcijom, performansa raste veoma brzo sa vremenom ili sa rastom ulaganja, što je najčešće posledica otklanjanja tehničkih teškoća, koje su postavljene pred novu tehnologiju, pa se ova faza često naziva razvojnom. Treba međutim uočiti da je upravo zbog tih teškoća rast na samom početku veoma spor. Ili, ukoliko se kao nezavisno promenljiva posmatra ulaganje, kada se nedovoljno uloži dobije se vrlo mala performansa, što vodi u gubitak.

Drugo područje predstavlja evolutivnu fazu tehnologije i predstavlja se pravom linijom, što znači da tehnologija daje stabilan rast performansi, što je čini predvidljivom i pogodnom za primenu.

Treće područje je područje zasićenja, kada tehnologija dostiže svoj fizički limit, i rast performansi je vrlo spor, često aproksimiran logaritamskom funkcijom, bilo da je u pitanju potrebno dodatno ulaganje vremena ili sredstava. U tom slučaju, potrebno je tehnologiju T1 zameniti novom tehnologijom T2, koja će pružiti bolje performanse, kao na primeru na slici 7.2.

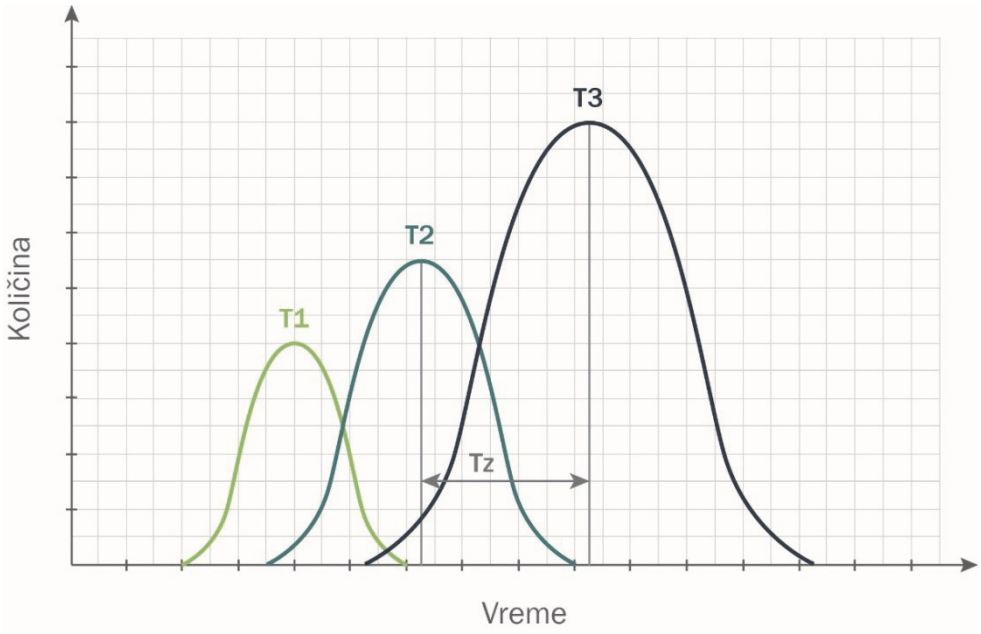


Slika 7.2. Tehnološka S kriva, u kojoj se kao nezavisno promenljiva može pojaviti i ulaganje, odnosno cena. Tehnologija T2 zamenjuje tehnologiju T1.

Spečifičnost koja razlikuje IT, a time i IS, od ostalih tehnologija je u tome što je vreme u kome nastaje zasićenje, odnosno vreme zamene tehnologije, često nazivano i životni vek tehnologije ili tehnološki ciklus, znatno kraće.

Tehnološki ciklus najčešće se procenjuje krivom korišćenja ili prodaje proizvoda na bazi određene tehnologije, koja ima zvonasti oblik i ciklično se ponavlja pri uvođenju nove tehnologije, slika 7.3. Može se uočiti vreme početka, rasta, maksimuma, opadanja i konačno gašenja proizvoda određene tehnologije, pa se kaže da takva kriva opisuje njegov životni vek.

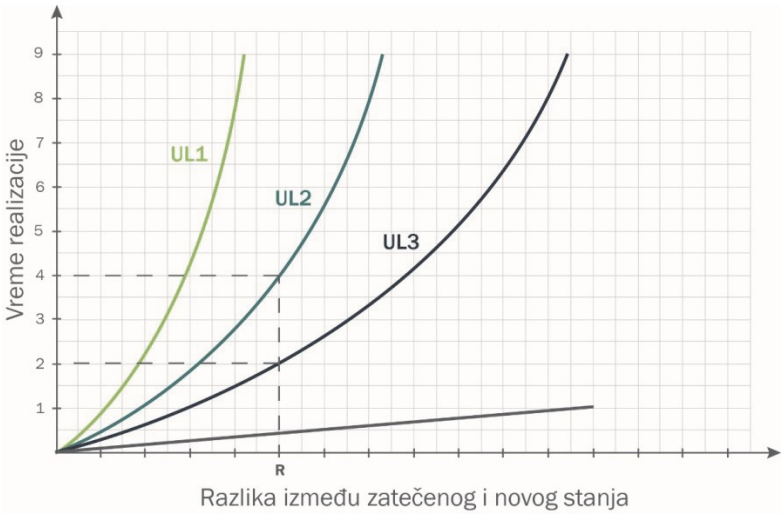
Od posebnog je interesa vreme koje protekne između dva susedna maksimuma prodaje proizvoda različitih tehnologija. Ono se naziva vremenom zastarevanja tehnologije, na slici označeno sa *Tz*. Smatra se da je danas u oblasti računarskog hardvera vreme zastarevanja tehnologije oko 12 meseci, pa čak i manje kada je u pitanju mobilno računarstvo (na primer pametni telefoni), dok je u oblasti softvera nešto duže i iznosi obično 18 do 24 meseca, što je i dalje znatno kraće od velikog dela proizvodnih tehnologija.



Slika 7.3. Vreme zastarevanja tehnologija

Vreme zastarevanja je izuzetno važno sa tehno-ekonomskog stanovišta, pošto je u okviru perioda jednakog trostrukom *Tz* neophodno pustiti u rad određenu tehnologiju i eksploatisati je tako da ona u potpunosti otplati u nju uložena sredstva, i konačno donese korist, odnosno dobit. U kontekstu IS, imajući u vidu vreme zastarevanja softvera kao njegove ključne komponente, može se zaključiti da ukupan period u kome je potrebno razviti, odnosno pustiti u rad IS, i zatim ga efektivno koristiti da bi se u potpunosti otplatio iznosi približno 5 godina.

