XXXII JUGOSLOVENSKA KONFERENCIJA ETAN-a, S A R A J E V O, 6--10. JUNA 1988

MILAN ZORIĆ
MARIJAN KUNŠTIĆ
ŽELJKO GALOVIĆ
Zavod za telekomunikacije
ELEKTROTEHNIČKI FAKULTET
ZAGREB, Unska 3

AKTIVNA PODRŠKA KORISNICIMA SDL-a

ACTIVE SUPPORT OF SDL USERS

SADRŽAJ - U radu je prikazan dio analiza na osnovu kojih se planira uvođenje SDL - a u proces proizvodnje programske podrške komutacijskih sistema. Primjerima selekcije elemenata jezika, odnosno definiranja internih standarda, ilustrirana je potreba da upotreba jezika bude podržana metodologijom korištenja i programskim pomagalima prilagođenim korisniku.

ABSTRACT - The way SDL is being introduced in the switching software production process is described. Throgh exaples of SDL elements selection and definition of internal standards, the need for a language to be supported with a methodology of it's use and software tools tailored for a particular user is ilustrated.

1. UVOD

Ključno pitanje u radu na projektiranju složenih funkcija komutacijskih sistema je uspješno transformiranje skupa apstraktnih ideja u njihovu implementaciju. U središtu tog problema leži izbor odgovarajućeg jezika za opisivanje predlaganih rješenja, definiranje metodologije korištenja jezika, te izgradnja odgovarajućeg skupa razvojnih pomagala.

Za specifikaciju i razvoj telekomunikacijkih sistema, sve je šire prihvaćen specifikacijki jezik SDL [1]. Osnovna karakteristika SDL-a je da omogućava da se funkcije opisivanog sistema mogu definirati na elegantan način, te da semantika jezika omogućava interpretaciju i analizu opisa. Činjenica da se radi o jednom od malobrojnih

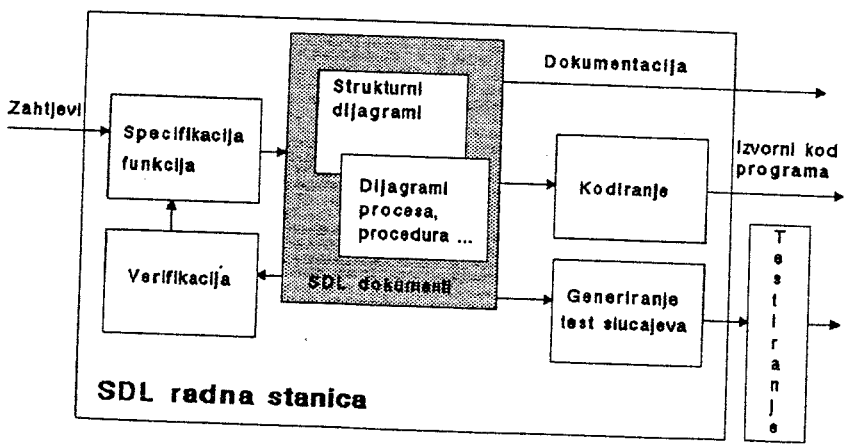
međunarodno priznatih standarda u ovom području dalji je razlog za njegovu širu primjenu.

Međutim, jedna je stvar imati dobro sredstvo za obavljanje posla, a druga je stvar pouzdana i efikasna upotreba tog sredstva. Dobro je poznato da i najbolje notacije mogu vrlo lako biti zloupotrebljene. To je posebno prisutno u ponekad apstraktnom svijetu specifikacije sistema, gdje se teži velikim i složenim jezicima, kao što je i sam SDL, kako bi jezik posjedovao dovoljnu moć izražavanja. Iz navedenog proizlazi da u upotrebi SDL-a u svakoj konkretnoj sredini postoje praktični problemi koji moraju biti riješeni kako bi rezultati bili u skladu s očekivanjima.

