Principi softverskog inženjerstva

Vežbe - II nedelja
Izrada SSU i prototipa aplikacije

Dražen Drašković, asistent
Elektrotehnički fakultet
Univerziteta u Beogradu

Projekat
Faze

- Izrada projektnog zadatka
- **Izrada SSU i prototipa aplikacije**
- Formalna inspekcija
- …

Slučajevi korišćenja

- Slučajevi korišćenja (*use-cases*) su način prikupljanja funkcionalnih zahteva sistema
- Slučaj korišćenja je scenario koji opisuje kako softver treba da se koristi u datoj situaciji
- Softverski inženjer (analitičar) treba da kreira skup scenarija na kojima će sistem biti izgrađen
- Da bi kreirao scenario, analitičar mora prvo da identifikuje različite vrste ljudi, koji koriste sistem

Scenario

- Scenario je niz koraka koji opisuje interakciju između korisnika i sistema
- Primer scenarija:
  - elektronska prodavnica
  - problem: šta ako podaci o kartici nisu tačni !?

Slučajevi korišćenja

- Slučaj korišćenja jeste skup scenarija povezanih jednim ciljem korisnika
- Najveća vrednost slučajeva korišćenja je u sadržaju, dok dijagram ima sporedni značaj

Sadržaj slučaja upotrebe

- Opisuje osnovni uspešan scenario
- Svaki korak je deo interakcije između učesnika i sistema
- Jedan korak pokazuje nameru učesnika, a ne način kako je ostvaruje!

Sadržaj slučajeva upotrebe - primer (1)

- Kupovina proizvoda
- Osnovni uspešan scenario:
  1. Kupac pregleda katalog i bira proizvode koje hoće da kupi
  2. Kupac završava pregledanja kataloga
  3. Kupac unosi podatke o isporuci (adresa, rok isporuke)
  4. Sistem prikazuje sve podatke o troškovima (uključujući poštarinu)
  5. Kupac unosi podatke o platnoj kartici

Sadržaj slučajeva upotrebe - primer (2)

6. Sistem proverava podatke o načinu plaćanja
7. Sistem potvrđuje prodaju

<u>Proširenja:</u>

3a. Kupac je redovan

.1: Sistem prikazuje tekuće stanje o isporuci, cenama i iznosu računa

.2: Kupac potvrđuje ili menja podrazumevane vrednosti, povratak u 6. korak

6a. Podaci o platnoj kartici nisu ispravni

.1: Kupac može ponovo uneti podatke o kartici ili prekinuti rad

Analiza

- Osnovni principi analize:
  - domen informacija mora biti dobro predstavljen i razumljiv
  - funkcije koje softver pruža, moraju biti definisane
  - ponašanje softvera (kao posledica eksternih događaja) mora biti predstavljeno
  - podela na delove koji otkrivaju hijerarhijske detalje (da bi se smanjila kompleksnost)
  - proces analize treba da se pomeri od osnovnih informacija ka detaljima implementacije

Principi

- Razumite problem pre nego što počnete da kreirate model analize
- Razvijte prototipove koje ćete prikazati korisniku
- Zahtevi korisnika
- Različiti pogledi na zahteve
- Rangirajte zahteve po prioritetu

Vaš zadatak
Šta uraditi?

- Za svaku funkciju opisanu u odeljku “Funkcionalni zahtevi” uraditi poseban SSU dokument i odgovarajuće forme korisničkog interfejsa
- Primeri SSU dokumenata:
  - Kreiranje korisnika od strane administratora
  - Dodavanje nove vesti
  - ...

Vaš zadatak
Ko treba da uradi?

- Svaki član tima uradi bar tri SSU dokumenta i prototip koji odgovara funkcijama opisanim tim SSU dokumentima
- Alternativa: jedan član tima pravi kompletan prototip, ostali dele SSU dokumenta

Prototip

- Vrste:
  - Papirni prototip
  - Računarski prototip
- Najčešći prototip kod Web aplikacija: statičke HTML stranice
- Stranice treba da budu povezane u celinu

Zašto koristimo prototip?