Već se može uočiti da ovako kratak period predstavlja veliki izazov za uspešno i rentabilno korišćenje IS, ali, da bi se stekla potpunija slika, neophodno je uspostaviti precizniji odnos između zahteva koji se postavljaju pred IS i resursa raspoloživih za njegovu realizaciju, pre svega ulaganja i vremena. Ova međuzavisnost je prikazana na slici 7.4, gde se na apscisi nalazi razlika između postojećeg stanja IS i novog željenog stanja u normalizovanim jedinicama (ukoliko za dati realni sistem ne postoji IS postojeće stanje je 0), a na ordinati vreme potrebno za realizaciju. Krive predstavljaju različite vrednosti ulaganja, pri čemu je UL1 < UL2 < UL3.

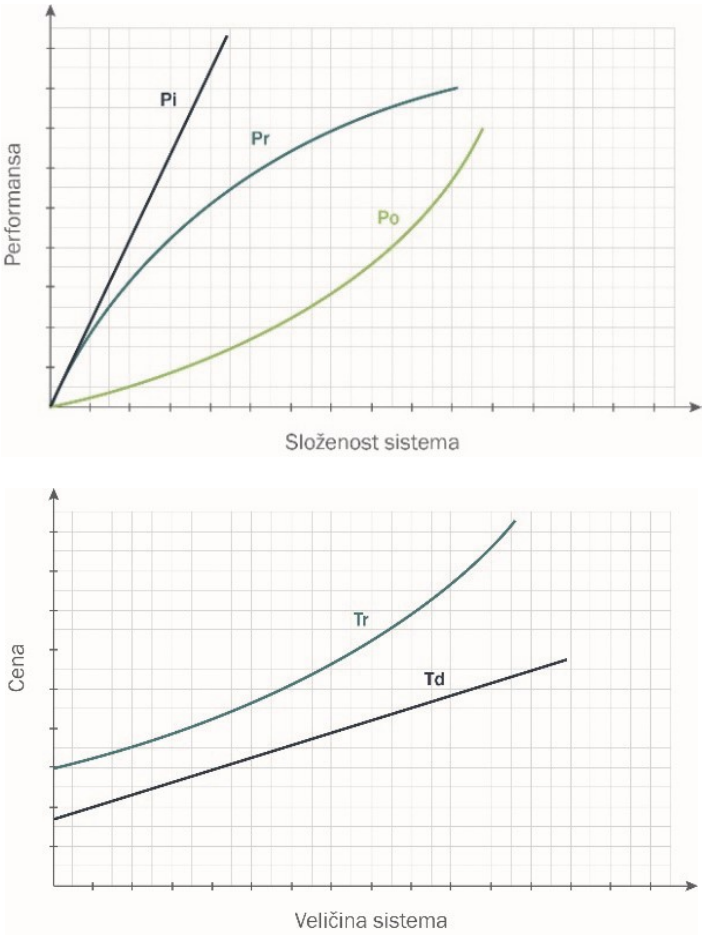


Slika 7.4. Vreme zastarevanja tehnologija.

Sa grafika se može uočiti da se, za određenu vrednost ulaganja, na primer UL3, željena razlika R između postojećeg i novog stanja IS, odnosno kompletno puštanje u rad novog IS, može ostvariti za 2 vremenske jedinice, odnosno 2 godine. Ukoliko bi umesto ovog nivoa ulaganja imali na raspolaganju UL2, vreme za kompletnu realizaciju bi bilo 4 godine, što znači da bi IS praktično već pri puštanju u rad bio zastareo, čime bi njegove šanse da se otplati postale veoma male. Očigedno je da je jedino moguće rešenje u uslovima manjih sredstava raspoloživih za ulaganje organizovanje razvoja u manjim koracima, na primer u ovom slučaju da se ulaganjem UL1 ostvari razlika od 1 jedinice, za šta bi bila potrebna 1 godina (slika 7.4), što bi dalo dovoljno vremena za eksploataciju i otplatu ulaganja u IS pre njegovog zastarevanja.

Na žalost, čak dovoljno visoka ulaganja nisu garancija za uspeh IS. Moguće su greške u različitim fazama razvojnog ciklusa IS (o čemu će biti više reči u narednom poglavlju), a sa povećanjem obima ili zahteva koji se postavljaju pred IS eksponencijano pada efikasnost i povećavaju se troškovi. Ovaj fenomen je već poznat iz opšte teorije sistema: sa nastavljanjem rasta složenosti sistema, sve veći udeo resursa sistema bavi se održavanjem samog sistema kao koherentne celine, a ne njegovom osnovnom svrhom, što dovodi do opadanja efikasnosti sistema, a u ekstremnim slučajevima, i do njegove diskfuncije. Istovremeno, troškovi razvoja (Tr) složenih sistema rastu brže od složenosti sistema, odnosno od direktnih troškova (Td), slika 7.5, što je potkrepljeno i skorijim istraživanjima, koja pokazuju da je stepen uspešnosti velikih projekata uvođenja IS znatno ispod prosečnog i kreće se od 10 do 20%. S druge strane, iako manji sistemi imaju mnogostruke

prednosti sa stanovišta brzine uvođenja, upravljivosti i druge, oni u sebi kriju opasnost ulaska u „zamku S-krive“, odnosno da tehnologija koja se upotrebljava nije dovoljna za rešavanje postavljenog problema IS, čime se može ući u logaritamski deo S-krive, izdvojiti previše vremena i sredstava bez adekvatnog rezultata i na kraju opet doživeti neuspeh.



Slika 7.5. Performanse i troškovi razvoja u zavisnosti od veličine sistema

Može se zaključiti da, bez obzira na zaista impresivan razvoj IT, uspeh u njihovom uvođenju i posebno u razvoju i eksploataciji IS nije unapred zagarantovan, već je, naprotiv, vrlo moguće da se ulaganja u IS ne opravdaju. Ovaj neželjeni scenario može se, međutim, izbeći razumevanjem tehno-ekonomskih specifičnosti IS, dobrom postavkom i vođenjem projekta razvoja i uvođenja IS, kao i dobrom konceptualizacijom i projektovanjem IS u odnosu na dati realni sistem.

7.2.2. Životni ciklus informacionih sistema i softvera

S obzirom na sve veću važnost IS i softvera uopšte, čije pravilno, odnosno nepravilno funkcionisanje, može imati velike ekonomske posledice, pa čak ugroziti i bezbednost i živote ljudi, kao i na činjenicu da u mnogim slučajevima njihove performanse mogu biti nezadovoljavajuće, o čemu je bilo reči u prethodnom poglavlju, životni ciklus IS i softvera su predmet intenzivnog proučavanja u prethodnih 50 godina. Kao rezultat ovih istraživanja, kao i sticanja praktičnih iskustava u razvoju i eksploataciji IS i softvera, razvijen je veliki broj pristupa, klasifikacija i metodologija, koje su se pokazale manje ili više pogodnim za različite vrste projekata i realnih sistema koji su modelovani, što je u krajnjoj liniji dovelo i do napretka u ovoj oblasti.

