PASKAL; YAPISAL BİR Dili GENEL BAKIŞ

Mehmet BODUR
OOTü Elektrik ve Elektronik Müh. Bölümü

1. Giriş

Genel amaçlı bir yüksek düzey prograaİMa dili olan Paskal, 1970'li yılların başında Prof. Niklaus Uirth tarafından düşünüldü. Prof Hirth'in Paskalı geliştir* amacı yazılışların belli bir düzen içinde gerçekleştirmesini, böylece yazılan progrularm yapısal yönden hatasız ve kolay anlaşılır oltasını sağlayacak yeni bir dil oluşturıaktı. Bu dilin kullanıının yaygınlaşabilmesi için o zaman kullanılan bilgisayarlara kolayca uygulanacak ve bellekte az yer kaplayacak kadar basit, fakat aynı zaunda hızlı ve veritli olmalıydı. Bu ataçla Prof. Hirth ve C. A. R. Hoare, Zurih'te Eidgenossische Technische Hochschule'de (EIH) bir CDC 6400 bilgisayarda Algol 60'tm yola çıkarak yeni bir dil için bir derleyici geliştirmeye başladılar. Bu çalışma 1970'te düşündükleri dili tüiüyle gerçekleştiren bir derleyicinin hazırlanıp, denenmesiyle başarıyla sonuçlandı. Dilin adı, Fransız Mtetatikçi ve düşünür Blaise Paskal'in (1623-1662) adına Paskal olarak konuldu. Bir yıl sonra Belfast Oueen's University'de bir başka grup, aynı dilin derleyicisini ICL 1900 tipi bir bilgisayar için hazırladı. Yeni dilin ilk fornal tanı» 1971'de, seıantik yapısının tanımlanması ise 1973'te yayınlandı.

1972-75 yılları arasında Paskal büyük bir gelişte dönemi geçirdi. 1972'de EIH den bir grup, Paskalın gereksiz sayılabilecek, ve bilgisayarın donamı özelliklerine bağlı yanlarını kısıtlayan, böylece her makinada uygulamasını kolaylaştıran bir düzeltme raporu hazırladı. Bu rapora uygun yeni derleyici, 1974 yılında taıanlandı. Paskal-P adlı yeni derleyicinin her çeşit bilgisayara kolayca uygulanabilmesi için ilgi çekici bir yöntH denendi. Paskal-P derleyicisi, P-code adında soyut, yiğit yönelisli (stack oriented) bir dilde yazıldı. Derleyici, aaac kodu (object code) olarak gene P-code üretiyordu. Bu yüzden, herhangi bir bilgisayarda Paskal derleyicisini kullanabilmek için o bilgisayarda yalnızca P-code yorunlayıcısı yazmak yeterli oluyordu. Bu yaklaşıl sayesinde Paskal-P bir anda DEC-Sistei 10, IBft-360, Univac-1106 ve benzeri çeşitli bilgisayarlara kolayca uygulandı.

Paskal dilinin oluşturulmasındaki temel nedenlerden biri de eğitimde kullanılmak üzere çeşitli dillerdeki ortak sistematik kavramları en basit özellikleriyle, ve en kolay anlaşılır biçimde kapsayan bir dilin gerektesiydi. Paskalın tasarlanmasında bu nokta da göz önüne alındığından Paskal sistematik programlama kavrulanrun öğretilmesinde önemli bir örnek oldu. Bu niteliği, dilin kısa zamanda yayınlarda algoritmaların açıklanmasında kullanılmasını yaygmlaştırdı. 1975'te ACM National Conference'de kurulan "Paskal Kullanıcıları 6rubu"nun üye sayısı 1978 başlarında *Si ilkede 2000'in üzerine* çıktı. Paskal, bugün tanınmış üniversitelerin çoğunda programlamaya giriş dili olarak öğretilmekte ve Paskal üzerine yazılmş pek çok ders kitabı bulunmaktadır. Dil aynı zinanda ticari olarak ta kendini kabul ettirmiştir. Bir çok yazılım fiması Paskalı ticari programlar yazmak için kullanmaktadır.