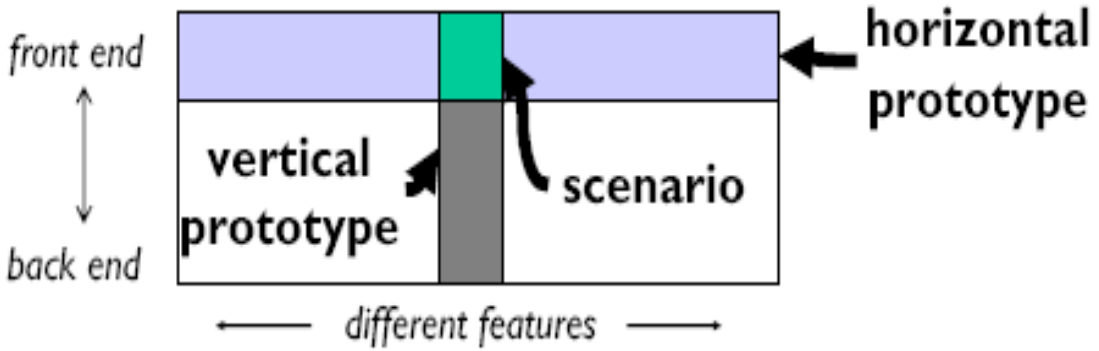
- Dobititi reakcije od naručioca ranije, jeftinije
- Eksperimentisati sa alternativama
- Jednostavnije promeniti ili odbaciti

Vernost prototipa

- Prototip male vernosti: izbeći detalje, koristiti jeftinije materijale ili druge implementacione tehnike
- Prototip velike vernosti: više podsećaju na završene projekte

Horizontalni i vertikalni prototip

- Po širini: % pokrivenosti traženih mogućnosti
- Po dubini: stepen funkcionalnosti
- Horizontalni prototip - u potpunosti po širini, veoma malo po dubini, to je frontend bez backend dela
- Vertikalni prototip - obrnuto, jedan deo interfejsa se implementira



Papirni prototip

- Interaktivni papirni prototip
  - Skiciranje izgleda ekrana
  - Delovi papira prikazuju prozore, menije, dijaloge
- Interaktivnost je prirodna
  - Pokret prstima simulira klik miša
  - Pisanje simulira unos sa tastature
- Čovek simulira operacije računara
  - Spuštanje i podizanje odgovarajućih delova
  - Ispis odgovora na “ekranu”
  - Opisivanje efekata koje je teško opisati na papiru
- Niska vernost u okviru izgleda i osećaja
- Visoka vernost u širini (samo se crtaju delovi aplikacije) i dubini (čovek simulira backend)

Zašto se nekad koristi papirni prototip?

- Brže se realizuje
  - skiciranje brže nego programiranje
- Jednostavnije se menja
  - Jednostavnije je praviti izmene između testova korisnika ili čak tokom testiranja korisnika
  - Nema proučavanja koda - sve se odbacuje (osim dizajna)
- Fokusira se pažnja na generalniji izgled
  - Nije bitan font, veličina, šema boja
  - Korisnik može pružiti kreativne sugestije
- Ne moraju da rade programeri - zahteva se mašta!

Prednosti papirnog prototipa

- Konceptualni model
  - Da li ga korisnici razumeju?
- Funkcionalnost
  - Da li radi kako treba? Da li nedostaju neke opcije?
- Navigacija i tok izvršavanja
  - Mogu li korisnici da se snađu?
  - Da li postoje preduslovi?
- Terminologija
  - Da li korisnici razumeju labele?
- Sadržaj ekrana
  - Šta treba da se nalazi na ekranu?

Nedostaci papirnog prototipa

- Izgled: boja, font, prazan prostor,...
- Osećaj - da li su elementi preblizu,...
- Vreme odgovora
- Da li se primećuju male promene?
- Korisnici su mnogo oprezniji,
  ne istražuju i ne greše u velikoj meri

Računarski prototip

- Interaktivna softverska simulacija
- Dobra mera za izgled i osećaj
- Loša mera za dubinu
  - Papirni prototip simulira čovek, kod računarskog tog dela nema
  - Računarski prototip je tipično **horizontalni**: pokriva mnoge opcije, ali ne i backend

Šta se može naučiti pomoću rač.prototipa?

