Rukovalac dokumentima

(RuDok)

Specifikacija projekta

Generacija: 2021/2022

SADRŽAJ

[**Predgovor** 3](#_Toc48409150)

[SOFTVERSKI PROCES: 3](#_Toc48409151)

[SOFTVERSKI PROIZVOD: 4](#_Toc48409152)

[**1. Kako FORMALIZOVATI IDEJU? - Osnovni koncepti** 5](#_Toc48409153)

[**2. DOMEN REŠENJA - šta treba da istakne rešenje koje je predmet razvoja?** 6](#_Toc48409154)

[**3. OPŠTE OSOBINE ALATA - GUI KOMPONENTA** 7](#_Toc48409155)

[**4. Prototipski razvoj - Evolutivni prototip** 8](#_Toc48409156)

[5. Prilozi 9](#_Toc48409157)

[**5.1. Format modela zahteva** 9](#_Toc48409158)

[**5.2. Primer modela zahteva** 10](#_Toc48409159)

[5.3. Modelovanje arhitekture programskog proizvoda na bazi orkestracije komponenti 11](#_Toc48409160)

[5.3.1. Formulisanje organizacije artifakata aplikativnog radnog okvira 11](#_Toc48409161)

[5.3.2. Primer - Objektni model Arhitekture Aplikacije (Aplikativni radni okvir) 12](#_Toc48409162)

[5.3.3. Primer - Dijagram komponenti 13](#_Toc48409163)

[**5.4. Model statičke strukture komponenti - UML Dijagram KLASA** 14](#_Toc48409164)

[5.4.1. Primer nacrta modela DOKUMENTA 14](#_Toc48409165)

[**5.5. Model komponente - Python** 15](#_Toc48409166)

[**5.6. Elementi primopredaje** 18](#_Toc48409167)

[5.6.0. Organizacija u timskom radu (proces) - oslonac na Microsoft Teams 18](#_Toc48409168)

[5.6.1. Vizija softverskog proizvoda - tekstualni dokument koji sadrži specifikaciju ideje 18](#_Toc48409169)

[5.6.2. Specifikacija zahteva - Astah Professional - model zahteva prema opisanom standardu 18](#_Toc48409170)

[5.6.3. Model Arhitekture - Astah Professional - objektni (komponent orujentisani) model. 18](#_Toc48409171)

[5.6.4. Modeli Komponenti - Astah Professional - objektni (klasni) model 18](#_Toc48409172)

[5.6.5. Izvršna verzija aplikacije - Rukovalac dokumentima 19](#_Toc48409173)

[5.6.6. Dokumentovanje aplikacije - Rukovalac dokumentima 19](#_Toc48409174)

# **Predgovor**

Predmet **Metodologija razvoja softvera** je sintetski predmet sa misijom podizanja nivoa veština i znanja potrebnih za izradu složenih softverskih proizvoda koje uključuju razvoj metoda i tehnika:

1. modelovanja i izgradnje fleksibilne arhitekture
2. komponentizacije i
3. upotrebe šablona za refaktorisanje interne arhitekture komponenti složenih softverskih proizvoda.

# SOFTVERSKI PROCES:

* + ***Model životnog ciklusa***:
    - **Agilni pristup razvoju softvera** (Agile Software Development)
      * ***Timski rad***
        + ***rad u timovima*** (Team Based Software Development, Pair Programming)
        + brain storming tehnika specifikacije zahteva i dizajna
        + praćenje aktivnosti članova tima (Activity Tracking)
    - ***Prototipski razvoj*** - evolutivni prototip
  + ***Metodologija - Modelom upravljani razvoj softvera:***
    - ***Izrada vizije softverskog proizvoda i njeno formalno predstavljanje*** (oslonac na ranije usvojene formalizme - tekstualna specifikacije);
    - ***Specifikacija i modelovanje zahteva*** *(uz oslonac na usvojeni alat za modelovanje);*
    - ***Specifikacija i modelovanje interakcija*** *(UML - USE CASE formalizam uz oslonac na usvojeni alat za modelovanje)*
    - ***Specifikacija i modelovanje arhitekture -*** *(model komponenti)*
    - ***Specifikacija i modelovanje statičke strukture komponenti*** *- (UML - KLASNI MODELI uz oslonac na usvojeni alat za modelovanje)*
    - ***Specifikacija i modelovanje ponašanja*** *- (UML dinamički dijagrami odabrani iz skupine usvojenih formalizama za opis ponašanja softverskih sistema - dijagram stanja, dijagram aktivnosti, dijagram sekvence i dijagram saradnje)*