Metodik programlama özellikleriyle, ve her bilgisayara kolayca uygulanabilmeyle Paskal artık bilgisayar programcılığında FORTRAN ve BASIC gibi çok tanınmış dillerin yanında genel amaçlı, temel yüksek düzey dili olarak yerini almıştır.

2. Paskal Dilinin Yapısı ve özellikleri

Profesör Hirth, Paskal üzerine ilk çal IŞMI arından bu yana bu dilin geliştirilme amaçlarını şöyle belirtmiştin

a) Programlamayı sistematik temel kavramlara dayanarak öğretmekte kullanılabilecek, bu temel kavramları doğal yapılarıyla içeren bir dil olmalıdır.

b) fjünüaüzde her çeşit bilgisayara kolay, güvenilir, hızlı ve kullanışlı biçimde uygulanabileclidir.

Paskal dilinin tasarlanış amaçlarından kaynaklanan bu özelliklerini incelemeden önce, dilin gramer (syntax) yapıtının çok kısaltılmış bir özetini verelim.

2.1 Paskalın Gramer Yapısı:

i. Program öğeleri:

PROGRAM adl(INPUT, OUTPUT);
Bildirimler; (Declarations)
Bileşik Tümce; (Compound Statement)

Adlar (Name), bir alfabetik karakterle başlayan, içinde boşluk ve özel işaretler bulunmayan alfanümerik karakter dizgisinden oluşturulan tanıtıcı (identtfier) sözcüklerdir. Programın adı tanıtıldıktan sonra programda veri giriş çıkışı bulunduğu, giriş çıkış anlama gelen input, output sözcükleriyle belirtilir.

Bildirimler bölümünde, programda kullanılacak her tanıtıcı (identifier) sözcüğün belli bir sıra içinde bildirilmesi veya tanımlanması gerekir. Program son öğesi, program çalışınca yapılacak işlerin yazıldığı bir bileşik tümeldir.

ii. Bildirimler (Declarations):

a. Tümce Başlığı (Label) Bildirimi: Tümce başlığı olarak yalnız bildirilıiş sayılar kullanılabilir.

LABEL Ll, L2, .. Ln

b. Değismlrin (ConstınU) Bildirili: Değistezlr için kullanılacak tanıtıcıların karsılarına değerleri yazılır.

```
COHST adi =>dtğri;
      ad2 = değeri ;
        .
        .
```

c. Tip (Type) Bildirili: Yapısal veya türetilıiş yeni tip tanıtıcılarının tcıel tiplerden oluşturul» bicili bildirilir.

```
TYPE adi • tip tanıılaıasıl;
     ad2 «tip tanıılauı2;
        .
        .
```

d. D»9i«km (Vıriable) Bildirili! Progratda kullanılacak bütün genel (global) değişken adlarını, ve bunların veri tiplerini bildırıek gerekir.

```
VAR adi : tip adıl;
    ad2 ! tip adı2;
```

e. IıIM (Procedure) ve işlev (Function) Bildirin: tşletler; paratttreler ve genel değişkenler aracılığıyla dejer geçişine olanak sağlayan adlandırıl»; prograa bloklarıdır. Yapıları progrıaıa çok benzer. Ulu, adıyla çağırılınca isleedeki bileşik tüm islenir.

```
PROCEDURE işlendi (paraıetre bildirili);
Bildiriıler;          (yerel tanıtıcı adlar için)
Bileşik Tince;
```

İşlevler (Function), islıvadı ile değer taşıyabilen ve isleıe benzeym prograı bloklarıdır. Islevadlarınm tipi ve değeri vardır, işlevdeki bileşik tûtce içinde islevadına değer aktırıl•ası zorunludur.

```
FUNCTİON isltvadıtparaıctre bildirili): tip bildirili;
Bildirilir;          (Yrel tanıtıcı adlar için)
Bileşik Tüm;
```

iii. Tümlr (Statexnts):

Tümlr, prograıda yapılacak isİMİeri bildırıek için kullanılır. Bu bolünle tinceler S, Sİ, .. 9n gibi kısaltılacaktır.

