OBJEKTNA ARHITEKTURA NA OSNOVI SEMANTIČNE ZGRADBE INFORMACIJE

V. ŽUMER,
P. KOKOL,
L. PIPAN

UDK: 681.3.07

VISOKA TEHNIŠKA ŠOLA, MARIBOR, JUGOSLAVIJA
FAKULTETA ZA ELEKTROTEHNIKO, LJUBLJANA, JUGOSLAVIJA

Članek podaja semantično zgradbo informacije, kjer imajo objekti omejitev, lastnost, atribut in predstavitev. Prikazana je arhitektura za nekatere značilne objekte. Ta koncept je zelo primeren za zanesljivo neposredno izvajanje.

OBJECT ARCHITECTURE ON THE SEMANTIC STRUCTURE OF INFORMATION. The paper gives the semantic structure of information where objects have scope, property, attribut and representation. Storage architecture for some interesting objects is described. This concept for relisble direct execution of HLL programs is appropriate.

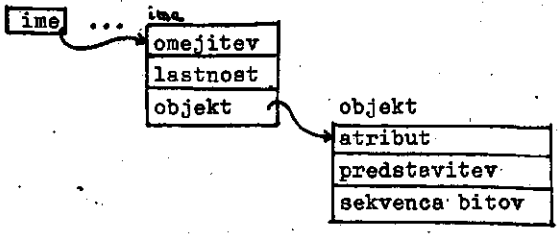
1. UVOD

Kljub izrednemu razvoju elektronskih vezij po eni strani in razvoju programskih jezikov in tehnik programiranj po drugi strani, večina računalnikov deluje na principu klasičnega von Neumannovega koncepta. Posledica tega je, da imamo danes še zmeraj velik semantični razkorak med uporabnikovim okoljem in materialno opremo. Do sedaj znane izboljšave von Neumannovega koncepta v zgradbi informacije so značke (1), ki vnašajo atribut podatka in deskriptorji (2), s pomočjo katerih se dosegajo podatkovne strukture. Prav tako je znan koncept capability adresiranja (3), s katerim se doseže pravilno dodeljevanje informacije. Še višjo abstrakcijo podatkov zasledimo v zadnjem času s tako imenovano objektno orientirano arhitekturo, kjer imamo namesto linearnega pomnilnika množico objektov.

Vse navedene sheme pa uporabljajo še vedno fiksno dolžino informacije kot npr. zloge ali besede. Slabost tega je, da ni mogoče opravljati operacij med različno dolgimi podatki. Prav tako ni mogoče dinamično spreminjati narave objektov. Izboljšanje navedenih slabosti dosežemo s semantično zgradbo informacije, ki se zrcali že v arhitekturi računalnika.

2. SEMANTIČNA ZGRADBA INFORMACIJE

Semantično zgradbo informacije predstavimo kot množico odnosov med dosegom in vrednostjo (4). Odnose razdelimo v štiri hierarhične kategorije: omejitev, lastnost, atribut in predstavitev (sl. 1). S tem je običajno definirani tip razdeljen na lastnost in atribut.



Slika 1

Ime vsebuje omejitev, lastnost ter kazalec na objekt, sam objekt pa vsebuje atribut, predstavitev in sekvenco bitov. Omejitev definira, na kakšen način in komu je dosegljiva informacija preko imena. S tem dosežemo pravilno dodeljevanje informacije in zaščito. Predstavitev daje možnost spremenljive dolžine in strukture informacije. Iz sekvence bitov dobimo glede na atribut in predstavitev vrednost objekta ali kazalec na drug objekt.

