SOFTVERSKI (NE)INŽENJERING I INFORMACIONA SIGURNOST U CILJU POSTIZANJA KVALITETE

SOFTWARE (NON)ENGINEERING AND INFORMATION SECURITY FOR ACHIEVING QUALITY

Muharem Redžibašić
Univerzitet u Zenici
Politehnički fakultet
Zenica

REZIME

U modernom dobu razvoja informacionih tehnologija sve popularniji je trend potrebe veće radne snage na tržištu, a samim time i pojava novih radnih mjesta. Dok se jedni obrazuju da bi radili u softverskoj industriji, drugi pak tvrde da za to nije potrebno nikakvo formalno obrazovanje. Kao i u ostalim strukama poput građevine i u polju softverskog inženjeringa su već odavno klasifikovana radna mjesta i imamo inženjere koji poznaju cjelokupan proces razvoja softvera i istom pristupaju sistemski, a imamo i radna mjesta koja su dio toga procesa ali za ista je potreban samo određeni skup tehničkih vještina koji se često može steći i bez formalnog obrazovanja. Dakle, ovdje jasno pravimo distinkciju između sofverskog inženjeringa i samog kodiranja (pisanja programskog koda) koji mnogi podrazumijevaju kada kažu da su programeri. Pojava i raširena upotreba sistema vještačke inteligencije upravo je potvrda ovim navodima. U samom procesu nastanka softvera jako je bitno obratiti pažnju na informacionu sigurnost, te stoga će ovaj rad obuhvatiti smjernice za kvalitetan kompletan proces razvoja softvera postavljajući jasne granice između kodiranja i ostalih bitnih stvari u procesu softverskog inženjeringa, a ponajviše iz ugla sajber sigurnosti.

Ključne riječi: softverski inženjering, kodiranje, razvoj aplikacija, sajber sigurnost, NIS2

ABSTRACT

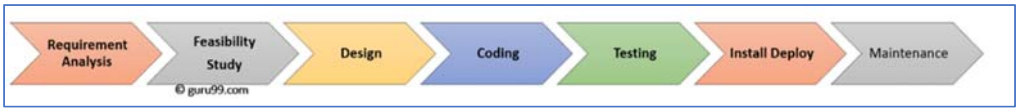
In the modern era of information technology development, the trend and the need for a larger workforce on the market are becoming increasingly popular, and with it the emergence of new jobs. While some are educated to work in the software industry, others claim that no formal education is required for this. As in other professions such as construction, in the field of software engineering, they have long since classified jobs and we have engineers who know the entire software development process and approach it systematically, and we also have jobs that are part of that process but only require a certain set of technical skills that can often be acquired without formal education. So, here we clearly make a distinction between software engineering and the actual coding (writing program code) that many people mean when they say they are programmers. The emergence and widespread use of artificial intelligence systems is precisely confirmation of these statements. In the process of software creation itself, it is very important to pay attention to information security, and therefore this paper will include guidelines for a quality complete software development process by setting clear boundaries between coding and other important things in the software engineering process, especially from the perspective of cybersecurity.

Keywords: software engineering, coding, application development, cybersecurity, NIS2

1. SOFTVERSKI INŽENJERING I KODIRANJE

Pojam softverski inženjering možemo posmatrati kao jedan cjelokupan širok proces koji ima za cilj stvaranje efikasnog i svrsishodnog softvera. Kao i svaka inženjerka aktivnost sastoji se od niza faza koje treba ispoštovati kako bi se proces kreiranja softvera na kraju i evaluirao. Po većini autora postoji u prosjeku 7 faza koje ćemo izdvojiti, a to su [2]:

- Prikupljanje i analiza zahtjeva,
- Studija izvodljivosti,
- Dizajn,
- Kodiranje,
- Testiranje,
- Instalacija/uvođenje,
- Održavanje.