a. Bileşik Tüeceler (Copound Statetents): Birden fazla tümnin (Sİ, S2, .. Sn) birleştirileni için kullanılır.

```
KBIN Sİ; S2; .. ; Sn; EKD
```

b. Değer Aktarıa (AMigmmt) TüKesi: Eşitliğin sağ tarafındaki aritMtik (Mntık veya başka tipten) deyıı, işlıvadı veya değişenin değeri eşitliğin solundaki aynı ya da uygun tipten değişkene aktarılır.

```
Değisktnadı :• deyıı
```

c. Denetil (Control) Tüecıleri: İstenen tümlerin belli kosullr altında islemesini ya da islmMccsini denetleyen yapılar olufturulıaımı sağlar,

1- IF «antıkdeyini THEN Sİ ELSE S2

Eğer MntıkdeyUi'nin sonucu doğruysa Sİ, doğru değilse S2 işlenir.

2- CASE secicideyim OF
değer la, dejerlb ... ı Sİ;
değer2a ... : S2;
.
değefN : Sn;
END

Secicideyii'in değerinin sırasıyla hangi tüacelere karşılık gelen değerlere uyduğu denetlenir. Uygun değerlerin tümleri islenir. Diğer tütceler işlemeden geçilir.

3- FOR skalardeğisken := ilkdeğer TO sondeğer DO S

Skalardeğisken'in (tasayı veya sıralı cüile eleıanları) ilkdeğer'den başlanarak bireri ik ilerletelerle sondeğer'e kadarla bütün değerleri için S tekrarlanır.

4- «HİLE untıkdeyiii DO S

Hantıkdeyiii"nin değeri doğru olduğu sürece S tüıcesi tekrarlanır.

5- REPEAT Sİ; S2; ... Sn UNTIL lantıkdeyiii

Sİ, .. Sn tümleri islendikten sonra nantıkdeyiii'nin değerine bakılır. Değeri doğru oluncaya kadar Sİ .. Sn tekrar islenir,

b- GOIO tüıcebaşhğı

Bu tünce, prograın akısını tümbaslığı ile belirtilen tümceye yöneltir.

d. Giriş Çıkış (Input-Output) isİMİeri: Veri giriş çıkışı standart işletirle gerçekleştirilir.

```
REAO veya READLN(değiSkenl, değiSken2, ... )
«İTE veya NRITELNtdeğrl, değr2, ... )
```

READ işieti değişkenler* değer okutıak, HRİTE işleıi ise deyi» veya karakter dizgileri gibi değerler listesini standart çıkış birisine yazdırıak için kullanılır. READLN ve HRITELN, giriş çıkış işleıi tanılandıktan sonra standart çıkış birisinde yeni satıra geçilenini sağlar.

2.2 Paskalın Sistemli Prograılau özellikleri

Paskal, sisteıli prograılaıaya yatkınlığını sağlayan çok sayıda denetil tümsi çeşidine sahiptir. Yapısal prograılaıanın teıeli, programları hiyerarşik olarak düzenlemiş tek girişli tek çıkışlı bloklardan veya döngülrden oluştunaya dayanır. Hiyrarşik düzenlemiş bloklar ya biri tüüyle diğrinin içinde, veya tüüyle birbirlerinden bağısız durutda olıahdır. Prograıların