Međutim, jedna od neželjenih posledica ove raznolikosti je i nedostatak precizne terminologije, klasifikacije i definicija, kako na srpskom tako i na engleskom jeziku, što često dovodi do toga da se slične ili iste stvari potpuno različito nazivaju, organizuju i izvršavaju, čime dolazi do nesporazuma, problema u projektima, a ponekad i do ozbiljnih grešaka. Ovo se odnosi čak i na sam pojam životnog ciklusa IS, o čemu će nadalje biti reči, mada se najveći deo razmatranja može primeniti na softver generalno.

Iako su i faza razvoja i faza eksploatacije kompleksne i neophodne za uspešan krajnji rezultat upotrebe IS, faza razvoja se uobičajeno smatra složenijom, kritičnijom i podložnijom greškama, koje mogu dovesti do degradacije osobina IS, sve do njegove neupotrebljivosti. Zbog ovoga se često životni ciklus IS često poistovećuje sa razvojnim ciklusom IS ili metodologijom razvoja IS, i ovi termini se naizmenično upotrebljavaju i u literaturi na srpskom i u literaturi na engleskom jeziku. Sa druge strane, razvojem IT, faza eksploatacije i održavanje IS koje se u tom periodu dešava dobijaju sve više na značaju, s obzirom da u nekim slučajevima troškovi u toj fazi dostižu i preko 50% ukupnih troškova u okviru životnog ciklusa sistema.

Za razumevanje svih faza životnog ciklusa IS, od koristi je takozvana SDLC metodologija (od engl. *Systems Development Life Cycle*), koja predstavlja hronološki prvi metodološki okvir za razvoj i korišćenje IS. SDLC na pregledan i jasan način sistematizuje životni ciklus IS, mada treba napomenuti da postoje i drugačije podele i varijacije, a pojedini procesi u okviru faze životnog ciklusa softvera su detaljnije razrađeni u međunarodnom standardu ISO/IEC 12207.

Po SDLC metodologiji životni ciklus IS se sastoji od 10 faza, koje će ovde biti ukratko opisane:

1) Faza inicijalizacije, koja počinje uočavanjem potrebe za IS. U ovoj fazi već je potrebno definisati ko upravlja projektom razvoja, odnosno uvođenja IS (engl. *Project Manager*) i napraviti predlog koncepta IS.
2) Faza razvoja koncepta IS obuhvata detaljno razmatranje koncepta IS, definisanje obuhvata IS, kao i potrebnih resursa.
3) Faza planiranja podrazumeva razradu potrebnih aktivnosti i njihovih međuzavisnosti, terminiranje, definisanje potrebnih sredstava (na primer hardvera, sistemskog softvera i softverskih alata), razmatranje eventualnih sigurnosnih aspekata, kao i načina na koji će se projekat kontrolisati.
4) Faza analize zahteva, je faza u kojoj se formalno definišu funkcionalni i nefunkcionalni zahtevi koji se postavljaju pred IS. U funkcionalne spada ono što je protrebno da bi IS na odgovarajući način modelovao realni sistem, odnosno jednostavno rečeno kako treba da radi, kakav mu je obuhvat i generalna struktura podataka. Nefunkcionalni zahtevi obuhvataju preduslove da bi IS ispunjavao svoju namenu na zadovoljavajući način i obuhvataju potrebne performanse i brzinu odziva, bezbednost, mogućnost održavanja i proširenja, portabilnost i slično. Svi ovi zahtevi moraju biti definisani tako da se mogu meriti odnosno testirati.
5) Faza projektovanja, je faza tokom koje se definišu detaljne strukture podataka i podsistema, njihovih ulaza i izlaza, kao i njihovih sastavnih delova – modula i jedinica, pri čemu se za svaki modul definiše detaljna logička specifikacija. Na primer, model Objekti-Veze, pomenut ranije u ovom poglavlju, primenjuje se u ovoj fazi. Takođe, svi elementi specifikacija koji eventualno zahtevaju uvid i odobrenje krajnjih korisnika IS u ovoj fazi moraju biti detaljno dokumentovani i odobreni.
6) Faza kodiranja, u toku koje se prethodno definisane specifikacije prevode u izvršni programski kod i eventualno hardver i komunikacije. U ovoj fazi se, dakle, obavlja programiranje.
7) Faza testiranja i integracije, je faza u kojoj je potrebno sistematski testirati komponente IS. Testiranja se često obavljaju i od strane krajnjih korisnika, da bi se osiguralo da su funkcionalni zahtevi ispunjeni na pravi način. Takođe, u ovoj fazi se obavlja integracija modula i podsistema, što se takođe mora testirati, kako u pogledu funkcionalnih, tako i nefunkcionalnih zahteva.
8) Faza puštanja u rad dolazi posle obavljenih testiranja i prijema sistema od strane korisnika, i obuhvata instalaciju sistema u okruženju u kome će se koristiti, sa potrebnim podešavanjima i eventualnim manjim doterivanjima, sa ciljem da IS radi u skladu sa definisanim specifikacijama. U slučaju

većih modifikacija IS, ova faza se ponavlja u toku perioda njegovog korišćenja.

9) Faza korišćenja i održavanja je po pravilu najduža faza životnog veka IS. U okviru ove faze radi se praćenje sistema, kao i njegovo održavanje, što u ovoj oblasti predstavlja uvođenje potrebnih modifikacija u sistemu da bi IS pratio eventualne promene u realnom sistemu, bolje modelovao realni sistem, bolje ispunjavao nefunkcionalne zahteve, obezbedio bolju jednostavnost rukovanja za korisnike i prilagodio se eventualnim promenama u svom IT okruženju (na primer usled promene operativnog sistema ili hardvera). Kao što je ranije napomenuto, osim zbog svoje dužine, ova faza je značajna i zbog toga što su troškovi ostvareni u ovoj fazi često veći od troškova svih ostalih faza zajedno.
10) Faza dispozicije ili odlaganja, je faza u kojoj se obezbeđuje organizovani prestanak rada sistema i čuvanje svih važnih informacija o IS, da bi se on u budućnosti mogao reaktivirati, ukoliko je potrebno. U ovoj fazi često je potrebno obratiti posebnu pažnju na odgovarajuće čuvanje i arhiviranje podataka iz IS, da bi se oni po potrebi mogli prebaciti (migrirati) na drugi IS.