Slika 1. Faze softverskog ciklusa [2]

Svaka faza je jako bitna ukoliko želimo postići zadati cilj, međutim mi ovdje želimo naglasiti da nivo znanja i vještina koje zahtijevaju određene faze se uveliko razlikuje. U agilnim metodama razvoja softvera svakako je bitan modularni pristup što se ponajviše koristi u fazi dizajna. Tu nam treba mnogo iskustva, ulaznih parametara i širok spektar inženjerskih znanja kako bi pružili najbolji metod i ponudili najbolje rješenje određenog problema, a u ovom kontekstu to je najčešće neki poslovni proces koji treba digitalizovati. U zavisnosti od kvalitetno urađenih prethodnih faza, a pogotovo faze dizajna zavisi i kvalitet cjelokupnog sotvera jer nakon ove faze slijedi faza kodiranja gdje nije potreban širok spektar znanja već poznavanje određenog programskog jezika u kojem će se kodirati softver. U pogledu procesa i zadataka softverskim inženjeringom se fokusiramo na procese, a sam dio pisanja koda, kodiranje, odnosno programiranje svodi se na rješavanje određenih zadataka (eng. tasks). [6]

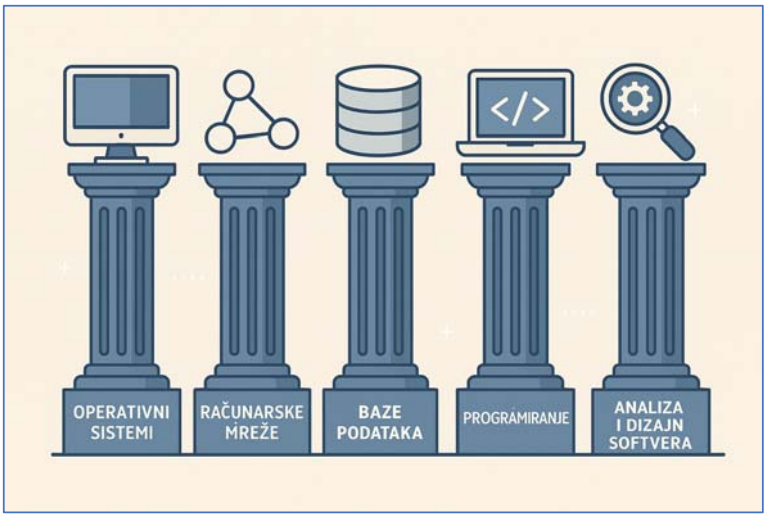
U većini slučajeva da bi bili koder, odnosno programer nije potrebno formalno obrazovanje niti određena akademska titula, dovoljno je da su učesnici toga procesa tehnički osposobljeni i da poznaju potreban programski jezik koji se koristi u toj fazi, a praksa je pokazala da u većini slučajeva to rade studenti ili osobe entuzijasti koji imaju uspješno evaluirane neke od certifikata iz određene oblasti.

Dakle, distinkcija je sada više nego jasna. Sa jedne strane trebamo sistemski da pristupimo problemu i pronađemo najbolji način za njegovo rješavanje, a da pri tome koristimo znanja i vještine iz mnogih oblasti. To uglavnom zahtjeva i akademsku titulu iz tehničkih nauka i relevantno iskustvo, s druge strane imamo potrebu da znamo samo određenu sintaksu potrebnog programskoj jezika i da izvršimo zadatke iz faze dizajna u najboljem mogućem redu i roku. Nedvosmisleno, kodiranje/programiranje možemo da nazovemo zanatom modernog doba gdje nije potreban širok spektar znanja i vještina već sklonost ka apstraktnom načinu razmišljanja kako bi kroz određeni programski jezik preveli naše zadatke u jezik razumljiv računaru.

Ukoliko se osvrnemo unazad par godina za pisanje programskog koda u bilo kojem programskom jeziku za potrebe neke jednostavne „log in“ forme za ograničavanje pristupa aplikaciji i definisanja korisničkih prava pristupa trebalo je više dana kodiranja. Sada uz sisteme

vještačke inteligencije kroz par adekvatnih upita možemo da dobijemo „gotov“ programski kod kroz par minuta. To je samo jedan od dokaza u nizu da se tu radi o repetitivnim stvarima i da tu nije potreban nikakav sistemski pristup kako kod inženjeringa. Tu se svakako postavlja pitanje koliko se smiju koristiti sistemi vještačke inteligencije zbog same tajnosti određenih projekata i koliko mi ustvari dajemo važnih informacija o nekom projektu prilikom upotrebe tih sistema postavljanjem samih upita, kopiranje dijela izvornog koda tražeći odgovor na greške pri kompajliranju ili u želji da unaprijedimo sam izvorni kod. To je posebna smjernica za dalje istraživanje obzirom da na tržištu rada postoje radne uloge, a vezuju se za poziciju QA inženjera[1]. Generalno o aspektima sigurnosti pri procesu softverskog inženjeringa će biti govora u nastavku rada.