bu stkilde düzenlemesi acık ve anlaşılabilir oltalarını önetli derecede etkiler. Paskalda BEBIN-E», İF-TOHLSE, NHILE-DO, REPEAT-UNTIL, FOR-DO ve CASE-END yapısal denetil tüıceleri, yapısal blokların acık ve anlaşılır biçude birbirinden ayrılıs olarak kullanılıasını sağlar. Paskalda GOIO tüKHi de vardır. Ancak GOIO tüKHi, diğer denetil yapılarıyla kurulan blokların daha anlaşılır oltası sebebiyle pek kullanıltaz. îç içe bloklardan oluşan bir progru örneği verelit. Bu prograı, bir kişinin haftanın hangi gününde kaçar sut çalıştığını okuyarak, cmartesi için her süte yarıı saatlik, pazar için ise bir saatlik fazla MUI ücreti üzrinden bir haftalık topla ücretini buluktadır.

```
PR06RM Ücret UNPUT.OUTPUT);

CONST Sutucrcti   « 200.0; {Ssaat bası is ücrctis

TYPE Gun«(PtKi,Sali,Cars,Pers,CuM,Ctesi, Pazar);

VAR  Isgunu          :6un;
     Toplaıucret, Fazlattsai t REAL}
     Issaati         INTEGER;

FUNCTİON Ücret(KatsayıiREAL) ı REAL;
   {Katsayı adlı grceksayı tipinde paratetresi vars
  BE6IN  {Ucret i»ltvinin bileşik tütcesis
    Ücreti- KatsayıtlssutüSutucrtti;
  EM};   {Ucret işlevinin sonu»

BEBİN      {Prograt Tûecesinin basış
 Topluucrrt :• 0;
 Ekucret    :• 0)
 HRITEUK'Pazartesinden Pazara kadar calisilan');
 HRITELNC sıatlri girin.');
 FOR Isgunu i» PtMi T0 Pazar DO
    BEBIN     {Fx'un bileşik tÜKisis
     REMHlssuti);
     CASE Isgunu OF
        Ctni ı Ekucret ı= Ekucrıt + Ucret(0.5);
        Pazar : Ekucret ı> Ekucret + Ucrtt(l.O);
       END; {Case için»
     Toplaflucrttı-Tcplaucrıt • Ucret(i.0)|
    EM};    {For'un sonus
 Toplaıucretı»Toplatucret + Ekucret;
 WTELN<'Haftalık Toplu Ücret «  Joplatucret);
END.   {Prograt tüıceıinin sonus
```

firntk prograıda görüldüğü gibi, sözcükleri bölutek koşuluyla programın isten» yerinde istenen likttrda boşluk bırakıak serbesttir. Bloklar arasında boşluklar bırakarak ve iete kalan blokları daha icriden yazıaya başlayarak progm daha kolay okunacak duruM getirilebilir.

2.3 Paskalın Hodulr Prograılaıaya Yatkınlığı

Paskalda, progrulari daha küçük progrıt basaıaklarına ayrıştırarak her bir prograı basaugı için bir altprogrn yaztak olanağı vardır. Altprograılar ISIM (procedure) veya işlev (funetion) yapısında olabilirler, ISIN ve işlevler kendi adı, bildiritleri ve tüace btilûaü bulunan ayrı bir progru yapısındadır, âletlerin ve işlevlerin kullanıluıı Paskala blok yapı özelliği kazandırır. Her bir blok içindeki bildirişlerle yeni yerel değişkenlerin kullanıltasına izin verir. Prograıların yukarıdan aşağıya, teatiden ayrıntılara doğru batauk basaeak altprogratlara ayrıstırılıası hu taşanımı, hM de yazılısını kolaylaştırır. Böyle nduler biçiadc yazılan progratlar, uzun olsalar bile kolay okunabilir ve anlaşılabilir.

2.4 Paskalda Veri Yapıları

Veri yapıları ve veri tipleri acısından Paskal, örnek alındığı Algol 60 ve Algol N'yc gön üstün nitelikler tafir. Paskalda gerctksayı (real), tatsayı (integtr), eantık (boolean) ve karaktr (charactr) oluk üzere dört te»l standart skalar veri tipi tanıtlıdır. Kullanıcının yeni «kalır veri tipleri tanıılayabilıesi Paskalın kullanıla özel değişik veri tipleriyle donatılabiltesini sağlar. Örneğin

```
TYPE Sun = (Ptesi,Sali,Cars,Pers,CuH,Ctesi,Pazar);
VAR  Isgunu : 6un;
```

bildiriilerinde, önce yeni bir skalar veri tipi olan 6un tanıtlanıştır. 6un tipi, sıralı yedi sözcükten olusturuleustur. Isgunu değişkeni, 6un'ü oluşturan yedi değerden (sözcükten) herhangi birini teesil edebilir. Böylece, prograeda Ucretibul adlı bir ISIM de bildiriliis IM

```
FOR Isgunu ı« PtHi TD CUM DO Ucretibul;
IF Isgunutttısi THEN NRITELUCHaftasonu');
Isgunut'Sali;
```

doğru progru tûtceleridir. Paskalın bu özelliğinin uygun secillis adlarla kullanılıası, yazılan progratların okunmını olağanüstü derecede kolaylaştırır, üstelik, bu stkilde derleyicinin bu tip skalar değişkenler için yanlış isİMİre izin vertMesi de sağlanır.