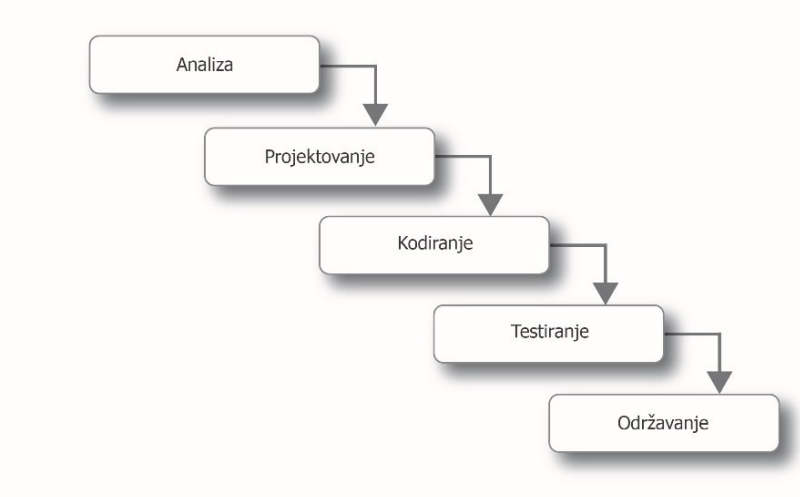
Zanimljivo je napomenuti da se u našim uslovima vrlo često daleko najveća pažnja posvećuje fazi kodiranja, koja se često naziva i razvoj (od engl. *development*), čime se informatika i razvoj IS poistovećuje sa programiranjem. Ovo može imati veoma negativne posledice, pošto strateške greške u koncipiranju i projektovanju dovode do toga da IS ne modeluje realni sistem na adekvatan način, što zakonomerno vodi do neuspeha u uvođenju IS, bez obzira koliko dobro bilo urađeno programiranje.

7.2.3. Modeli razvoja informacionih sistema i softvera

U toku prethodnih decenija rada na razvoju IS i softvera, postalo je jasno da se faze životnog ciklusa i prelazak između njih mogu organizovati na različite načine, što je dovelo do stvaranja većeg broja modela i metodologija razvoja. S obzirom na velika ulaganja industrije i napore da se razvoj softvera učini kvalitetnijim i produktivnijim, ova oblast se u novije vreme veoma razvila i još uvek se vrlo brzo menja, što dovodi do stalnog stvaranja novih modela. Sveobuhvatna analizu modela razvoja softvera izašla bi izvan granica ove knjige i može se naći u literaturi, ali će ovde biti kratko opisani najčešće korišćeni modeli, čije je poznavanje od praktičnog značaja za razumevanje i postizanje boljih rezultata u razvoju IS u oblasti inženjerstva i zaštite životne sredine: model vodopada, prototipski razvoj, inkrementalni razvoj, spiralni razvoj i RAD model. Svaki od ovih modela ima svoje dobre strane i svoje slabosti, pa izbor optimalnog modela razvoja zavisi od karakteristika realnog sistema odnosno IS koji treba razviti,

materijalnih i ljudskih resursa koji su na raspolaganju, kao i vremenskih i drugih organičenja.

Model vodopada predstavlja hronološki najstariji i još uvek jedan od najzastupljenijih modela razvoja IS. U okviru ovog modela, faze razvoja slede jedna drugu, što grafički prikazano podseća na vodopad, slika 7.6.



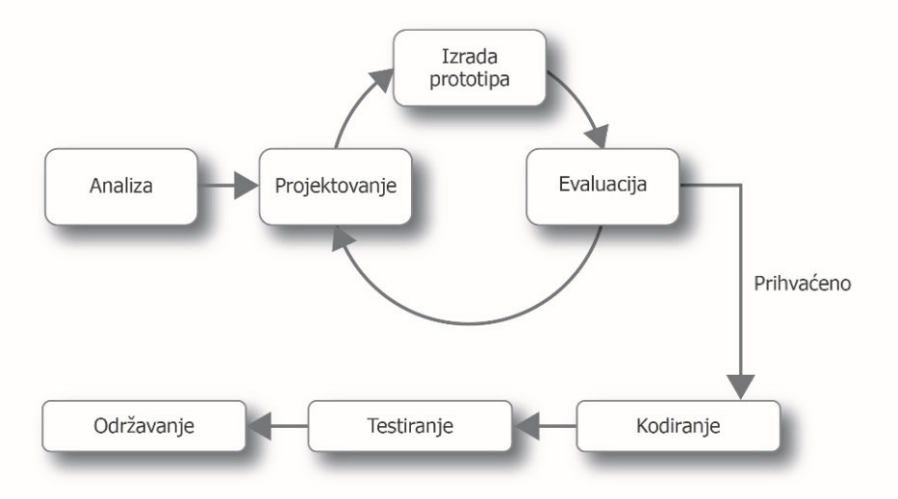
Slika 7.6. Model vodopada.

S obzirom na logički sled faza životnog ciklusa odnosno razvoja IS, ovaj model je je veoma pogodan za uspotavljanje čvrste kontrole između faza, što omogućava bolju kontrolu i pouzdanost proizvedenog softvera, lako merenje napretka projekta i određenu optimizaciju angažovanih ljudskih resursa.

Međutim, model vodopada ima i ozbiljnih nedostataka. Zbog potrebe da se čeka potpuni završetak jedne faze da bi se prešlo u drugu model vodopada je spor, nefleksibilan i srazmerno skup, što je uzrokovano između ostalog dosta formalizovanim i komplikovanim prelaskom iz jedne u drugu fazu. Najveća slabost ovog pristupa je, međutim, u tome što se sistem u celini testira na samom kraju procesa, što znači da se eventualne strateške greške u projektovanju ili kodiranju mogu uočiti tek na kraju razvoja, što može dovesti do velikih troškova i kašnjenja. stvara dosta dokumentacije

Kao odgovor na ove slabosti, pojavio se prototipski pristup kao model razvoja IS. Ideja ovog pristupa je da se rizik grešaka u razvoju smanji dekomponovanjem projekta odnosno IS na manje delove, koji se svaki pojedinačno mogu brže razviti, testirati i proveriti, čime se troškovi i vreme za uklanjanje većih grešaka značajno

smanjuju (slika 7.7). U okviru prototipskog razvoja, često se razvijaju mali pokazni primeri za demonstraciju određenih funkcionalnosti IS (engl. *mock-up*) i prototipovi koji se kasnije odbacuju, mada se u nekim slučajevima iz prototipova nadogradnjom može formirati finalni IS.



Slika 7.7. Prototipski razvoj.

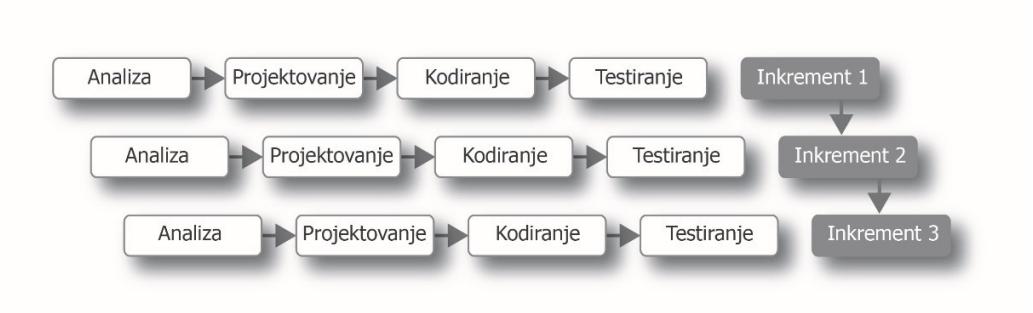
Prototipski pristup se pokazuje posebno pogodnim kada ciljevi nisu potpuno jasno definisani i kada realni sistem nije dovoljno poznat, što je čest slučaj u praksi, naročito kada specijalisti iz oblasti realnog sistema (npr. emisije zagađenja u vazduh ili druge medije, monitoring zagađenja ili EMS u preduzeću) nisu u potpunosti u stanju da formalno definišu funkcionisanje realnog sistema ili svoje potrebe za informacijama.