Dakle, u cilju rezimiranja ovog poglavlja, autor je stava da se softversko inženjersko zasniva na određenim stubovima, odnosno oblastima unutar infromacionih tehnologija i da dobar softver inženjer treba da jako dobro poznaje ovih 5 stvari, a to su: operativni sistemi, računarske mreže, baze podataka, programiranje i analiza i dizajn softvera. Ovdje nije navedena sajber sigurnost jer u nastavku ćemo vidjeti da je to nešto što se mora poštovati u svakoj fazi izrade softvera i sama izgradnja softvera nema smisla bez adekvatne sigurnosti. U većini slučajeva informacije su novac, pogotovo ako se nalaze u digitalnom obliku, obaveza je iste zaštititi.



Slika 2. Stubovi IT-a (izvor: autor, AI generisana slika)

2. DIGITALNA SIGURNOST

Bilo koju digitalnu aktivnost da započnemo, teško da možemo zaobići pojam sigurnosti jer većina procesa koji se digitalizuju (poslovne ili nekim druge aplikacije) zahtijevaju velik stepen sigurnosti. U digitalnom dobu je nezamislivo napraviti nešto da nije sigurno pogotovo u sferi poslovnih aplikacija gdje je jako bitno sačuvati integritet te urediti autorizaciju i autentifikaciju. U nastavku govorimo o *security by design or default principima i NIS2 direktivi.*

2.1. Security by design or default

Danas kada se bavimo aspektima sajber sigurnosti to svakako nije više stvar nečega što se radi retroaktivno i nešto što ima ad-hoc pristup. Kada smo ranije govorili o fazama u stvaranju

[1] Inženjeri zaduženi za osiguranje kvalitete, u softverskoj industirji ih nazivaju još i „testerima“

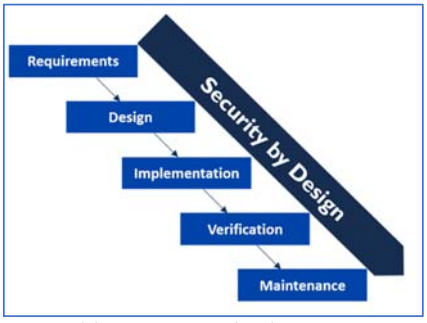
kvalitetnog softvera, pogotovo o fazi dizajna, svakako je podrazumijevano da se u toj fazi bavimo i načinima kako zaštiti naš budući informacioni sistem odnosno softverski produkt bilo kojeg tipa od neželjenog pristupa u cilju čuvanja integriteta i neželjenog curenja važnih informacija. U procesu digitalizacije niko nema za cilj napraviti određeno softversko rješenje i pored toga raditi dupli posao i voditi evidencije „na papiru“. Svi stari podaci sa papira postaju digitalni i briga o njihovoj sigurnosti je neprekidan proces koji nikada na prestaje i postaje svojevrstan izazov da se isti sačuvaju od naprijed pomentih stvari.

Security by design princip se odnosi na to da u svakoj fazi softverskog projekta razmišljamo o potencijalnim sigurnosnim prijetnjama i da koristimo najbolja znanja i najnovije dostupne metode zaštite budućih softverskih proizvoda i rješenja. Da li se tu radi o najboljim enkripcijskim algoritmima odnosno metodama šifrovanja, politici postavljanja jakih lozinki, više faktorskoj autentifikaciji ili automatskim ažuriranjima sistemskog softvera odnosno ažuriranja aplikacije, manje je više bitno, bitno je da se obuhvate svi aspekti i analiziraju sve do tada poznate potencijalne prijetnje kako bi se djelovalo preventivno.