Paskalda yeni skalar tipler üretilirken, tanıtlı skalar tiplerin alt erillerinden (subrange) de yararlanılır.

```
TYPE 6un * (PtMi,8ali,Cars,Ptrs,CuM,CtMi,Pazar)|
     Haf tasamı • Cttsi .. Pazr;
     Sinavnotu • 0 .. 100 ;
W    Notortalausı : Sinavnotu;
     Bugin      t Haftasonu ;
```

Bu örnekte, Sinavnotu tipindeki değişkenlerin 0 ile 100 arasında bir tatsayı, Haftasonu tipindeki değişkenlerin ise daha önce 9un tipinin bildiriiindeki skalar erilinin Ctesi'den Pazar'a kadarki değerlerinden biri (Ctesi ya da Pazar) olabileceği bildiriliyor.

Bu durumda, program çalışırktn Bugün Ctesi'den önceki değerleri alacak oluru, değişkenin erit dışı değerinden ötürü prograı hata vererek durur. Alt ritli tiplr kullanmak, hata ayıklama illetini kolaylaştırır.

Daha yüksek düzeyli veri tipleri acısından incelenirse, Algol 60'ın kayıt, cümle, kütük ve göttergc vri tipleri Paskala genişletilerek uygulanıştır. Kayıt (rKord) veri yapısı, değişik tipteki değişkenlerden diziler (array) oluşturmada kullanılır.

```
TYPE Öğrencikaydi * RECORD
        adi :ARRAYçl .. 20Ç OF CHAR;
        yui JINTEGER;
        cinsi:(Kiz,Erkek);
        no :INTEGER;
        sini-f:1 .. 4;
          END;
VAR Ogr t Ogrencikaydi;
```

Bir kayıt yapısında, istenen alana yol adı (path name) verilerek erişilir.

```
Ogr.yasi ı* 17;
Ogr,cinsi t* Erkek;
```

Kayıtlar, yeni kayıtların tanılanıasında, veya kayıtlardan dizi oluşturmakta kullanılabilir.

```
TYPE Kayitdizisi > ARRAYçl .. 20C OF Ogrencikaydi;
VAR Gecenlr ı Kayitdizisi;
    1       : IHIEBER;
FOR I :« 1 TO 20 DO
        READLNI6Kenler.naçIC,6Kenlv.adicIC);
```

KÜM (set) vri tipi, küM ılcHnlarının hr biri bir bite karılık olan bit dizileri olarak grçeklntirilıiştir. Böylece Küne tipinde bir değişkenin değeri, bir ikalar tip erilinin (ringe) herhangi bir altküMsi olabilir.

```
TYPE Bun « (PtNi,Sali,Cars,Prs,CuH,Ctesi,Pazar);
     Bunler • SET OF Sun;
VAR  Zorgunler, Vrıiligunlr,
       İyigünler, Kotugunler :  Günler;

Zorgunler ı »ei'tesi, CuaaC;
Vritligunlr ı^ çCars, Prt, CUHC;
lyigunler ı« Vcrıiligunler • Zorgunler; Sarakesitş
Kotugunlr t= Zorgunler - Vrıiligunlr; Sfarkş
```

Bu örnekte, Bunler tipindeki değişkenler, bellekte yedi bitlik bir dizi olarak tutulacaktır. Zorgunler 1000100 dizisi, Verıiligunler 0011100 dizisiyle saklanacaktır, lyigunler, Verir ligunler ile Zorgunlr'in arakesiti olan OOOOlOO'diziıidir. Kotugunler ise Zorgunler'den Vcrıiligunler'deki eleaanların çıkarılMsıyla 1000000 olrak bulunur. Bu yaklaşıl, küM değişkenlri. *trnınit birini»* W, fark {-), arakesit (*) küt* işimleriyle eşitlik <«), kapsau «, <», >, >«) ve i'çer* (İN) gibi kÜMİi •antik illetlerinin tanılanabilıesini Mğlaıaktadır.

