SAÜ Fen Bilimleri Enstitüsü Dergisi
1 (1997) 13-18

VISUAL FOXPRO 3.0 İLE NESNEYE DAYALI PROGRAMLAMA VE HASTA TAKİP UYGULAMASI

Bülent Çağlar[1], Osman Çerezci[2]

[1] **Sakarya Üniversitesi, Sos. Bil. Ens., Üretim Yönetimi ABD,** ***Adapazarı***
[2] ***Sakarya Üniversitesi, Bilgisayar Mühendisliği Bölümü, Adapazarı***

ÖZET

Bu çalışmada; son yılların güncel bir programı olan Visual FoxPro 'nun temel özellikleri ve nesneye dayalı programlama kısmı tanıtılarak, güçlü bir veritabanı yönetim sistemi oluşturulmuş ve buradan hareketle sağlık merkezlerinde kullanılacak nesneye dayalı hasta takip uygulaması geliştirilmiştir. Diğer hasta takip uygulamalarına göre daha ergonomik ve daha kapsamlı bir program hazırlanmıştır.

I. GİRİŞ

Visual FoxPro; uygulama geliştirme ortamı olan güçlü bir veritabanı yönetim sistemidir. Veriye mouse ile doğrudan girilebilmesi mümkün olup hızı ve gücü sayesinde daha önce sadece sistem ile başarılan birçok veri yönetim işleri, masaüstü ile başarılmaktadır. Visual FoxPro 'nun temel bazı özellikleri aşağıda verilmiştir:

- Yeni bir sınıfı hızla oluşturmak için Visual FoxPro sınıf tasarımcısı kullanılabilir. Bu sınıfların aynı veya diğer uygulamalarda tekrar tekrar kullanıldığı için, programlama süresi önemli ölçüde azalır. Sınıf özelliklerindeki ve Visual FoxPro 'daki değişmeler ilgili alt sınıflara otomatik olarak yansır.
- Client/Server araçları ve dil gelişmeleri, verinin Xbase formatında depolanması ile birlikte, veriye uzak bir yerleşimden erişilmesini mümkün kılar.
- Client/Server araçları, lokal bir makinada bir protip uygulama oluşturulmasında ve sonrada son dakika kod değişikliklerini yapmaksızın bir Client/Server çevreye son uygulamaya hareket ettirmede kullanılır.
- Visual tasarım araçları (araç çubukları ve tasarımcılar), kuyruk ve rapor gibi konuları hızlı ve grafiksel olarak oluşturma ve düzenleme imkanına sahiptir.
- Veri sözlüğü, resmi ve geçerli olan iş kuralları açısından destek sağlar. Ayrıca 128 karaktere kadar uzunluk alan ismi kullanılabilir, tablolar arasında kalıcı ilişkiler oluşturulabilir.
- Modal olmayan uygulamalar ek kod olmadan yazılabilir. Bu husus kullanıcıya, aynı pencerede çok örnekler olsa bile çok kartlı pencereler açılma imkanını sağlar.
- Çok kullanıcılı sistemler çok az ek kod gerektirir. Visual FoxPro kullanmak suretiyle; tampon kilitleyici kayıt ve tablo takımı yapılabilir. Kullanıcının değişikliklerini başkalarınınkinden koruma için yapılabilir. Ayrıca işlem akışı; son başarısız uygulamaların geri gelmesine imkan verir.

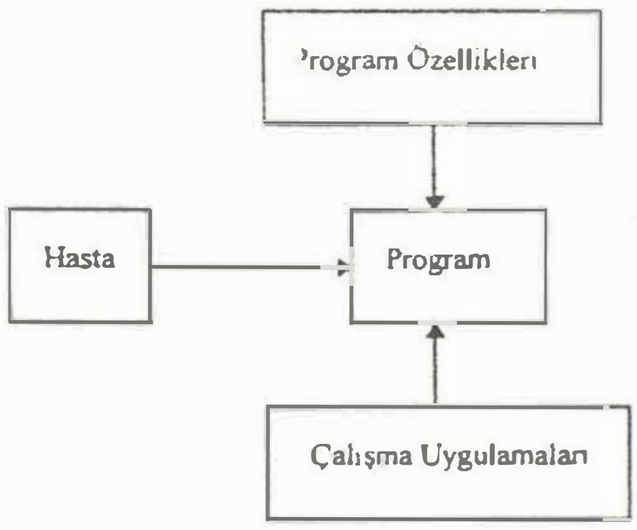
II. NESNEYE DAYALI PROGRAMLAMA

Nesneye dayalı programlama her yeni bir kavram gibi öğrenilecek yeni sözdizimlerine sahiptir. Bunlar aşağıda kısaca açıklanmıştır.

II.a. SOYUTLAMA

Soyutlama, karışık sistemin kolayca anlaşılabilen daha küçük bileşenlere bölünmesidir ve bir siyah kutu olup gerçek dünyadaki bir maddenin modelidir. Ve karışıklığı kullanıcıdan gizler. Bir konunun kullanılması nesne sadece nasıl yapar sorusu değil. Ne yapar? sorusunun bilinmesini gerektirir. Bir yönde ilerleyerek araba kullanırken, arabanın istenen yönde gitmesini sağlamak yeterlidir. Arraba ilerlerken motor içerisindeki işlevler, tekerlerin dönmesi gibi olayları bilmek arzu edilen yönde ilerlememiz için gerekmez[1].