Security by default znači da sve to što smo ranije planirali podrazumijevano i imamo uključeno. Dakle, kada korisnik (pretpostavimo da nema veliko znanje o sajber sigurnosti) počne da koristi softverski produkt, ne razmišlja mnogo o sigurnosti već da ona bude podrazumijevano na najvećem mogućem tehničkom nivou. To znači da prilikom upotrebe toga softverskog produkta podrazumijevano koristi najnovije i najbolje sigurnosne protokole, enkripcijske algoritme, strogu ili najstrožiju politiku generisanja i korištenja lozinki, itd.

Slijedeći navedene principe mnogo stvari će biti spriječeno preventivno i korisnici neće morati naknadno da se brinu o sigurnosti već će sistem osim što ispunjava funkcionalne zahtjeve biti i siguran. Dakle, od najranijih faza dizajna sistema razmišlja se o sajber sigurnosti pa do same implementacije i evaluacije. Ovaj pristup ne samo da štedi vrijeme i čini sistem sigurnijim od početka već štedi i novac jer pitanje sajber sigurnosti je nedvojbeno da li treba i stvar je samo hoće li se uraditi odmah ili naknadno. U slučaju naknadnog bavljenja ovom problematikom raste i ukupna cijena određenog softverskog produkta što je svakako posebna tema za istraživanje. [1]

Istraživanjem na ovu temu možete pronaći i razne kurseve, a odnose se na pricnicpe security by design i kako ih uključiti u rane projektne faze.[5]



Slika 3. Security by design [5]

2.2. NIS2

NIS2 direktiva (eng. Directive on security of network and information systems) počela se primjenjivati početkom 2023. godine, a pojavila se kao nasljednik prethodne NIS direktive u cilju smanjena rizika od sajber napada i povećanja otpornosti na sajber napade. Sama potreba za uvođenjem nove direktive ukazuje da su bili postojani problemi pri vođenju prethodnom direktivom ili bolje rečeno da su se države članice (EU) različito vodile i različito implementirale pomenutu direktivu. Kao novija i naprednija, revidirana verzija pod nazivom NIS2 donosi sa sobom strožija pravila po pitanju implementacije mjera iz oblasti sajber sigurnosti. U zavisnosti čime se određen poslovni subjekat bavi, shodno tome postoje i standardi koje treba poštovati, na vrhu ljestvice su svakako poslovni subjekti čiji prekid poslovanja uzrokovan sajber incidentom može da ugrozi zdravlje i sigurnost.

Pored analize rizika informacione sigurnosti na vrhu ljestvice minimalnih mjera nalaze se pojmovi kao što su kriptografija i enkripcija i plan aktivnosti koje će se poduzeti nakon što eventualno nastane sigurnosni incident, odnosno kako adekvatno upravljati kriznim situacijama. Dakle, sve se čini da do sigurnosnog incidenta ne dođe ali se ostavlja pretpostavka da do istog može doći i radi se na spremnosti kako taj incident što bezbolnije prevazići. Kazne za nepoštivanje propisa ove direktive su jako stroge, a akcenat se stavlja na kaznenu odgovornost kompanije i rukovodstva, a obavezne su i edukacije iz oblasti sajber sigurnosti jer sigurnost je proces, a ne jednokratan zadatak. [3]

Ova direktiva jasno nalaže da se svaka članica donese strategiju za sajber sigurnosti, odnosno donese istoimeni Zakon i podzakonske akte u određenom roku (koji je istekao 17.10.2024. godine) i po zadnjim dostupnim informacija (07.05.2025. godine) Europska komisija poziva 19 članica koje to nisu uradile, dakle nisu prenijele NIS2 direktivu u nacionlno pravo da to urade u što kraćem roku, a ne duljem od 2 mjeseca, da poduzmu mjere i odgovore jer u suprotnom bit će podnijete prijave sudu Europske unije. Informacije radi, naša susjedna država Repubika Hrvatska nije na pomenutom spisku.[4]

Ono što je važno napomenuti, obzirom da digitalno poslovanje ne poznaje granice u smislu zvaničnog članstva EU jeste da zemlje koje posluju, odnosno pružaju usluge unutar EU, a ne nalaze se u EU vjerovatno će morati da ispune obaveze iz NIS2 kao da su EU. Iako nema nacionalnog Zakona usklađenog sa ovom direktivom, ne radi se samo o eventualnim kaznenim odredbama već je stvar i internacionalne reputacije ukoliko firma izvan EU posluje sa klijentima iz EU.