Paskalda kütük vri tipi diğr dillrdeki kütük tiplerin* göre sınırlıdır. Kütük Paskalda hr zatın homojen yapılı sıralı vri dizisidir. Bu dizinin bir anda yalnız tflçon değişkeni (buffr vviable) olarak adlandırılan bir tek elceını kul lanı ta acıktır. Tatpon değişkeni i İri gri kaydınak için BET veya PUT, en başa getirıek için ise RESET veya RENRITE gibi teael işletir kullanılır.

Göstrge (Pointr) vri tipi, gerçekti diğr yüksek düzey dillerindeki gibi kullanılıaz. Bu tip, çeşitli zincir yapılı (linked strueture) veriler olusturtak için düşünülmüştür. Örneğin, eleunları birr tasayı ile bir göstrgeden oluşan kayıtlardan tekli zincirlemiş bir liste oluşturıak için şu bildiri» kullanılabilir.

```
TYPE Sostrge * üEletan;
    Eletin  » RECORD
        Siyi : INTEGER;
        P    16ostrge;
    END,
```

Bu bildirisde 6ostrge, bir sonraki Eletin kayıdınm adresi olarak tanıtlandıktan sonra, Eletan kayıdı içinde Sostrge için ayrılacak alan da bildiriliştir. Böylece hr kayıttan sonra gelen kayıdın adresi tutulduğundan, bir sonraki kayıda kolayca erişilir.

Paskalın en tetel ve en önemli özelliklerinden biri prograı için gereken en uygun vri yapısının yukarıdan aşağıya sistematik bir yapı içerisinde geliştirilmesine olanak sağlaıasıdır. Vri tiplrinin çeşitliliği ve isteğe uygun şekillendirilcbİMSİ sayesinde progrular daha iyi düzenlenebilmekte, cıbuk anlaşılmakta ve gerekirse kolayca değiştiril tektedir.

2.5 Diğer özellikleri

İçerdiği denetil yapılarına ve vri tiplrine ek olarak Paskal, progratlata öğrenitinde hst kavratsal, net de uygulan yönünden önetli olan bir çok özellik taşımaktadır. Bunlar arasında işlevlrin birbiri içinde ardışık (itrative) kullanılabilıesi, önceden drlenıiş işlevlrin prograılara eklenebiltesi (linking), işlev ve işlemlerde değr ve değişken paratetre çeşitleri (cali by value, cali by refrence), işlev ve işlet isiilerinin paratetrik kullanılabileni, bildirilirin genel ve yerel özellikleri unutulması gereken önemli kavrulardır. Mgol 68 veya PL/I'ın yanında Paskalı kapsadığı kavraılar acısından oldukça basit bir dil olarak yorulatak tutkundur. Paskalın fortal semantik tanımlaması yalnızca 26 sayfaya sığmaktadır. Bu biçimde tat {orıal tanıt, K. Jenten ve N. Hirth'in Pascal User's Hanual and Report adlı kitabında bulunmaktadır.

3. Paskalın Bilgisayarlara Uygulanın

Paskal dilinin basitliği, onun tasarlanışında ikinci amaç olan tüt bilgisayarlara kolay ve verilli biçilde uygulanabilmesi acısından önem taşıyordu. CDC 6400'de yazılan ilk Paskal derleyicileri yaklaşık onbin assetbler satırından oluşuyor ve bellekte 40 kilobytc'lık yer kaplıyordu. Bu derleyiciler ortalama saniyede 80 progru satırını derleyebiliyordu. Paskal-2'nin derine hızı, Algol 68'e göre hemen heten yüzde 45 daha fazlaydı.

Paskal derleyicilerinin bu kadar küçük yere sığabilmesi ve bu derece verimli çalışabilmesi, Paskal dilinin cok temel sayılmayacak, yi da uygulamada asın yer gerektiren bazı kavramlar dışlanarak sağlanmıştır. Bunlardan en önemlileri:

1. Bercek Blok Yapısı: Algol 68 ve PL/1, gereken veri yapılarının herhangi bir blok içinde bildirilebilmesini desteklerken, Paskalda yerel değişmez, tip ve değişken bildirimleri ancak program, işlev veya işlem bloklarının (adlandırılmış blokların) başında yapılabilir.