Imamo dve osnovni operaciji: asocijacijo in evaluacijo. Asocijacija poveže ime in objekt ter se uporabi tedaj, kadar spremenljivka dobi neko vrednost. Če je bilo ime povezano z objektom, asocijacija sprosti stari objekt in poveže ime z novim objektom. Pri asocijaciji je treba najprej pregledati pravico dosega glede na omejitev v imenu in nato še primernost asocijacije glede na lastnost v imenu. Z evaluacijo se zgradi nov objekt in to pri kakršnih-koli opracijah s starimi objekti. Glede na vrednost omejitve v imenu se ugotavlja primernost evaluacije in možnost operacije. Novi objekt ima samo atribut in predstavitev, ne more pa imeti lastnosti niti omejitve tako dolgo, dokler se ne poveže z imenom.

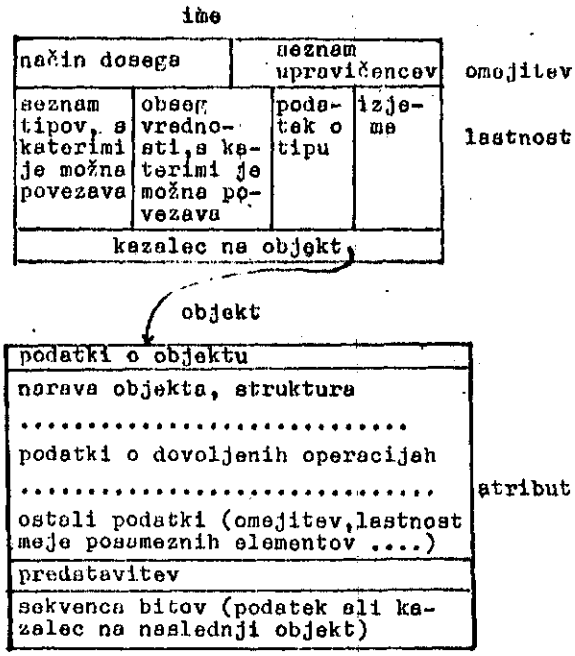
Omejitev vsebuje dva dela: način dosega in seznam upravičencev, ki lahko dosežejo to ime. Zlasti pri dodeljenih in porazdeljenih informacijah omejitev omogoča večjo zanesljivost izvajanja.

Lastnost definira odnos med imenom in množico objektov, ki se lahko povežejo s tem imenom. Lastnost torej določa, kateri atribut v objektu in kakšne vrednosti se lahko povežejo z imenom. V lastnosti se lahko podajo razne izjeme, ki so morda potrebne zaradi implementacije arhitekture.

Atribut definira odnos med objektom in med množico dovoljenih operacij s tem objektom. Atribut določa naravo objekta kot je npr. procedura, celo ali realno število, prav tako pa določa tudi strukturo objekta kot npr. polje, zapis, itd.

Predstavitev določa odnos med objektom in nje-

govo fizično predstavitvijo. Predstavitev določa osnovo številskega sistema, dolžino števila, način zapisa kodiranja, način kako je predstavljeno polje, dolžino dinamičnih struktur itd.



Slika 2

Na sliki 2 prikazujemo podrobnejšo zgradbo imena in objekta.

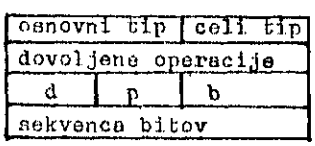
Podatki o objektu vsebujejo dodatno informacijo o objektu, ki je potrebna pri upravljanju s pomnilnikom (dolžina objekta, število segmentov objekta, aktivnost objekta, ...).

3. ARHITEKTURA OBJEKTOV

Predpostavljamo dovolj velik pomnilnik, v katerem je seklјalna tabela in množica naključno razporejenih imen in objektov. Takšen koncept se izkaže kot zelo uporaben pri neposrednem izvajanju visokega programskega jezika, saj moramo v tem primeru preglјedovati primernost evaluacije oz. asocijacije tik pred izvajanjem.

V našem primeru so objekti lahko različne podatkovne strukture, seklјalne tabele, okviri, slike procedur in krmilnih konstruktov kot so npr. if, while, when.