Prototipski razvoj ima i slabosti, u koje spada najčešće slabo dokumentovanje sistema, relativno slabija kontrola, što može dovesti do problema sa kvalitetom softvera i projektovanja IS, kao i stvaranje preteranih i nerealnih očekivanja o mogućnostima realizacije IS na osnovu funkcionalnosti koje su izgledaju urađene, a koje su često samo pokaznog karaktera, odnosno mogu biti tek manji deo celokupnog IS.

Neki autori ne smatraju prototipski razvoj jedinstvenom metodologijom, već više opštim pristupom, iz koga je proizišlo više metodologija, koje imaju za cilj da prevaziđu slabosti generalne slabosti prototipskog razvoja. U te metodologije spadaju inkrementalni razvoj, spiralni razvoj i RAD.

Inkrementalni razvoj se zasniva na relativno jednostavnoj ideji o dekomponovanju projekta na manje segmente, u okviru kojih se sprovode „mini vodopadi“, slika 7.8. U okviru inicijalne konceptualizacije i projektovanja sistema, može se primeniti i neka vrsta prototipskog razvoja.

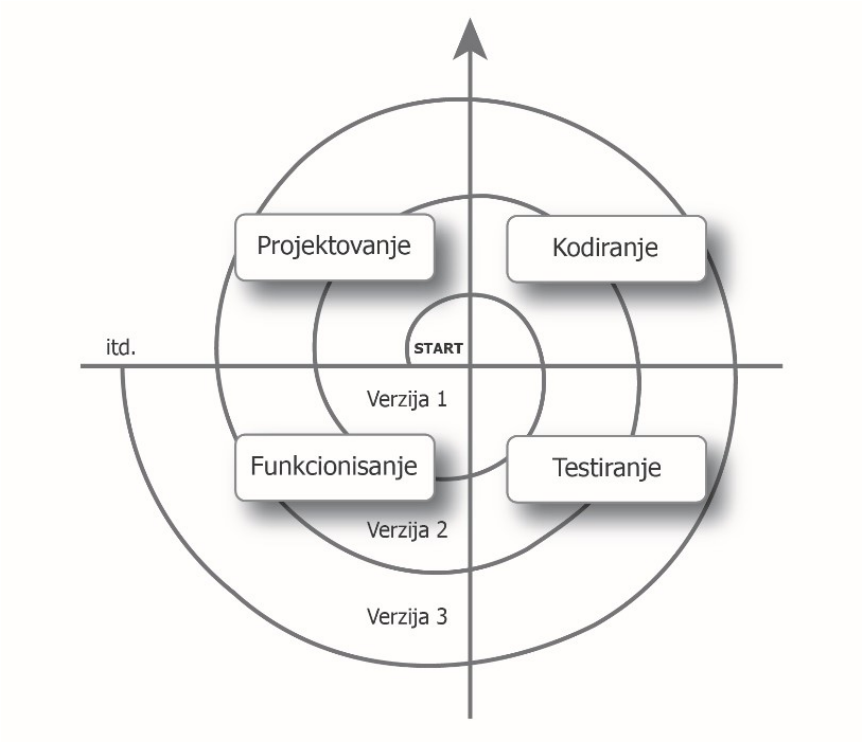
Koncept inkrementalnog razvoja spaja dobre osobine modela vodopada i prototipskog razvoja, ali je kod njega inicijana faza konceptualizacije, slično modelu vodopada, kritična za uspeh. Uz to, pri razvoju segmenata metodom „mini vodopada“ često ne postoji dovoljno poznavanje ostatka realnog sistema, što može dovesti do grešaka, a otvara i pitanje kasnijeg povezivanja (integracije) ovih segmenata.



Slika 7.8. Inkrementalni razvoj.

Spiralni razvoj se takođe zasniva na dekomponovanju projekta na manje celine, ali na cikličan način, pri čemu se u svakom ciklusu ili iteraciji napreduje kroz niz istovetnih koraka, koji se mogu predstaviti kroz 4 kvadranta: (1) određivanje ciljeva, alternativa i ograničenja za ciklus, (2) procena i izbor odgovarajućih alternativa, (3) kodiranje i testiranje delova IS predviđenih za ciklus, i (4) planiranje sledećeg ciklusa.

U spiralnom razvoju akcenat je na proceni rizika u svakoj fazi, pa je ovaj model vrlo uspešan u smanjenju rizika. Takođe, zbog istih koraka u okviru svakog ciklusa, olakšano je formiranje dolumentacije, a veoma je velika i fleksibilnost, pošto se u svakom od koraka mogu upotrebiti različite tehnike (npr. model vodopada ili prototipski razvoj). Ova fleksibilnost je istovremeno i slabost spiralnog razvoja, pošto nije jednostavno odrediti najbolju kombinaciju metodologija za svaki pojednični projekat, pa čak ni svaku iteraciju. Uz to, ne postoje tačni kriterijumi za prelazak iz jednog ciklusa u drugi, kao što ne postoje ni jasni rokovi, što zajedno dovodi do toga da je projektima zasnovanim na spiralnom razvoju vrlo složeno upravljati.



Slika 7.9. Spiralni razvoj.

RAD model (od engl. *Rapid Application Development*) je specifična metoda prototipskog razvoja, koja za glavni cilj ima što brži razvoj relativno kvalitetnog IS, uz istovremeno umanjenje troškova razvoja. Da bi se ovo postiglo, u velikoj meri se koriste softveri za automatizaciju programiranja, kao što su CASE (engl. *Computer Aided Software Engineering*) softveri i generatori aplikacija. Razvoj je iterativan, što znači da se formirani prototipovi ne odbacuju i podređen je ispunjenju rokova, čak i po cenu neispunjenja određenih, nekritičnih funkcionalnosti.

Brzina kojom se dolazi do gotovog i upotrebljivog IS predstavlja najbolju stranu RAD modela, pogotovo što se, pod uslovom da je projekat dobro vođen, istovremeno smanjuje angažman ljudskih resursa potrebnih za razvoj, kao i troškovi. Uz to, primena RAD obezbeđuje dobru koncentraciju na ključne delove IS, što za uzvrat često znači da će najvažniji delovi realnog sistema biti dobro modelovani.

