3.7.2002 CVVS 2-2000

Interni standard za razvoj in preverjanje programske opreme

Tomaž Dogša

1. Splošne zahteve

1.1. Standard je namenjen majhnim projektom, ki jih izvajajo študenti elektronike v okviru seminarskih nalog in diplom. Namen standarda je dvig kakovosti produktov in poenostaviti njihovo vzdrževanje.

1.2. Standard je napravljen za projekte, ki se razvijajo po modelu, ki ga prikazuje slika 1 in 2.

1.3. Vsak projekt mora imeti svoje ime oziroma kodo, ki je definirana v Projektnem planu.

1.4. V projektu se uporablja naslednja dokumentacija:

D1. Naročnikove zahteve
D2. Plan projekta
D3. Sistemske specifikacije
D4. Testni primeri
D5. Poročilo o preverjanju
D6. Načrtovalska dokumentacija

D7. Uporabniški priročnik
D8. Izvorna koda
D9. Izvršilna koda
D10. Opis in organizacija datotek, ki so na disketi

1.5. Standard se lahko izjemoma po potrebi poenostavi. Izpustijo se lahko nekateri dokumenti. Minimalni nabor dokumentov je D7, D8, D9, D10.

1.6. Vsak dokument mora vsebovati med ostalim:

1. Ime projekta in verzijo dokumenta
2. Reference na dokumente, na podlagi katerih je nastal
3. Podpis osebe, ki je odgovorna za pregled dokumenta (to se lahko nahaja tudi v Projektnem planu)

2. Zahteve glede programa

(Glej tudi točko Izvorna koda v nadaljevanju)

2.1. Uporabnost

2.1.1 Vsak program mora imeti izpis pomoči.

2.2.2 Program mora biti odporen na napačen oziroma pomanjkljiv vnos podatkov.

2.2. Testabilnost:

2.2.1. V izvorni kodi mora biti oznaka za verzijo, ki se mora skladati s tisto, ki je v dokumentih. Uporabnik mora imeti možnost, da to oznako izpiše.

2.2.2. Največja logična kompleksnost katerekoli funkcije v izvorni kodi ne sme presegati 20. Logična kompleksnost je število odločitvenih stavkov.

2.2.3. Program naj deluje najmanj v dveh režimih: v normalnem ali v testnem. Režim izbiramo z določenim parametrom ali spremenljivko okolja. Če ta parameter ni nastavljen, program deluje v normalnem načinu.

2.2.4. Vsak menu oziroma izpis, ki vsebuje več kot dve vrstici, mora biti označen (npr. M10). Ta oznaka se pravilom izpiše samo v testnem načinu.

3. Procesne zahteve

3.1. Razvojni model

Standard je napravljen za razvojni model, ki ga prikazuje slika 1 ali slika 2.

3.2. Preverjanje in odpravljanje neustreznosti

3.2.1. Prototip preveri avtor s testnimi vzorci, ki jih pripravi preverjevalec.

3.2.2. Skladnost s standardom preverja preverjevalec.

3.2.3. Ciklus preverjanja-popravljanja se mora izvršiti najmanj dvakrat. (Preverjevalna skupina dobi najmanj dve verziji programa in drugih objektov, da jih preveri.)