# SOFTVERSKI PROIZVOD:

* + ***Interaktivan, GUI orijentisan, Događajima upravljan programski Alat***
  + ***Standardizacija arhitekture***:
    - ***Spoljašnja arhitektura*** - dinamička kolekcija GUI komponenti
      * Komponente koje manipulišu sa radnim okruženjem:
      * Komponente koje manipulišu sa ***artifaktima*** ***softverskog proizvoda*** (proizvodi koje softverski proizvod kreira i sa njima rukuje)
      * Uputstva i pomoć pri ovladavanju programskim proizvodom;
    - ***Unutrašnja arhitektura komponenti***:
      * MVC (Model-View-Controller) - arhitektonski obrazac
      * Refaktorizacija unutrašnje arhitekture - upotreba šablona (*Observer, Singletone, State, Composite, Factory Method, Abstract Factory, Proxy, Bridge, Command)*
    - ***Podešavanje programskog proizvoda*** (*Parametrizacija, Lokalizacija i personalizacija*)
      * Logotip i o aplikaciji
        + ***Algoritmi, strukture podataka i organizacija podataka***
    - ***Model dokumenta***
    - ***Serijalizacija i deserijalizacija - (****čuvanje radnog prostora, kolekcija dokumenata, modela dokumenata i instanci modela dokumenata****)***
  + ***Prototip*** *- Evolutivni prototip uz oslonac na Objektno Orijentisanu Paradigmu (slobodan izbor* ***jezika*** *i* ***grafičke biblioteke*** *i* ***razvojnog okruženja*** *u skladu sa ranije usvojenim znanjima i veštinama)*

# **1. Kako FORMALIZOVATI IDEJU? - Osnovni koncepti**

Rukovalac dokumentima: (RuDok)

* **Domen problema: - Dizajniranje i operativna upotreba dokumenata** (*Dokument - složeni proizvod sa potencijalno miltimedijalnim sadržajem.*)
* **Zainteresovane strane: - *Organizacioni sistemi i pojedinci koji u osnovi svoje delatnosti imaju rad sa dokumentima slobodne strukture i organizacije.***
  + **Koga kontaktirati u cilju smanjenja stepena nepoznavanja domena problema i domena rešenja?**
    - **Eksperte iz domena problema -** operativno se bave kreiranjem dokumenata (dizajneri dokumenata) i poseduju neophodna iskustva vezana za domen izvršavanja;
    - **Eksperte iz domena upotrebe -** operativni korisnici kreiranih dokumenata;
    - **Poslovni eksperti** - poseduju operativne pokazatelje (podatke/informacije/bazu znanja) vezane za poslovne aspekte softverskog proizvoda;
    - **Eksperte za razvoj softvera -** vladaju konceptima izgradnje fleksibilnih programskih proizvoda.
* **Vrsta proizvoda koji želimo razviti: - Alat (proširivi radni okvir)**
* **Način plasmana: Slobodno tržište - ne projektuje se za poznatog kupca.**

# **2. DOMEN REŠENJA - šta treba da istakne rešenje koje je predmet razvoja?**

* Rukovanje Dokumentima - Opis, definisanje, operativna upotreba i trajno čuvanje dokumenta proizvoljne strukture;
* Rukovanje Kolekcijama dokumenata - Slobodno formiranje kolekcija arhitekture dokumenata u skladu sa modelom dokumenta (Prilog 1.);
* Rukovanje Radnim prostorima alata - Slobodno formiranje radnih prostora koji sadrže kolekcije dokumenata;
* Delenje celih dokumenata - Dokumente je moguće deliti između kolekcija dokumenata;
* Delenje delova dokumenata - Stranice je moguće deliti između različitih dokumenata;
* Delenje delova stranica - Delove stranica je moguće deliti između različitih Stranica ili Dokumenata;
* Rukovanje Parametrima Komponenti Arhitekture Softverskih Proizvoda

# **3. OPŠTE OSOBINE ALATA - GUI KOMPONENTA**

Opšta Arhitektura GUI komponente softverskog proizvoda:

1. **Naslovna linija ( *titlebar* )**

2. **Glavna linija menija ( *menubar* )**

3. **Paleta alatki ( *toolbar* )**

4. **Radna površina ( *workspace*)**

5. **Statusna linija ( *statusbar* )**

****

# **4. Prototipski razvoj - Evolutivni prototip**

# 5. Prilozi

## **5.1. Format modela zahteva**

Model zahteva je neophodno predstaviti u donjoj tabelarnoj formi:

Tabela 5.1.1. Format modela zahteva

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| R.br. | Opis | Tip | Prioritet | Zainteresovane strane | Rizik | Status | Procena napora |
| 1. | *Tekstualna formulacija zahteva* |  |  |  |  |  | *Efektivni broj dana* |
| 2. |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| n- |  |  |  |  |  |  |  |

|  |  |
| --- | --- |
| Status | Značenje |
| Dr | Draft |
| De | Definisan |
| Po | Potvrđen |

|  |  |
| --- | --- |
| Rizik | Značenje |
| Vi | Visok |
| Sr | Srednji |
| Ni | Nizak |

|  |  |
| --- | --- |
| Oznaka | Naziv |
| 1 |  |
| 2 |  |
| 3 |  |
|  |  |
| k |  |

|  |  |
| --- | --- |
| Prioritet | Opis tipa |
| ES | Esencijalni |
| VP | Vrlo poželjan |
| PO | Poželjan |
| OP | Opcion |
| NP | Nepoželjan |

|  |  |
| --- | --- |
| Tip | Opis tipa |
| F | Funkcionalni |
| N | Nefunkcionalni |

Model zahteva treba da ima potpun obuhvat (svi grupi zahtevi) uz specificiranje detalja u skladu sa rezultatima pojedinačnih etapa u procesu prikupljanja i analize zahteva.

## **5.2. Primer modela zahteva**

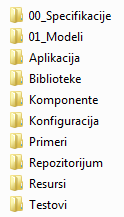
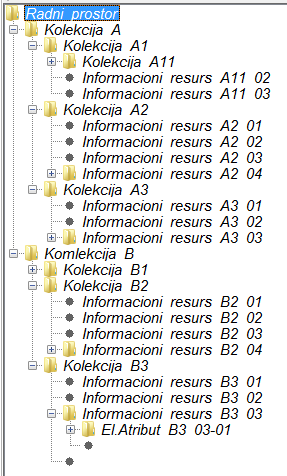
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| R.br. | Opis | Tip | Prioritet | Zainteresovane strane | Rizik | Status | Procena napora |
| 1. | **Instalacija**  Neophodno je obezbediti podršku instalaciji PROGRAMSKOG PROIZVODA. | N | Op | Administrator alata | Vi | De | *35* |
| 2. | **Pokretanje programskog proizvoda**  Neophodno je obezbediti pokretanje programskog proizvoda na način kako je to definisano prilikom njegove instalacije. Prilikom pokretanja neophodno je dovesti programski proizvod u kontekst u kom je ostavljen kod poslednjeg prekida rada. | N | Es | Operativni korisnik | Vi | De | 12 |
| 3. | **Operativna upotreba programskog proizvoda** | F | Es | Operativni korisnik | Vi | Dr | 0 |
| 4. | **Prekid rada programskog proizvoda**  Neophodno je, u bilo kom trenutku operativne upotrebe, obezbediti kontrolisani prekid rada programskog proizvoda. Prilikom prekida rada neophodno je očuvati konzistentnost svih artifakata sa kojima je manipulisano u toku rada. Potrebno je sačuvati kontekst upotrebe programskog alata u cilju njegove restauracije kod sledećeg pokretanja. | F | Es | Operativni korisnik | Vi | De | 12 |
|  | . |  |  |  |  |  |  |
|  | . |  |  |  |  |  |  |
|  | . |  |  |  |  |  |  |
| n- |  |  |  |  |  |  |  |

### 5.3. Modelovanje arhitekture programskog proizvoda na bazi orkestracije komponenti

### 5.3.1. Formulisanje organizacije artifakata aplikativnog radnog okvira

Gradivne elemente aplikativnog radnog okvira je potrebno organizovati u sklopu strukture namenskih kataloga (mogući primer):