Naravno, brzina ima i svoju drugu stranu, pa se može desiti da pojedini delovi realnog sistema ne budu dobro proanalizirani, zbog čega može doći do grešaka u koncipiranju i projektovanju IS. Kao i ostale faze, i kodiranje i testiranje sistema su po pravilu kraće, što može dovesti do generalno slabijeg kvaliteta IS. Uz to, kod IS razvijenih RAD modelom često se javljaju problemi sa dokumentacijom, mogućnošću izmene odnosno proširenja sistema i međusobnim povezivanjem modula.

7.3. Informacioni sistemi u zaštiti životne sredine

Informacioni sistemi u zaštiti životne sredine (engl. *Environmental Information Systems*, EIS) obuhvataju široki spektar softverskih tehnologija, uključujući klasične informacione sisteme, modele za simulaciju, sisteme za donošenje odluka i prilagođene aplikacije (rađene po potrebi specifičnog korisnika, engl. customized). EIS se koriste za različite namene, koje uključuju upravljanje podacima, informacijama i znanjem, komunikaciju, podršku odlučivanju na različitim nivoima i istraživanje, i mogu se podeliti u dve velike grupe:

1) Informacioni sistemi u užem smislu –za prikupljanje i čuvanje informacija, i
2) Informacione sisteme za analizu prikupljenih podataka i simulaciju procesa.

Najvažnije oblasti primene EIS prikazane su u Tabeli 7.2, a u daljem tekstu biće ukratko prikazani menadžerski informacioni sistemi i sistemi za podršku odlučivanju u zaštiti životne sredine, kao dve grupe EIS koje su posebno interesantne sa stanovišta primene u praksi.

Tabela 7.2. Opis najvažnijih oblasti primene EIS

Oblast primene	Opis
Monitoring	ICT sredstva se primenjuju za prikupljanje podataka o stanju u životnoj sredini, pri čemu se oni povezuju sa geografskim podacima o lokacijama i vremenskim intervalima
Modelovanje i simulacije	Razvoj modela za procena uticaja različitih procesa na životnu sredinu, eventualno uz socio-ekonomsku analizu. Modelovanje u oblasti životne sredine obično obuhvata analizu podataka prikupljenih monitoringom, kako bi se simulirao uticaj zagađenja na životnu sedinu, na primer na klimatske promene i biodiverzitet, i simulirali različiti scenariji zagađenja i njegovog rasprostiranja.
Socio-ekonomska analiza	Procena troškova i benefita različitih ekoloških strategija, koje se defifnišu kako bi se negativni efekti zagađenja, smanjenja biodiverziteta i osiromašenja prirodnih sirovina sveli na najmanju meru.
Strategija i planiranje	Obuhvata planiranje na različitim nivoima: internacionalnom, nacionalnom i regionalnom, pri čemu se koriste podaci i rezultati iz gore navedenih oblasti primene radi informisanja menadžera i donosioca odluka, definisanja strategija i planova, procene rizika i prognoze stanja životne sredine.
Razvoj kapaciteta i saradnje	Namenjeni su širokom krugu korisnika, sa ciljem edukacije i podizanja svesti u javnosti o neophodnosti zaštite životne sredine, kao i radi participacije u procesu donošenja odluka i efektivnijeg uključenja što šireg kruga zainteresovanih strana.

7.3.1. Menadžerski informacioni sistemi u zaštiti životne sredine

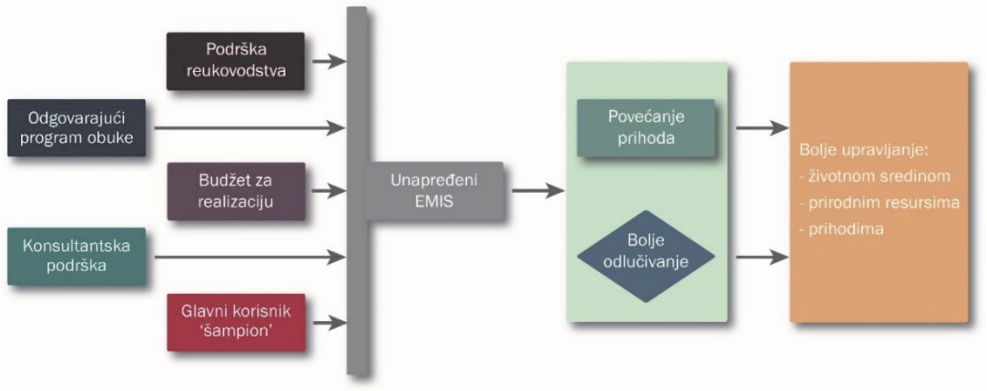
Začetak pojave menadžerskih informacionih sistema u oblasti zaštite životne sredine (engl. *Environmental Management Information Systems*, EMIS) se vezuje za naučne publikacije sa početka 90-ih godina prošlog veka, a poseban zamah im je dao razvoj menadžmenta zaštite životne prema standardima serije ISO 14000, o čemu se više detalja može naći u poglavlju 2 ove knjige.

EMIS se uglavnom koriste od strane kompanija, lokalnih vlasti, dobavljača, banaka, osiguravajućih društava, investitora i drugih organizacija, vrlo često, posebno u kompanijama, kao deo sistema za rano upozoravanje, kroz identifikaciju ekoloških rizika procesa proizvodnje i dobijenih proizvoda.

Iako se na prvi pogled mogu učiniti relativno specijalizovanima, EMIS obuhvataju prilično širok dijapazon primena i pristupa, pa se mogu podeliti na više različitih grupa:

- informacioni sistemi za eksterno izveštavanje,
- sistemi za unutrašnje procese i operacije,
- sistemi za procenu životnog ciklusa,
- sistemi zasnovani na ključnim indikatorima performasi (KPI),
- sistemi za ekološko računovodstvo,
- sistemi za izveštavanje o održivosti (engl. *sustainability reporting systems*),
- procesno i proizvodno orijentisani sistemi.

Kao primer, faze u razvoju EMIS sistema za potrebe upravljanja na opštinskom nivou prikazane su na Slici 7.10.



Slika 7.10. Faze razvoja EMIS.