3.2.4. Preverjevalec mora vedno preveriti ali so neustreznosti v resnici odpravljene.

3.3. Temeljitost testiranja

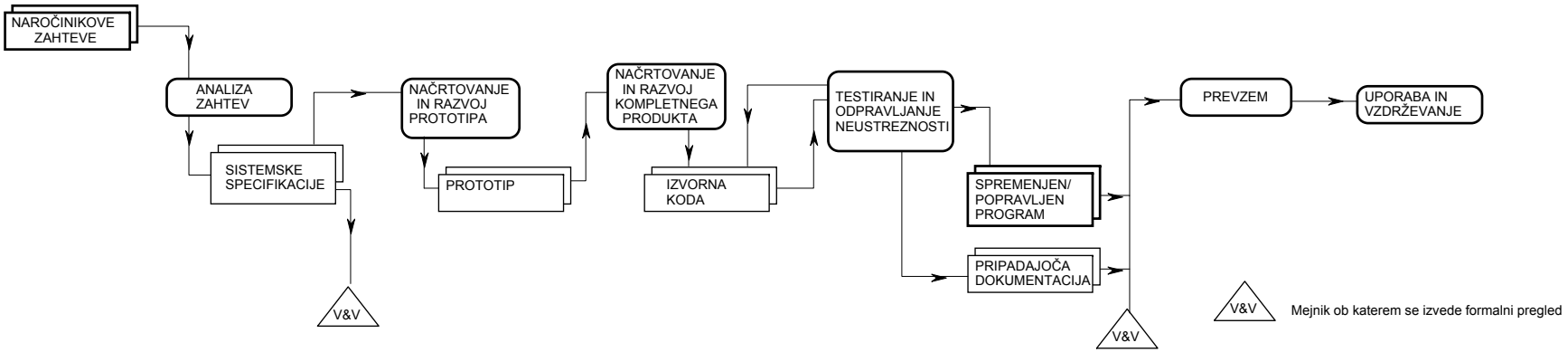
Za zadosten kriterij preverjenosti se uporabljajo naslednja merila:

3.3.1. 100% pokritost zahtev oziroma funkcij (tvoriti moramo toliko testnih vzorcev, da se bo videlo, da je vsaka zahteva oz. funkcija implementirana).

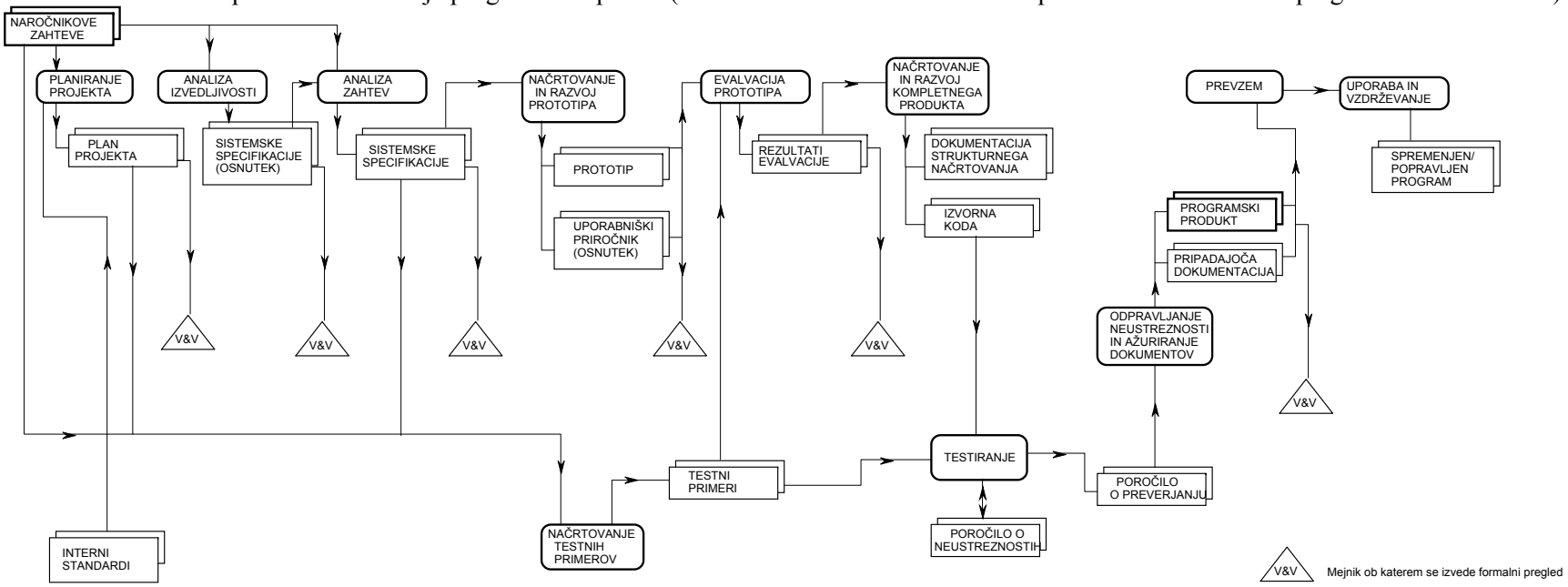
3.3.2. 100% pokritost rutin (vsaka rutina mora biti klicana najmanj enkrat).