Želja nam je da u ovom radu ukažemo na međusobnu povezanost jezika, metodologije korištenja SDL - a, funkcijske strukture programskih pomagala i sredine u kojoj se sve to treba primijeniti.

2. PROBLEMI UVOĐENJA SDL-a

Prilikom uvođenja SDL-a u razvojne poslove opći je cilj formiranje zaokruženog procesa proizvodnje i održavanja programske podrške. Globalni funkcijski elementi takvog procesa prikazani su slikom 1. Ovakvo viđenje tog procesa proizlazi iz današnjeg stanja kao i sagledivih mogućnosti tehnologije proizvodnje programske podrške s jedne strane, odnosno mogućnosti koje pruža SDL jezik kao osnovni oslonac u tom procesu.



Slika 1. Proces razvoja programske podrške komutacijskih sistema

U tako koncipiranom procesu, osnovna dokumentacija oko koje se sve vrti je SDL dokumentacija. Programska pomagala [2, 3, 4] omogućavaju efikasno kreiranje i ažuriranje SDL dokumenata. Takvi dokumenti predstavljaju ulaz u procese verifikacije, generiranja test slučajeva, preslikavanje u implementacijski kod i testiranje sistema. Održavanje sistema svodi se na ažuriranje SDL dokumenata, te ponavljanje ostalih faza razvoja nad promijenjenim SDL dokumentima [5].

Na ovom mjestu želimo istaći da, iako se SDL dokumenti kreiraju i kontroliraju uz pomoć programskih pomagala, oni prvenstveno predstavljaju sredstvo za komunikaciju ideja među ljudima. Stoga smatramo da realizacija elemenata gore naznačenog globalnog procesa proizvodnje nužno nije jednoznačna, već mora biti prilagođena konkretnoj sredini.

Prilikom uvođenja SDL-a u neku sredinu potrebno je analizirati slijedeće elemente:

a) dosadašnja iskustva u razvoju komunikacijkih sistema, posebno obzirom na način specificiranja funkcija, te korištenje SDL-a odnosno njemu sličnih tehnika,

b) način kreiranja, čuvanja i ažuriranja dokumentacije,

c) karakteristike implementacijkih jezika koji se koriste odnosno planiraju koristiti,

d) korištena odnosno planirana pomagala na nivou implementacijkih jezika,

e) pomagala i metode testiranja, počem od načina generiranja testova do njihovog izvršavanja,

f) koncepciju održavanja programske podrške u eksploataciji.

U ovoj analizi potrebno je za određenu sredinu sagledati na koji način osigurati četiri ključna cilja:

a) razumljivost specifikacija,

b) točnost specifikacija,

c) lagano preslikavanje specifikacija u implementaciju,

d) efikasno testiranje implementacije.

Polazeći od ovih elemenata potrebno je:

a) izvršiti selekciju elemenata SDL-a koji će se koristiti u okviru jednog razvojnog projekta,

b) za korištene elemente definirati vlastite standarde korištenja,

c) definirati kontrole koje su, u okviru usvojenog pristupa potrebne i moguće,

d) prilagoditi funkcijsku strukturu pomagala,

e) razraditi plan školovanja kadrova.

Neke od ovih elemenata ilustrirati ćemo u narednim poglavljima.

3. SELEKCIJA ELEMENATA SDL - a

SDL jezik je toliko širok, da među svim organizacijama koje ga koriste, izvjesno ne postoji niti jedna koja koristi sve njegove elemente. U narednim primjerima ukazati ćemo na neke elemente čije korištenje smatramo da treba dodatno evaluirati, kao i probleme koji su u dosadašnjim analizama uočeni.