- Sve što može pomoću papirnog plus:
  - Layout prozora
  - Boje, fontovi, ikone, drugi elementi...
  - Interaktivni feedback
  - Pravilan izgled
- Koriste se alati za prototip
  - brže nego kodirati
  - nema debagovanja
  - jednostavnije promene

Tehnike računarskog prototipa

- Storyboard - niz naslikanih ekrana (screen shot) povezanih hyperlink-ovima
- Realizacija formi - stvarni prozori dobijeni korišćenjem palete komponenti (dugmad, tekst polja, labele,...)
- Storyboard tehnike: HTML, Flash/Director, PowerPoint,...
  Svi ovi alati mogu da koriste i script jezike!
- HTML stranice i forme: dobar izbor ako se realizuje web aplikacija, ali u drugim slučajevima ne pružaju jasan osećaj izgleda

Tehnike početnog dizajna

- Tehnike početnog dizajna:
  - Skiciranje (smisliti i nacrtati sopstvene ideje na papiru ili tabli)
  - Scenario (priča kako da korisnik koristi sistem)
  - Storyboard (ilustracija scenarija)
- Koristite rezultati analize da saznate:
  - da li su pokriveni najvažniji zadaci?
  - koji su najvažniji aspekti upotrebe sistema?

Projekat – Faza 2

- U svakom SSU dokumentu navesti ime autora
- U svakom fajlu prototipa, u vidu komentara na početku navesti ime autora
- Sve dokumente i prototip komprimovati u arhivu i poslati na adresu: si3psi.etf@gmail.com
- Subject:
  TIM <ime_tima> SSU+PR verzija X.Y
- Krajnji rok za ovu fazu je četvrtak 14.3.2024. u 23:59.

Korisni linkovi

- Pencil v3.1: http://pencil.evolus.vn
- Figma: https://www.figma.com

BDD - Behavior driven development

- Identifikacija poslovnih ciljeva
- Svaka funkcija predstavlja priču
- BDD pristup: definisati i identifikovati priču, kao i kriterijume prihvatljivosti
- Razlozi uvođenja ove metodologije:
  - Programeri - da razumeju probleme
  - Testeri - da lako pišu testove
  - Menadžer projekta - da razume sve kao celinu
  - Klijenti/kupac proizvoda - da pojasne korisničke zahteve, ako nešto nije dovoljno razjašnjeno

BDD - Behavior driven development

- Efikasno pretvoriti ideju u delo - implementiran, testiran i za produkciju spreman kod
- Opisati korisničke zahteve svima - poslovnim ljudima, analitičarima, programerima, testerima
- Cilj da se izbegne:
  - Identično razumevanje - „Sve je gotovo“
  - Kontradiktornost - „To nije ono što sam tražio“
  - Zamke - „Zaboravio sam još nešto da vam kažem o toj drugoj stvari“

Uloga priče

- Detaljan opis zahteva i poslovne koristi
- Set kriterijuma
- Agilne definicije:
  - „obećani dogovor“
  - „opis svih karakteristika/funkcionalnosti“
- BDD priča može takođe opisati i nefunkcionalne zahteve, sve dok aktivnost ima cilj, obim posla (trajanje) i da može da se postigne dogovor

Struktura priče

```
Title (one line describing the story)
Narrative:
As a [role]
I want [feature]
So that [benefit]

Acceptance Criteria: (presented as Scenarios)

Scenario 1: Title
Given [context]
  And [some more context]...
When  [event]
Then  [outcome]
  And [another outcome]...

Scenario 2: ...
```

Primer priče (1)

```
Story: Account Holder withdraws cash

As an Account Holder
I want to withdraw cash from an ATM
So that I can get money when the bank is closed

Scenario 1: Account has sufficient funds
Given the account balance is \$100
 And the card is valid
 And the machine contains enough money
When the Account Holder requests \$20
Then the ATM should dispense \$20
 And the account balance should be \$80
 And the card should be returned
```

Primer priče (2)

```
Scenario 2: Account has insufficient funds
Given the account balance is \$10
 And the card is valid
 And the machine contains enough money
When the Account Holder requests \$20
Then the ATM should not dispense any money
 And the ATM should say there are insufficient funds
 And the account balance should be \$20
 And the card should be returned

Scenario 3: Card has been disabled
Given the card is disabled
When the Account Holder requests \$20
Then the ATM should retain the card
And the ATM should say the card has been retained

Scenario 4: The ATM has insufficient funds
...
```