2. Dinamik Dizi Kullanımı: Paskalda bir dizinin boyutlarının alt ve üst sınırlan tip bildiriminde açıklanmak zorundadır. 6u yüzden dizinin boyutlarını değiştirebilmek için programın değiştirilerek yeniden derlenmesi gerekir. Paskal, program çalışırken boyutları değişebilen dinamik dizilere izin versez.

3. Kalıcı Blok Değişkenleri *{m* variables): Paskal, Algol'daki yerel değişkenlerin her çağrılışlarında eski değerinden başlama özelliğini desteklemez.

4. Derleme Sırası Makroları: Paskal, Algolda olduğu gibi derleme sırasında makro tanımlama yapılmasını desteklemez.

Diller ve dil tasarımı üzerine çalışan uzmanlar, dilden hangi nedenle çıkarılmış olursa olsunlar bu çeşit eksikliklerin dilde bir boşluk yarattığını ileri sürmekteler. Bu görüşler Paskalın çeşitli amaçlara dönük yerel uzantılarının geliştirilmesine neden olmuştur. Deri i toplu yapısı nedeniyle Paskal derleyicileri değiştirme ve eklemelere uygundur. Paskala yapılan eklenti ve değişikliklerin başlıcaları şunlardır:

1. Ilkdeğer aktarma bildirimi;
2. CASE tümcesi için kullanım türü belirleyicisi;
3. Giriş çıkış tümcelerine formatlı kullanım yöntemleri;
4. Dinamik dizi parametreler olanağı;
5. Yapısal değişmez olanağı;
6. işlem ve işlevler için kalıcı blok değişkenleri;
7. READ/HRITE işlemlerine ek özellikler;
8. Sırasız erişimli ve indeksi i sıralı kütük kullanımı;
9. Yerpaylaşımı (över 1ay);
10. LABEL ve BOTO'nun kaldırılması;
11. Kanusıkuyı, dizgi (string) gibi tk tml veri tiplri;
12. üst alma IŞINI;

Bu çeşit yerel pek cok eklentiler yapılmış olması, değişik yerlerde geliştirilmiş Paskal yazılımların ortaklaşa kullanımını engellemektedir. Paskal dilinde çeşitli değişiklikler yapılmış olması, Paskal kullanıcılarının genel ve ortak bir dilde anlaşmalarını ve bu yönde örgütlenmelerini zorlaştırmaktadır. Pek çok küllinin, dilin tanımı olarak 1972'dm yayınlanmış olan düzeltilmiş raporu kabul etmektedir. Fakat gelişmeler, bu rapordaki formal yapı ile kullanımdaki genişletilmiş Paskal derleyicileri arasındaki farkların giderek arttığını göstermektedir.

4. Sonuç

Paskal, ortaya çıkışından bu yana bilgisayar dünyasına önemli etkilerde bulunmuştur. Bu dil, sistematik program yapısı oluşturmaya yönelik kavramlara sahip olması, ve geniş veri yapısı olanakları ile diğer yüksek düzey dilleri arasında önemli bir boşluğu doldurmuştur.

Paskalın her bilgisayara kolayca uygulanabilmesi, bu dilin çok kısa bir sürede yaygınlaşmasını sağlamıştır. Modüler ve sistematik programlamaya yatkınlığı, Paskalın programlama eğitiminde örnek olarak verilmesini, ve yayınlarda algoritma açıklama dili olarak yaygınlaşmasını sağlamıştır.

Bütün bu nitelikleriyle Paskal artık tanınmış tml yüksek düzey diller arasında yer almıştır.

Bu yazı, 6. Kchael Schneider'in Computer dergisi Nisan-1979 sayısındaki "Pascal: An Overvie»" yazısından yararlanılarak hazırlanmıştır.

KAYNAKLAR:

1. D. Cooper, H. Clancy; "Oh Pascal!", H.H. Norton t Company Inc.;NeKYxk, 1974

2. Findlay Hilliam; "Pascal, An Introduction to «ethodical Programming", Computer Science Press, Inc.; Haryland, USA;197B