Problem selekcije elemenata možemo ilustrirati slijedećim primjerima:

a) procedure i servisi

Ukoliko se kod dekompozicije procesa koriste i procedure i servisi, mogu se javiti teškoće u interpretaciji SDL opisa. Zbog toga je u zadnjim revizijama Z100 preporuka uvedeno ograničenje da uz korištenje servisa procedure unutar istog procesa ne smiju posjedovati stanja. Ovakve greške mogu biti otkrivene prilikom kontrole semantike, ali smatramo problem treba biti predupređen odgovarajućim vlastitim ograničenjima ili uputama.

b) SAVE

Na specifikacijskom nivou ovaj se element može dobro objasniti, ali njegova implementacija znatno komplicira manipuliranje s ulaznim događajima procesa. Ovisno o planiranom načinu implementiranja sistema, ovaj konstrukt možda treba izbjegavati odnosno zabraniti. Ujedno, postoji nekoliko drugih elemenata (npr. ENABLE) koji implicite sadrže SAVE, tako da problem treba cjelovito sagledati.

c) EXPORT/IMPORT

Korištenje ovih mehanizama može znatno smanjiti opis danog problema, što je nesumljivo vrlo poželjna osobina. S druge strane, semantika ovih elemenata nije jednostavna, te korisnik lako može previdjeti skrivene posljedice njihovog korištenja. Greške koje se na ovaj način mogu unijeti u specifikaciju moguće je otkriti prilikom dinamičke analize semantike. No treba upozoriti da je analiza dinamičke semantike najsloženiji problem u kontroli ispravnosti SDL dokumenata, te da će ove komponente pomagala biti u pravilu zadnje implementirane. Adekvatne upute i trening mogu umanjiti problem.

d) Neformalni tekst

Ovo je ključno pitanje koje razgraničava dva bitno različita načina korištenja SDL - a. Ukoliko se koristi neformalni tekst, specifikacija nije dovoljno formalna, moguće je kontrolirati samo rudimentarne elemente semantike, nije moguća

automatizirana interpretacija i simulacija itd. Iako čak i ovakvo korištenje SDL - a predstavlja bitan napredak u odnosu na verbalne opise funkcija, zalažemo se da se koristi formalni SDL. To ne isključuje upotrebu manje formalnog SDL - a u procesu deriviranja detaljnih formalnih specifikacija. Obzirom da su u najvećem broju slučajeva dosadašnja iskustva vezana uz neformalne specifikacije, posebnu ulogu u prelasku na formalne opise treba odigrati školovanje kadrova.

d) Apstraktni tipovi podataka

Apstraktni tipovi podataka mogu predstavljati vrlo moćno sredstvo za neproceduralno specificiranje željenih funkcija. No treba reći da su iskustva s njihovim korištenjem u svjetskim razmjerima vrlo ograničena, te bi bilo nužno steći vlastita iskustva na određenim pilotskim projektima. Smatramo da se u prvom trenutku treba ograničiti na korištenje unaprijed definiranih tipova podataka, bez mogućnosti korištenja elemenata za kreiranje novih tipova. U svakom slučaju potrebno je definirati vlastiti skup pravila koji može biti sličan pravilima navedenim u [8 pp 235-245].

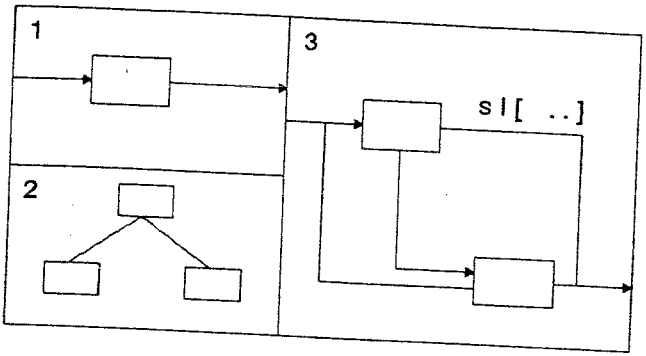
4. INTERNI STANDARDI KORIŠTENJA SDL - a

Osnovni preduvjet za uspješnu komunikaciju među članovima razvojnog tima je dobro poznavanje semantike korištenih elemenata jezika. To je moguće postići odgovarajućom kombinacijom CCITT materijala, internih uputa i školovanja kadrova. Pored toga, moguće je razumljivost povećati definiranjem vlastitih standarda dokumentiranja, odnosno uvođenjem dodatnih ograničenja u odnosu na CCITT standard.