Za zelo kakovostne programe se naj namesto pokritosti rutin uporablja pokritost stavkov:

3.3.3. 100% pokritost stavkov (vsak stavek mora biti izvršen najmanj enkrat).



Slika 1 Preprost model razvoja programske opreme (iz vsake faze se lahko vrnemo na predhodne faze - zaradi preglednosti ni narisano)



Slika 2 Izbran model razvoja programske opreme (iz vsake faze se lahko vrnemo na predhodne faze - zaradi preglednosti ni narisano)

4. Struktura posameznih dokumentov

4.1. Naročnikove zahteve

To je seznam oštevilčenih zahtev, ki jih sestavi naročnik. Zahteve so lahko formulirane tekstovno oziroma grafično.

Struktura dokumenta:

1. Ime dokumenta
2. identifikacijski podatki (kdo je naročnik, datum, verzija)
3. seznam oštevilčenih zahtev

4.2. Plan projekta

Namenjen je osebju, ki mora imeti pregled nad projektom.
Struktura dokumenta:

1. Identifikacija projekta, povezava z drugimi dokumenti.
2. Kratek opis problema
   - 2.1. globalni cilji (globalne zahteve), ki jih želimo s produktom doseči,
   - 2.2. omejitve (operacijski sistem, aparaturna oprema, standardi...)
   - 2.3. rok za zaključitev projekta, skupni stroški
   - 2.4. funkcije (bistvene funkcije, ki jih mora sistem izvajati, da bodo globalni cilji doseženi)
   - 2.5. pomembne karakteristike (katere karakteristike so za končni produkt zelo pomembne)
3. Zagotavljanje kakovosti (standardi, ki jih bomo upoštevali oziroma katere aktivnosti bomo izvajali).
   - 3.1. Načrt preverjanja
   - 3.3.1. Kaj je potrebno preveriti?
   - 3.3.2. Kako bomo preverjali (metode, strategije)?
   - 3.3.3. Po kakšnih kriterijih bomo preverjali (odpovedna kriterijska funkcija)?
   - 3.3.4. Kdaj bomo končali s preverjanjem?

4. Naloge in rezultirajoči dokumenti (izbira razvojnega modela)
5. Resursi
   - 5.1. Osebje (Kdo bo sodeloval, kakšna je njegova vloga, kakšne morajo biti njegove sposobnosti?)
   - 5.2. Potrebna programska orodja, knjižnice
   - 5.3. Potrebna aparaturna oprema
6. Razdelitev stroškov
7. Terminski plan (Pregledna tabela oz. graf, kjer navedemo, kdaj bodo končane posamezne faze in kdaj predani dokumenti, dodamo še podpise oseb, ki so zadolžene za pregled določenega dokumenta).
8. Pojmovnik
9. Priloge (opisi uporabljenih testirnih strategij)

4.3. Sistemske specifikacije

Sistemske specifikacije napravimo v dveh korakih:

1. najprej napravimo **analizo izvedljivosti** (poiščemo okvirno rešitev problema, določimo kaj moramo napravili sami oziroma česar ni treba, ker je že razvito). Tako dobimo **osnutek**, ki ga pregleda naročnik. Ko ga ta potrdi, sledi

2. **analiza zahtev**, katere rezultat so **sistemske specifikacije**

Sistemske specifikacije so namenjene:

- razvijalcem (potrebujejo za načrtovanje strukture),
- preverjevalcem (tvorba testnih primerov) programske opreme in
- naročniku.