Objekt za celi tip (integer)



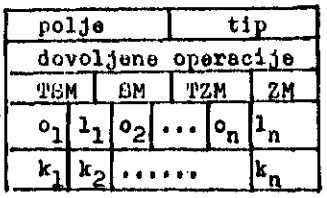
d - dolžina sekvence bitov
b - baza številskemu sistemu
c - predznak števila

Objekt za sestavljeni tip polja (array)

Zaradi enostavnosti implementacije je polje deklarirano enodimenzionalno, več dimenzij dosežemo z rekurzijo.

Splošni stavek za deklaracijo polja je:

ime is array SM:TSM ... ZM:TZM of tip



SM - spodnja meja indeksa polja
TSM - tip SM
ZM - zgornja meja indeksa polja
TZM - tip ZM
$o_1, o_2 \dots o_n$ - omejitve posameznih elementov polja
$l_1, l_2 \dots l_n$ - lastnosti posameznih elementov polja
$k_1, k_2 \dots k_n$ - kazalci na naslednje elemente tega polja, ki so lahko objekti osnovnih tipov ali pa objekti sestavljenih tipov

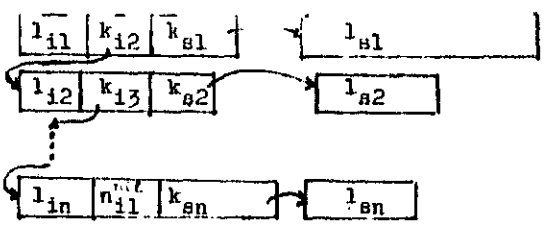
Objekt krmilnih stavkov

Podobno kot za podatkovne strukture, lahko tudi za krmilne stavke upoštevamo semantiko že v arhitekturi. Objekt krmilnega stavka ne potrebuje imena, ima samo vstop v seklјalni tabeli.

Splošna oblika krmilnega stavka je:

struct	logični izraz 1 do lista stavkov 1
else	logični izraz 2 do lista stavkov 2
⋮	
else	logični izraz n do lista stavkov n
end	

Objekt krmilnega stavka vsebuje več objektov:



l_i - logični izraz
l_s - lista stavkov
k_i - kazalec na logični izraz
k_s - kazalec na listo stavkov

4. ZAKLJUČEK

Z navedeno arhitekturo je mogoče zelo zanesljivo izvajanje programov, s čimer se zmanjša cena vzdrževanja programov. Prav tako se poveča učinkovitost izvajanja programov in zaščite informacije. Če zahtevamo izvajanje programov z veliko zanesljivostjo, potem je v navedenem konceptu izvajanje programov mnogo hitrejše kot pa v običajnih računalnikih. To je razumljivo, ker je koncept preverjanje vnesen že v arhitekturo, kjer se dinamično preverjajo omejitev, lastnost, atribut in predstavitev. V primeru, da želimo enako zanesljivost doseči z običajnimi računalniki, opravljajo to delo programi v okviru operacijskega sistema.

Današnja VLSI tehnologija nam ponuja pomnilnike z velikimi kapacitetami na osnovi asociativnega dosega in z dodatno logiko. Ceneni mikroprogramirani procesorji pa omogočajo izdelavo procesorjev za naše posebne namene. Razvoj tehnologije kaže, da bo v nekaj letih implementacija objektne arhitekture kljub kom-

pleksnosti tudi ekonomsko upravičena.

LITERATURA:

1. E.A. Feustel, On the Advantages of Tagged Architecture, IEEE Tr. on Computer, C-22(7), 1973

2. T.A. Welch, An Investigation of Descriptor Oriented Architecture, Proc. of the Third Annual Symposium on Computer Architecture (1976)

3. J.B. Dennis, E.C. Van Horn, Programming Semantisc for Multiprogrammed Computations, CACM 9(3), March 1966

4. M. Tokoro, T. Takinaka, On the semantic structure of information, 9-th Annual Symposium on Computer Architecture, Austin 1982

5. V. Žumer idr., Raziskava mikroprogramirane arhitekture za implementacijo visokega programskega jezika, raziskovalna naloga, VTŠ, Maribor 1982