3. ZAKLJUČAK

Kvalitetan i siguran softver nije samo želja pojedinca, odnosno naručioca, to već postaje standard, a i Zakonska obaveza. Ne postupanje po tim standardima odnosno mjerama zaštite implicira moguću kaznenu odgovornost i lošu reputaciju na tržištu.

Ovaj rad je jasno odvojio sami proces kodiranja od inženjeringa, a faze razvoja softvera su obogaćene *security by design* principima koji će biti neminovni jer će olakšati primjenu nacionalnih Zakona, a u vezi sajber sigurnosti.
Svjesnost o ovome trebaju da imaju svi, pogotovo ljudi iz poslovnog svijeta.

Akademska zajednica tu igra važnu ulogu, pogotovo ako uzmemo rastući trend upisa na IT fakultete sa akcentom na softverski inženjering. Nijedan pojedinac u nastavnom procesu ne može niti smije zanemariti činjenicu sistemskog pristupa rješavanju bilo kojeg problema niti akcenat previše stavljati na proces kodiranja pri učenju o softverskom inženjeringu pogotovo

jer se programski jezici jako brzo mijenjaju. Ranije smo definisali stubove IT-a i baš svaki stub se naslanja na sajber sigurnost jer je to oblast koja počiva na tim stubovima. Skoro da nemamo danas nijednog softverskog rješenja, a da u pozadini nije podržan od strane operativnog sistema, da ne razmjenjuje informacije putem računarske mreže, da nema baza podataka gdje se nalaze podaci i da nije nastalo analizom i preciznim dizajnom sistema popraćenim adekvatnim kodiranjem.

Svakako softver inženjer može da radi po potrebi i kodiranje, ali koder nikako ne može, niti je softver inženjer.
Praviti kvalitetan softver pored zadovoljenja funkcionalnih zahtjeva podrazumijeva da treba biti i siguran. Sigurnost je standard, pojam koji se veže za kvalitet i reputaciju, a kako već spomenusmo, postaje i Zakonska obaveza

4. LITERATURA

[1] Alaina L. (2024.) - Security by Design or Default?, Gobalna zajednica IT profesionalaca ISACA, https://www.isaca.org/resources/news-and-trends/isaca-now-blog/2024/security-by-design-or-default?gad_source=1&gad_campaignid=21147302543&gclid=Cj0KCQjwiqbBBhCAARIsAJSfZkZ3YSH66W2wXJuWDcYt3T5cF6t2YNH4EAUQ7u2EpDmFW4Z-HFREfkoaAhGuEALw_wcB , pristupljeno 05.05.2025. godine

[2] L. Benet (2024.) - Faze i modeli životnog ciklusa razvoja softvera (SDLC), Obrazovna platforma GURU99, https://www.guru99.com/hr/software-development-life-cycle-tutorial.html , pristupljeno 04.05.2025. godine

[3] Službena pod stranica kompanije DUPLICO posvećena NIS2 direktivi (2025.), Što je NIS2 direktiva, https://nis2direktiva.hr/sto-je-nis2-direktiva , pristupljeno 06.05.2025.

[4] Službena web stranica Eurpopske unije (2025), Komisija poziva 19 država članica da u potpunosti prenesu Direktivu NIS2, https://digital-strategy.ec.europa.eu/hr/news/commission-calls-19-member-states-fully-transpose-nis2-directive , pristupljeno dana 06.05.2025. godine

[5] Službena web stranica sajber savjetovališta CETOME (2025.), Security-by-design training, https://cetome.com/training/security-by-design , pristupljeno 06.05.2025. godine

[6] Wim H. (2025.) - Koja je razlika između softverskog inženjerstva i programiranja?, IT Pedia – baza članaka, https://bs.itpedia.nl/2019/07/12/wat-is-het-verschil-tussen-software-engineering-en-programmeren/ , pristupljeno 04.05.2025. godine