Struktura dokumenta:

1. Identifikacija dokumenta, povezava z drugimi dokumenti.
2. Povzetek (Povzetek dokumenta in pregledni opis bodočega produkta - dobimo ga iz Projektnega plana)
3. Zahtevane karakteristike (zanesljivost, prenosljivost, varnost itd. - v planu so nakazane, sedaj jih lahko že bolje preciziramo)
4. Omejitve in druge zahteve (zagon programa, zahteve glede aparaturne opreme, napotki za načrtovanje ipd.)
5. Opis sistema (Uporabimo diagram toka podatkov, diagram prehajanja stanj, ...
6. Opis podatkovnih tokov in terminatorjev (Če smo uporabili diagram toka podatkov)
7. Podroben opis in indeksiranje funkcij, ki jih mora sistem izvajati.
8. Zunanji videz (Izgled najpomembnejših izpisov na ekran oziroma na papir. **Izberemo konkreten zgled, ki ga kasneje uporabimo v Uporabniškem priročniku.**)
9. Opis funkcij, ki bodo najprej implementirane v obliki prototipa
10. Prevzemni kriteriji (Kakšni pogoji morajo biti izpolnjeni, da bo programski produkt zrel za predajo naročniku.)
11. Pojmovnik

4.4. Testni primeri

S pomočjo projektnega plana, kjer smo izbrali testirne strategije, naročnikovih zahtev in sistemskih specifikacij tvorimo toliko testnih primerov, da bo zadoščeno terminalni kriterijski funkciji in strategijam. V tem dokumentu so opisane izbrane strategije (v kolikor tega nismo storili v dokumentu Projektni plan), testni primeri in postopek testiranja. Testni primeri morajo biti ustrezno dokumentirani in urejeni, tako da imamo pregled nad njihovo količino, namenom in strategijo, na podlagi katere so nastali. Hkrati tudi pripravimo formular za naša opažanja (seznam nepravilnosti) pri testiranju. Na podlagi testnega primera mora biti nepravilnost **ponovljiva**, kar je nujen pogoj za njeno odpravo. Če so testni primeri dobro opisani, lahko na podlagi tega dokumenta testiranje izvede tretja oseba, ki zna uporabljati testirni objekt.

Struktura dokumenta:

1. Identifikacija dokumenta, povezava z drugimi dokumenti.
2. Povzetek - kratek opis projekta
3. Seznam funkcij (zahtev), ki jih nameravamo preverjati in njihova pokritost v obliki testne matrike
4. Identifikacija objektov in strategij (identificiramo objekte, na katere se testni vzorci nanašajo in imena uporabljenih strategij
5. Opis testnih primerov (opišemo podatke, ki so skupni vsem testnim primerom, npr. kje se nahajajo)
6. Opis testirnega postopka (npr. opis potrebne opreme, opis postopka posredovanja testnih primerov, nastavitev raznih sistemskih spremenljivk itd.)
7. Priloga s
   7.1. Opisi uporabljenih strategij
   7.2. Opisi konkretnih testnih primerov - Vsak testni primer mora vsebovati: **opis začetnega stanja, opis konfiguracije oz. okolja, namen, ime strategije, opis protokola oz. vhodnih podatkov, opis pričakovanega odziva in odpovedno kriterijsko funkcijo**

4.5. Poročilo o preverjanju

Po izvedenem testiranju zapišemo svoje ugotovitve.
Struktura dokumenta:

1. Identifikacija dokumenta, povezava z drugimi dokumenti.
2. Ponovimo točke iz poglavja o preverjanju in jih komentiramo v smislu poteka izvedbe.
3. V poročilu morajo biti vsi podatki (opis aparaturne opreme, operacijskega sistema, testirnega okolja ipd.), ki naj zagotovijo ponovljivost.
4. Opis izvedenih aktivnosti in ugotovitev (Podamo kvantitativne podatke o vloženem naporu, kompleksnosti programske opreme, doseženi pokritosti in učinkovitosti testiranja.)
5. Klasificiran seznam ugotovljenih, odpravljenih ter preostalih nepravilnosti oziroma neustreznosti.
6. Priloga: podroben seznam neustreznosti oziroma nepravilnosti

4.6. Načrtovalska dokumentacija

Ta dokument je namenjen vzdrževalcem programske opreme. Struktura dokumenta:

1. Identifikacija dokumenta, povezava z drugimi dokumenti.
2. Povzetek iz specifikacij
    2.1. Kontekstni nivo
    2.2. Opis datotek, ki jih uporablja uporabnik
    2.3. Zagon programa
    2.4. Datoteke, ki jih potrebuje vzdrževalec
3. Strukturni diagram
4. Seznam modulov in podatkovnih tokov
5. Opisi posameznih modulov

    4.1. Ime, verzija, vrsta, avtor
    4.2. Kliče module
    4.3. Je klican
    4.4. Definicija
    4.5. Sklopljenost z globalnimi podatki oziroma parametri
    4.6. Kompleksnost
    4.7. Opis nadomestnega modula

6. Najpomembnejši parametri in opisi podatkovnih struktur.
7. Natančna identifikacija (verzije) uporabljenih orodij in knjižnic.
8. Postopek, ki je potreben, da ustvarimo izvršilno kodo. (identifikacija razvojnega okolja, nastavitvene datoteke v razvojnem okolju, vrsta prevajalnika itd).
9. Pojmovnik

4.7. Uporabniški priročnik

Namenjen je uporabniku. V njem so vsa potrebna navodila za uporabo in instalacijo. Struktura dokumenta:

1. Referenca na verzijo programske opreme.
2. Zahteve glede strojne opreme in programskega okolja
3. Instalacija.
4. Navodila za uporabo (sintaksa).
5. Zgled za uporabo (uporabimo itsi zgled kot vsmo ga v specifikacijah)
6. Hibe programa
7. Pojmovnik

4.8. Stil kodiranja

4.8.1. Imenovanje

Konstante : velike črke. npr.: MAX, PI, MAX_VOLUMEN

Imena funkcij: Začnejo se z veliko črko. Vsebujejo glagol.
npr. `Beri_podatke`

Spremenljivke: samo male črke. Npr.: `vrstica, polje, tekoca_vrstica`

Ločevanje imen samo glede velikosti črk, ni priporočljivo,
npr.: `vrstica, Vrstica`

Globalne spremenljivke naj imajo pred imenom g_ : npr. `g_korak`

V izrazih uporabi konstanto namesto konkretnih vrednosti.
Npr. `if (korak>100)` raje `if (korak>MAX_KORAK)`

4.8.2. Zamikanje in označevanje blokov

Gnezdenje naj bo označeno z zamikanjem zavitih oklepajev. Dolgi bloki naj vsebujejo ime na začetku in na koncu. Npr.:

```
for (sy = sytable; sy != NULL; sy = sy->sy_link)
{
     if (sy->sy_flag == DEFINED)
     {
        ...
     } /* if defined */
     else
     {
        ...
     } /* if undefined */
} /* for all symbols */
```

4.8.3. Komentiranje

4.8.3.1. Glavni program mora imeti v glavi opis (namen, odvisnost od datotek, avtor, njegov naslov, datum nastanka, datumi sprememb, oznaka za zadnjo verzijo.