4.1 Način formiranja dokumentacije

Na nivou opisa strukture sistema važno i načinom dokumentiranja istaknuti ono što je bitno za svaku komponentu strukture. Jedan od mogućih pristupa pokazan je na slici 2, pri če mu redni brojevi 1, 2 i 3 imaju slijedeće značenje:

1. dijagram sučelja - pokazuje mjesto komponente u nadređenoj strukturi
2. stablo podređenih komponenata - pokazuje sastavne dijelove
3. interakcijski dijagram - pokazuje način komuniciranja podređenih komponenata, tj. komunikacijske puteve (kanale odnosno signalne rute) i liste signala.



Slika 2. Način dokumentiranja na nivou strukturnih opisa

4.2 Ograničenja u korištenju skokova (JOIN)

Ima primjera da je u pojedinim projektima zabranjeno korišenje skokova van konteksta istog stanja [8 pp 3-9], iako to CCITT preporuke dozvoljavaju. Ovakvo ograničenje može nešto povećati obim specifikacije neke funkcije, ali u principu vodi na bolju strukturiranost i lakšu implementaciju, te stoga smatramo da je o ovakvom ograničenju vrijedno razmisliti.

S druge strane susrećemo proširenja SDL jezika novim sintaksnim elementima, što smatramo da pod svaku cijenu treba izbjegavati. Još je opasnije ako se interno promijeni semantika pojedinih elemenata.

4.3 Preporučena ograničenja na nivou strukturnih dijagrama

Kako bi opisi strukture bili primjereni ljudskim mogućnostima povezivanja činjenica moguće je za dekompoziciju sistema definirati slijedeće interne preporuke:

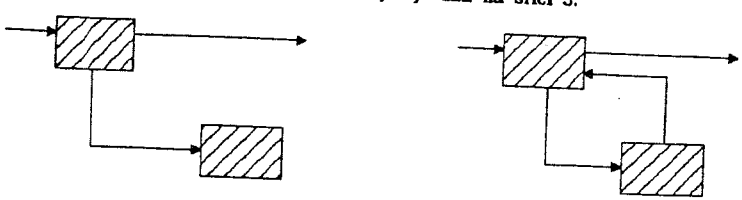
a) ne više od šest kanala na sučelju jedne komponente,

b) ne više od tri grane na stablu komponenata,

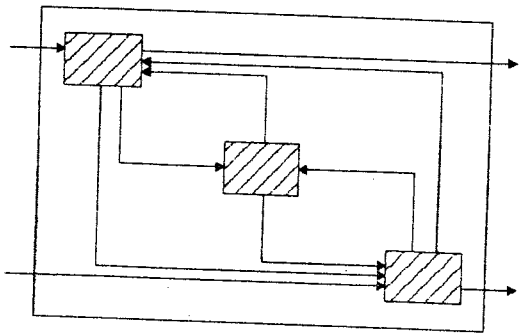
c) jedan dozvoljeni divergentan/konvergentan kanal na dvokomponentnom dijagramu,

d) na dijagramu sa tri komponente izbjegavati divergentne/ konvergentne kanale,

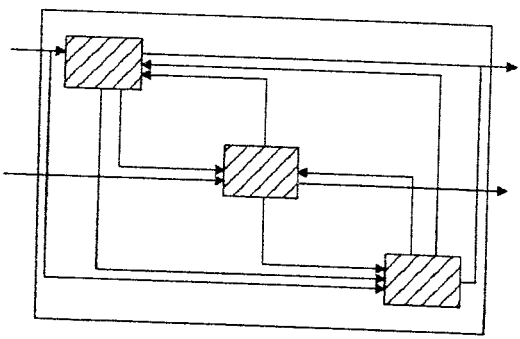
e) na interakcijskom dijagramu sa više od dvije komponente treba izbjegavati upotrebu strukturno redundantnih komponenata. Primjer je dan na slici 3.