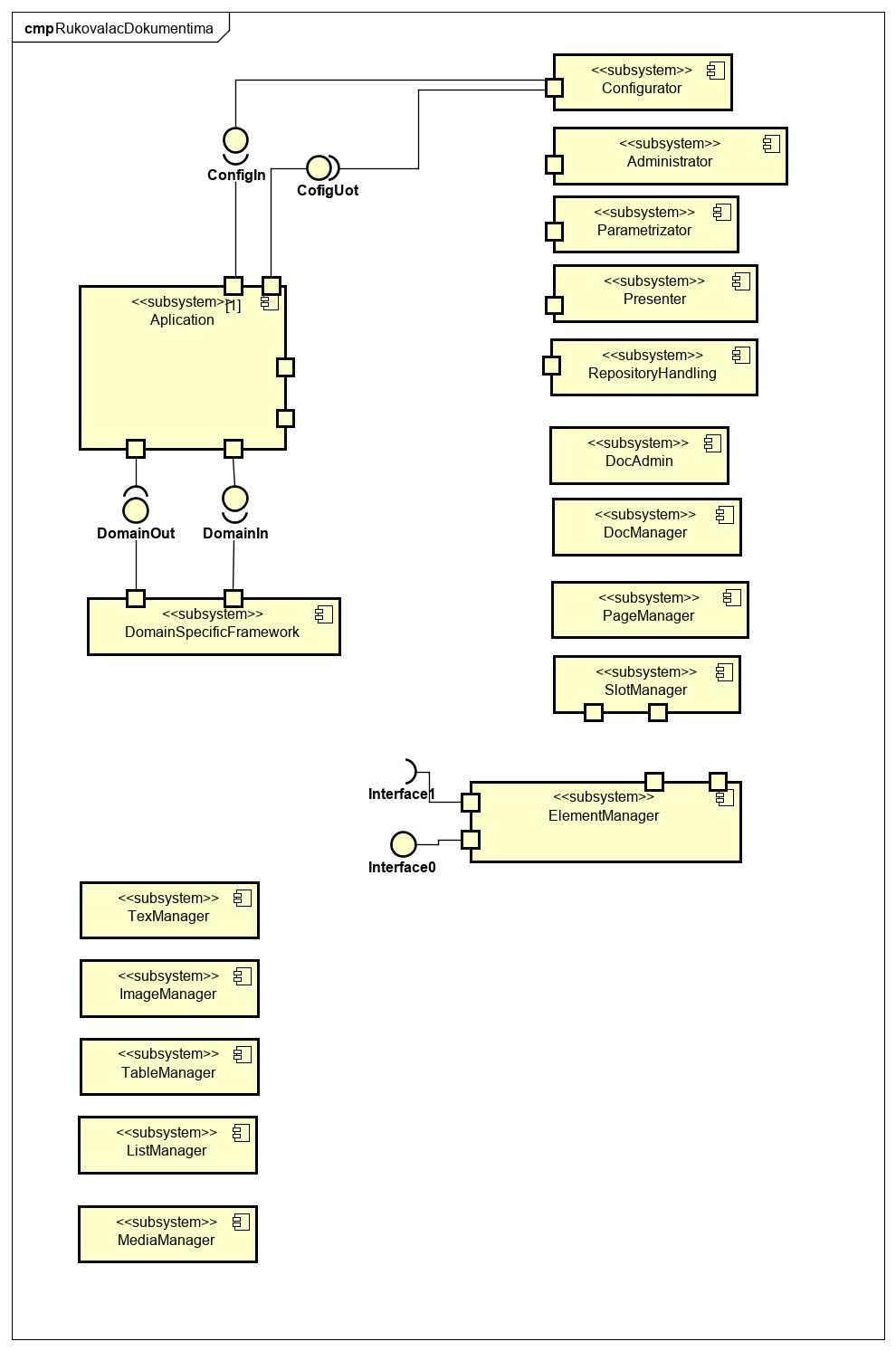




### 5.3.2. Primer - Objektni model Arhitekture Aplikacije (Aplikativni radni okvir)



### 5.3.3. Primer - Dijagram komponenti

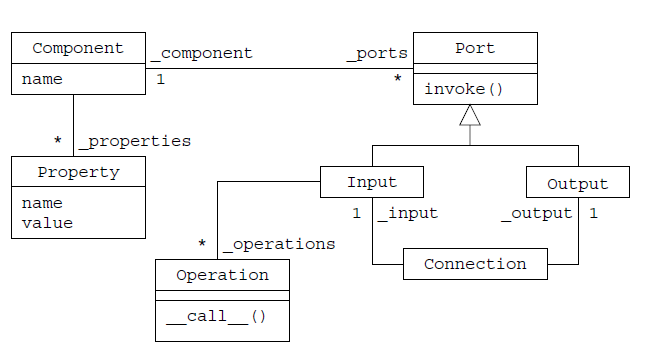


## **5.4. Model statičke strukture komponenti - UML Dijagram KLASA**

### 5.4.1. Primer nacrta modela DOKUMENTA

****

## **5.5. Model komponente - Python**



class Component :

def \_\_init\_\_ ( self , name , ports =None , properties = None ):

self . \_name = name

if not ports :

ports = dict ()

self . \_ports = ports

if not properties :

properties = dict ()

self . \_properties = properties

def bind\_port ( self , name , ref ):

self . \_ports [ name ] = ref

ref. \_component = self

def unbind\_port ( self , name ):

if name not in self . \_ports :

raise UnknownPort , name

self . \_ports [ name ] . \_component = None

del self . \_ports [ name ]

class Input ( Port ):

def \_\_init\_\_ ( self , ops = None ):