4.8.3.2. V glavi vsake pomembne rutine mora biti pojasnjeno, katere spremenljivke so vhodne, katere izhodne in katere krmilne. (Ne pozabi na globalno definirane spremenljivke.)

4.8.3.3. Minimalna dokumentiranost izvorne kode je **2 besedi/programsko vrstico (**merimo s številom komentarskih besed na programsko vrstico. (Vse vrstice, ki niso prazne ali pa komentar, so programske vrstice.)

4.8.3.4. Pomembne spremenljivke in konstante naj bodo pri deklaraciji komentirane.

4.8.4. Struktura izvorne kode

Aplikacijo sestavljajo programi in moduli, konfiguracijske datoteke, podatkovne baze in zagonske datoteke.

Struktura glavnega programa

```
glava programa
#include
#define
deklaracije spremenljivk
funkcija 1
funkcija 2
main
        deklaracije lokalnih spremenljivk
:
```

Struktura modula

```
Glava modula
funkcija A
funkcija B
:
```

Struktura posamezne funkcije

```
deklaracija funkcije
Glava funkcije
deklaracije lokalnih spremenljivk
```

4.8.5. Razno

4.8.5.1. Izogibaj se uporabi globalnih spremenljivk.

4.9. Opis in organizacija datotek, ki so na disketi

Vse datoteke, ki so se pri razvoju uporabljale oziroma direktoriji, ki so na priloženih disketah, morajo biti na kratko opisani.

Dodatek

Nekatere uporabne tabele in formularji

POROČILO O NEPRAVILNOSTI

Forma: ONEU V3.1s

1)PROJEKT:_____ 2)NEPRAVILNOST ŠTEV.:_____

3)TESTNI OBJEKT: _____ 4)VERZIJA:_____

5) KONFIGURACIJA:_____

6)VRSTA NEPRAVILNOSTI *(1-3)*:
1 Nepravilno implementirano
2 Ni implementirano
3 Ni bilo zahtevano

7)RESNOST NEPRAVILNOSTI *(1-3)*:
1 - Nepomembna
2 - Resna
3 - Zelo pomembna

8)LOKACIJA NEPRAVILNOSTI: _____

9)KRATEK OPIS NEPRAVILNOSTI: _____

_____

_____

_____

_____

_____

_____

10)JE PONOVLJIVOST ZAGOTOVLJENA *(da/ne)*?
11)PREDVIDENA POGOSTOST POJAVLJANJA: *1 zelo pogosto, 2 občasno, 3 redko*
12)TESTNI PRIMER/POSTOPEK : _____

_____

_____

13) PRILOGE: _____

14)POROČEVALEC:_____ 15)DATUM: _____

Izpolni vodja

16)ODGOVORNA OSEBA_____

17)PRIORITETA:*1 Takoj odpraviti 2 Čim bo mogoče*

18) KONČNI STATUS NEPRAVILNOSTI _____ 19) Podpis vodje_____
(glej polje 20)

Izpolni razvojna skupina

20) NEPRAVILNOST JE (1-9):

1 Odpravljena
2 Prestavljena (hiba)
3 Neponovljiva
4 Zavrnjena
5 Preklicana
6 Ignorirana
7 Ni več aktualna
8 Duplikat
9 Potrebne so dodatne inf.

21)OPOMBA _____

_____

22) NEPRAVILNOST ODPRAVIL _____ 23)DATUM _____

24)POPRAVILO PREVERIL _____ 25)DATUM _____

Testni primer:		**TESTNI PRIMER**	**Verzija:**		Projekt:
Garnitura:			Avtor:		

Začetno stanje		Strategija:	
Namen		Opomba:	
Konfiguracija sistema:		Odvisnost od drugih testov:	

korak	akcija / ime vhodne spremenljivke	vrednost	pričakovana reakcija/vrednost	opomba/odpovedna krit. funk.	opomba za testno poročilo
1					
2					
3					
4					
5					
6					
7					