Slika 3. Strukturno redundantne komponente



Slika 4. Trokomponentni dijagram koji nije u skladu s gornjim ograničenjima



Slika 5. Trokomponentni dijagram u skladu s internim preporukama

Ova se pravila smiju prekršiti ako za to postoje dovoljno jaki razlozi, ali to onda treba na odgovarajući način dokumentirati.

5. ANALIZA SDL SPECIFIKACIJA

U poglavlju 3. zalažemo se za to da korištenje SDL - a bude formalno. Upravo ta činjenica nameće potrebu da u sklopu SDL radne stanice predvidimo komponentu za kontrolu semantičke korektnosti unijetih specifikacija. Dok inicijalna kreacija specifikacija može biti oslonjena na interno definirana pravila, kompleksnost funkcija modernih komutacijskih sistema nužno zahtijeva pomoć računala u analizi specifikacija. Tako se nakon utvrđivanja sintaksne ispravnosti kontroliraju elementi semantike, među kojima ističemo slijedeće primjere:

- kontrola tipova varijabli i izraza - slično kao kod prevodilaca visokih programskih jezika
- kontrola labela - provjerava se da sve JOIN naredbe imaju definiranu labelu na koju se prenosi kontrola,
- kontrola signala - provjera da svi signali imaju dozvoljena odredišta
- kotrola parova kanal/signal - provjera da svaki signal ima definiran kanal
- kontrola vidljivosti - provjera da li su eksterne varijable dohvatljive

Sve ove provjere moraju biti provedene, jer je čovjek sklon da na nekom mjestu vidi ono što želi vidjeti, a ne ono što se tamo stvarno nalazi. Ove provjere slijede definiciju semantike SDL - a. Neki aspekti realizacije ovih funkcija pomagala opisani su u [7]. Moguće je proširiti skup provjera tako da budu obuhvaćene i provjere koje proizlaze iz internih standarda.

6. ZAKLJUČAK

Da bi specifikacijski standardi bili efikasno provedeni nisu dovoljne Z100 preporuke, već je potreban osmišljen pristup u smislu uputa za korištenje jezika i praktična podrška odgovarajućih programskih pomagala. U ovom su radu opisane naše trenutne aktivnosti u tom pravcu.

LITERATURA:

1. XXX, "Specification and Description Language SDL - Draft Recommendation Z100", SDL Newsletter No 10, 1986
2. Zorić, M., Ž. Galović, "Elementi programskog pomagala za rad sa SDL-om", Zbornik radova 28. Etan u pomorstvu, Zadar, 1986
3. Zorić, M., "An Approach to SDL Tool Development", in Saracco, R., P.A.J. Tilanus, Eds, "SDL' 87 State of the Art and Future Trends, Proceedings of the Third SDL Forum, The Hague, The Netherlands, April 1987, North-Holland, 1987
4. Galović, Ž., M. Zorić, "Realizacija SDL grafičkog editora", Zbornik radova ETAN-a, Bled, 1987
5. Zorić, M., "SDL radna stanica i razvoj programske podrške komutacijskih sistema", Zbornik radova ETAN-a, Bled, 1987
6. Zorić, M., "Sintaksna analiza SDL/PR uz pomoć LEX-a i YACC-a", Zbornik radova 6. MIPRO-TE, Rijeka, 1987
7. Zorić, M., "Strukture podataka i njihove transformacije u SDL radnoj stanici", Zbornik radova 7. MIPRO-TE, Rijeka, 1988
8. Saracco, R., P.A.J. Tilanus, Eds, "SDL' 87 State of the Art and Future Trends, Proceedings of the Third SDL Forum, The Hague, The Netherlands, April 1987, North-Holland, 1987