Port . \_\_init\_\_ ( self )

if not ops:

ops = dict ()

self . \_operations = ops

def get\_operations ( self ):

return self . \_operations

def invoke ( self , opname , args ):

if not opname in self . \_operations :

raise UknownOperation , opname

if not self . \_component :

raise UnboundPort

op = self . \_operations [ opname ]

return op( self . \_component , args )

class Output ( Port ):

def \_\_init\_\_ ( self ):

Port . \_\_init\_\_ ( self )

self . \_ref = None

def invoke ( self , opname , args ):

if not self . \_ref :

raise PortNotConnected

return self . \_ref . invoke ( opname , args )

class Operation :

def \_\_init\_\_ ( self ):

pass

def \_\_call\_\_ ( self , cmp , args ):

pass

class Connection :

def \_\_init\_\_ ( self , oport , iport ):

self . \_input = iport

self . \_output = oport

def connect ( self ):

if not self . \_input or not self . \_output :

raise ConnectionNotConfigured

if not isinstance ( self . \_input , Input ) or \

not isinstance ( self . \_output , Output ):

raise ImproperConnection

self . \_output . \_ref = self . \_input

def disconnect ( self ):

self . \_output . \_ref = None

## **5.6. Elementi primopredaje**

### 5.6.0. Organizacija u timskom radu (proces) - oslonac na Microsoft Teams

Specifikacija aktivnosti. pregled aktivnosti, specifikacija verzija softverskog proizvoda(Git repozitorijum), podela posla i statistika izgradnje softverskog proizvoda. Timovi realizuju dve grupe aktivnosti:

Edukativne - Plan aktivnosti edukativne prirode (aktivnosti na predavanjima, aktivnosti na vežbama, edukativni zadaci za evaluaciju postignuća,

Projektne - Pojedinačne aktivnosti na realizaciji Semestralnog projekta, podela komponenti, praćenje integracije (GIT- kontinuirana integracija)

### 5.6.1. Vizija softverskog proizvoda - tekstualni dokument koji sadrži specifikaciju ideje

Upotrebiti Office paket. Dokument treba da sledi standard definisan u sklopu predavanja. Dokument uključiti u Microsoft Teams za pojedinačne timove.

### 5.6.2. Specifikacija zahteva - Astah Professional - model zahteva prema opisanom standardu

Zahteve treba upotpuniti Poslovnim i Sistemskim slučajevima upotrebe (UML USE CASE dijagrami. Neophodno je jasno definisati veze između pojedinačnih zahteva i odgovarajućih UML USE?CASE dijagrama.

### 5.6.3. Model Arhitekture - Astah Professional - objektni (komponent orujentisani) model.

Oslonac na podsisteme, komponente , portove i interfejse komponenti. Formiranje Aplikativnog Radnog Okvira .

Modelovanje saradnje komponenti uz oslonac na specifikaciju interfejsa, modelovanje saradnje (definisanje protokola saradnje komponenti), definisanje razmene objekata između komponenti arhitekture (modelovanje razmene poruka - dijagrami sekvence)

### 5.6.4. Modeli Komponenti - Astah Professional - objektni (klasni) model

Oslonac na Modelovanje statičke strukture (klasni dijagrami) sa upotrebom MVC arhitektonskog šablona i odgovarajućih Dizajnerskih šablona. Specifikacija i modelovanje veza komponenti sa okruženjem (Interfejsi, Implementacija interfejsa).

Oslonac na modelovanje ponašanja. Dominantno unutrašnju dinamiku bazirati na upotrebi dijagrama stanja i šablona koji ga prate (State, Command, Structure, Factory, . . .). Pojedinačne komponente moraju slediti formu komponent radnog okvira. Unutrašnja arhitektura MVC sa upotrebom Observera. Unikat koristiti za enkapsulaciju deljenih elemenata unutar pojedinačnih komponenti.

Oslonac na dijagrame aktivnosti za specifikaciju saradnje objekata unutar komponenti.

### 5.6.5. Izvršna verzija aplikacije - Rukovalac dokumentima

Upakovana (rar ili zip) izvršna verzija sa kompletnim repozitorijumom.

### 5.6.6. Dokumentovanje aplikacije - Rukovalac dokumentima

ON\_LINE - Operativno uputstvo (Help) i About box.

OF\_LINE - Elektronska forma operativne dokumentacije (izvoz u pdf)