Niže su navedeni neophodni koraci u uspostavljanju navedenog EMIS na primeru jedne opštine, pri čemu se analogni pritup može primeniti i za preduzeća:

- Projektovanje EMIS baza podataka
  - Obuhvata prikupljanje i obradu podataka uz konstantno unapređivanje potrebnih baza podataka. Treba imati u vidu da se EMIS po pravilu sastoji od gotovih softverskih rešenja, što ubrzava

njegov razvoj i pripremu, ali često postavlja pitanje veze i integracije između različitih baza podataka.
    - Opštinska uprava, službe, osoblje i članovi radnih grupa moraju u potpunosti biti upoznati sa ciljevima, procesima i aktivnostima koji se vrše u okviru EMIS.
- Implementacija EMIS
    - Uprava mora biti upoznata sa EMIS i njegovim zasebnim komponentama, na primer Geografskim Informacionim Sistemom (GIS), da bi mogla da sagleda njegov značaj ali i troškove uvođenja i održavanja, kako bi bila u stanju da podrži njegovo uspostavljanje.
    - Neophodno je sprovesti i odgovarajuće programe obuke, kako bi se zaspoleni obučili za rad sa EMIS i njegovim komponentama.
    - Kako je uvođenje EMIS skupa i dugotrajna investicija, nije neophodno obezbediti potrebna sredstva na samom početku njegovog uvođenja, već je potrebno obezbediti dovoljna sredstva na godišnjem nivou kako bi se EMIS fazno implementirao.
    - Sa stanovišta finansiranja, EMIS ne bi trebalo sagledavati kao trošak za opštinu, već kao potencijalni izvor novih prihoda i benefit za zajednicu.
    - Od osoblja zaduženog za implementaciju EMIS-a mora postojati bar jedan ekspert koji dobro poznaje ICT i EMIS, kako bi mogao da bude odgovoran za njegov razvoj i implementaciju, ali i posredovanje između opštinske uprave i koristika EMIS.
    - Potrebno je obezbediti i konsultansku i tehničku podršku u svim fazama izrade i implementacije EMIS. Tehnička podrška je neophodna kako bi se što efikasnije EMIS integrisao i prilagodio potrebama i mogućnostima opštinskih službi.

Analiza efikasnosti EMIS pokazala je da u većini slučajeva dolazi do poboljšanja performansi kompanija i organizacija u ispunjavanju ekoloških standarda. Poređenje poslovanja pre i posle uvođenja EMIS pokazalo je da su kompanije nakon uvođenja EMIS sposobnije da unapređuju upravljanje zaštitom životne sredine i smanjen je rizik od ekoloških akcidenata. Takođe se pokazalo da EMIS naročito doprinosi efikasnom upravljanju resursima, kao što su potrošnja vode ili generisanje otpada.

7.3.2. Sistemi za podršku odlučivanju u zaštiti životne sredine

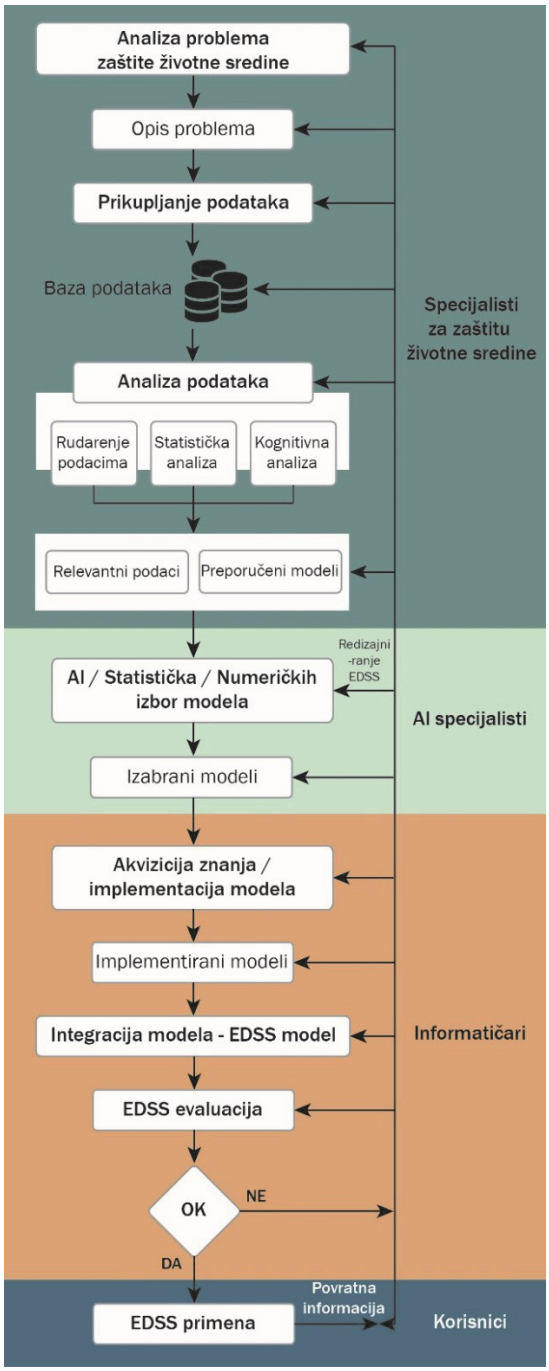
Koncept sistema za podršku odlučivanju (engl. *Decision Support Systems*, DSS) razvijen je na osnovu naučnih radova Herberta Simona, koji se bavio odlučivanjem u organizacijama, i koji je definisao tri glavne faze ovog procesa:

- Identifikacija problema, njihovog obima i potrebe za promenom, odnosno aktivnim delovanjem,
- Razvoj alternativnih strategija, planova ili opcija za rešavanje problema identifikovanih tokom prethodne faze,
- Proces ocenjivanja alternativa i izbora, odnosno donošenja odluke.

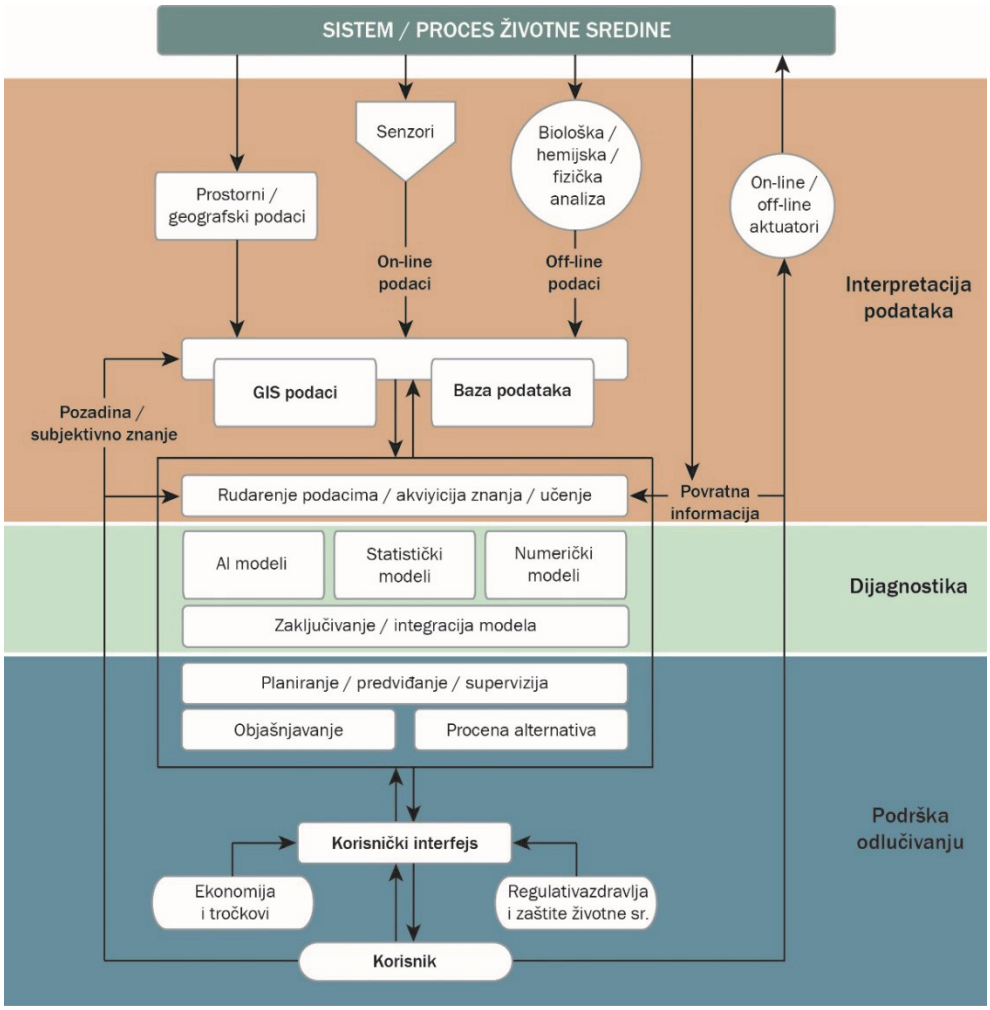
Rad na razvoju tehnologije koja bi se koristila kod upravljanja i donošenja strategija za rešavanje složenih problema u oblastima zaštite životne sredine i održivog razvoja dovelo je do nastanka DSS za primenu u zaštiti životne sredine (engl. *Environmental Decision Support Systems*, EDSS).

EDSS se definiše kao DSS koji je razvijen sa ciljem da se u okviru zaštite životne sredine i održivog razvoja integrišu modeli, baze podataka i druga sredstva za pomoć u odlučivanju, integrisani na način da donosioci odluka i menadžeri mogu da ih efektivno koriste. EDSS predstavljaju inteligentne informacione sisteme koji smanjuju vreme potrebno za donošenje odluka, istovremeno povećavajući njihov kvalitet i doslednost.

Već na osnovu definicije vidljivo je da su EDSS veoma kompleksni informacioni sistemi, čiji je razvoj i uvođenje veoma složeno. Šematski prikaz razvoja EDSS-a dat je na slici 7.11., dok je primer jednog EDSS-a prikazan na slici 7.12. Primeri novijih EDSS-a sa kratkim opisom dati su u Tabeli 7.3.



Slika 7.11. Razvoj EDSS-a



Slika 7.12. Primer EDSS-a

Tabela 7.3. Primeri EDSS-a sa kratkim opisom

Naziv	Opis
GAINS	Integrisani model za procenu (engl. *Integrated Assessment Model*, IAM) emisija u vazduh, sa modulima za simulaciju i optimizaciju, sa ciljem podrške strategiji poboljšanja kvaliteta vazduha sa najmanjim troškovima.
WARGI-QUAL	EDSS za modelovanje kompleksnih vodnih sistema sa većim brojem akumulacija i višestrukom upotrebom i recirkulacijom, sa dodatnom analizom sastava algi u akumulacijama.
CAPER	EDSS razvijen da istraži odziv akvatičkih sistema prema promenama u sadržaju nutrijenata i suspendivanih čestica.
IBIS	EDSS kaji se koristi za istraživanje i analizu mogućih uticaja klimatskih promena i scenarija sanbdevanja vodom.
SMOM	Višemodulni EDSS za višekriterijumsku analizu (engl. *Multi-Criteria Decision Analysis*, MCDA) i simulaciju vezanu za projektovanje novih poljoprivrednih postrojenja.

Da bi EDSS mogao uspešno praktično da se primenjuje on mora da zadovolji sledeće kriterijume:

- Sposobnost da podrži procese menadžmenta, analize scenarija i pravljenja strategija, što znači da EDSS mora da bude u stanju da pruži razumljive rezultate, koji predstavljaju adekvatne odgovore na pitanja korisnika, kao i da bude od pomoći pri analizi tako dobijenih rezultata, na primer korišćenjem dijagrama, mapa i slično. Dodatni kriterijum uspešnosti EDSS je i broj korisnika, organizacija i pojedinaca, koji ga primenjuju.
- Adekvatna naučna, odnosno inženjerske zasnovanost, koja se proverava kada god je moguće validacijom na osnovu stvarnih podataka, na primer na osnovu stvarno izmerenih emisija i imisija. U ovu grupu kriterijuma spadaju i metode kojima EDSS obrađuje i prikazuje nesigurne podatke i rezultate, kojih često ima u praksi.
- Posedovanje odgovarajućih performansi i kvaliteta samog softvera, što se pre svega odnosi na jednostavnost korišćenja, mogućnost proširenja i jednostavnost održavanja, ispravljanja bagova i unapređenja.

Reference

Athanasiadis, P. A., Mitkas, A. E., Rizzoli, J. M. Gómez, (Eds.), Information Technologies in Environmental Engineering, Berlin, Heidelberg: Springer, **2009**.

Elliott, G., *Global Business Information Technology: an integrated systems approach*, Pearson Education, **2004**.

Geneca, *Why up to 75% of Software Projects will Fail*, **2013**. https://www.geneca.com/blog/software-project-failure-business-development

Hermann, M., Pentek, T., Otto, B., *Design Principles for Industrie 4.0 Scenarios*, 49th Hawaii International Conference on System Sciences (HICSS), Koloa, HI, USA, IEEE, **2016**, pp. 3928-3937.

Hastie, S., Wojewoda, S., *Standish Group 2015 Chaos Report*, **2015**. https://www.infoq.com/articles/standish-chaos-2015

International Organization for Standardization, *ISO/IEC 12207: Systems and software engineering -- Software life cycle processes*, Geneva, **2017**

Jones, D., *66% of IT Projects Fail*, **2016**. https://projectjournal.co.uk/2016/02/20/66-of-it-projects-fail/

Krigsman, M., *Study: 68 percent of IT projects fail*, **2009**. http://www.zdnet.com/article/study-68-percent-of-it-projects-fail/

Lazarević, B., Marjanović, Z., Aničić, N., et al., *Baze podataka*, Beograd: Fakultet organizacionih nauka, **2006**.

Pocajt, V., Antanasijević, D. Ristić, M., et al., *Environmental Sustainability and Information Technologies: A Dynamic Interdependence.* International Science Conference Reporting for Sustainability, Bečići, Montenegro, **2013**, pp.39-47.

US Department of Health and Human Services, *Selecting a Development Approach*, **2008.** https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS - Information - Technology / XLC / Downloads / SelectingDevelopment Approach.pdf

US Department of Justice, *Systems Development Life Cycle Guidance Document*, **2003**. https://www.justice.gov/archive/jmd/irm/lifecycle/table.htm