Her yazılan programda soyutlama kullanılır. Basit bir soyutlama Şekil 1.1 de verilmiştir. Burada program belirlemesi ve hasta olmak üzere iki giriş ve çalışma uygulaması olmak üzere de bir adet çıkış vardır. Program belirlemeleri gıda çalışma uygulamalarına dönüşmekte ve özel bir şekilde programa girmektedir[2].



Şekil 1.1: Soyutlanmış Program

Ayrıca veritabanı uygulamaları tasarımlarında sistem verilerini alanlara dönüştürmek için soyutlama kullanılır. Soyutlama, sistemin fonksiyonel ayrıntılarını bir blok kod olarak ifade edinceye kadar basitleştirmede kullanılır.

İşlemsel dil olarak, sistem, proses ve prosedürlere soyutlanır. Nesneye dayalı dillerde, sistem nesneye soyutlanır.

Soyut veri tipleri, sayısal, karakter ve lojik gibi ana veri tiplerinden oluşan yeni veriler oluşturmayı sağlar. Ve aynı zamanda oluşan bu verinin gerçek dünyadaki verilerle uyuşmasını sağlar. Çoğu programlama dillerinde soyut veri tipleri uzun zamanda oluşturulur. Bir kişisel soyut veri tipi iki değişebilir karakter ve bir değişebilir tarihten oluşabilir. cİsim, cSoyad ve cDoğum gibi.

II.b. NESNE

Nesne; veri ve özelliklerin birlikte oluştuğu sınıflardan meydana gelir. Bir form bir nesne olup ve form üzerindeki kontroller de yine nesnedir[1]. Nesne ile bağlantılı terimler aşağıdaki gibidir:

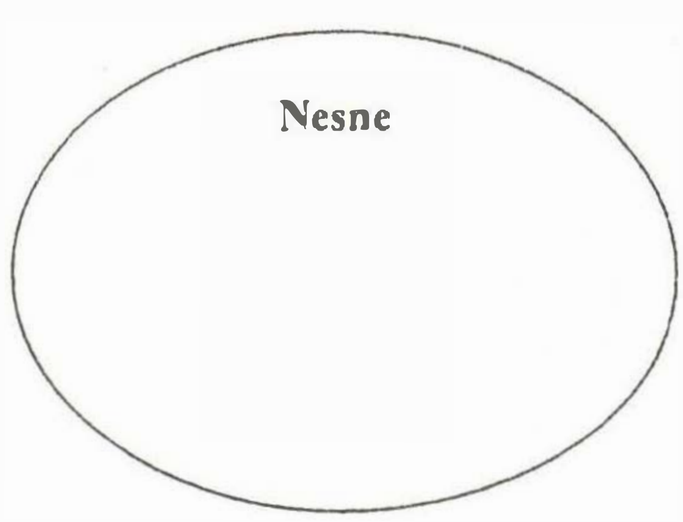
- ***Sınıflar ve alt sınıflar:*** Nesnelerin tasarımları veya modelleridir. Nesneler, uygulama anında sınıftan oluşturulur. Bir nesne; örnek tasarımın gerçeğe dönüştürülmesi sonucunda oluşur.
- ***Sınıf hiyerarşisi:*** Ana sınıf/alt sınıf/nesne ilişkisi anlamına gelir.
- ***Sınıf tasarımı:*** Nesnenin bir başka ifadesidir.
- ***Metotlar:*** Nesneye ilişkin programlar ve işlemler olup Command (komut) veya bir olay etkisi ile derlenir.
- ***Özellikler:*** Sınıf veya nesnenin davranış ve özellikleridir.

Hergün gerçek dünya nesneleri ile etkileşim içinde olunur. İki bilinen nesne olan araba ve bilgisayar örnek verilebilir. Her iki nesne birçok alt nesneleri kuşatır ve sonsuz sayıda ayrıntıları çalıştırır. Bu iki benzer örneğe göz atalım:

- Arabanız bir nesne olsun. Araba bir yerden başka bir yere seyahatte kolay ve güvenilir olarak kullanılır. Bunun için önce marşa basılır ve anahtarı çevrilerek motorun çalışması sağlanır. Motorda o anda olan karışık yanma olaylarını; elektriksel, kimyasal olayları düşünmek gerekmez. Tüm bu ayrıntılar arabanız tarafından kapsanmıştır. Basitçe şoför tutuşma anahtarı, gaz pedalı ve fren pedalı gibi nesnelerin standart arayüzeyleri ile etkileşir.
- PC yine bir nesne olsun. İlk keşfedildiğinden beri PC lerin entegre devre (ED) çiplerinde ve devre kartlarında önemli gelişmeler olmuş ve bu gelişmeler durmaksızın sürmektedir. Ancak başlangıcında da şimdi de standart ED nesneleri kullanılır. Silikon dioksitten oluşan bellek çipleri yapmak gerekmez. PC nesneleri düğmeye basılarak kullanıma başlanır. CPU 'larda yari iletken kapılarda elektron akışlarının fiziğini ya da monitörün arkasında yayılan ve çarpan elektronları, çarpan fosfor atomu elektronlarını, foton oluşumlarını göz önüne almak gereksizdir. Tüm bu ayrıntılar kapsanmaktadır. Sadece monitörde kelimelere, resimlere bakmak ve klavye veya mouse 'a dokunmak yeterlidir.

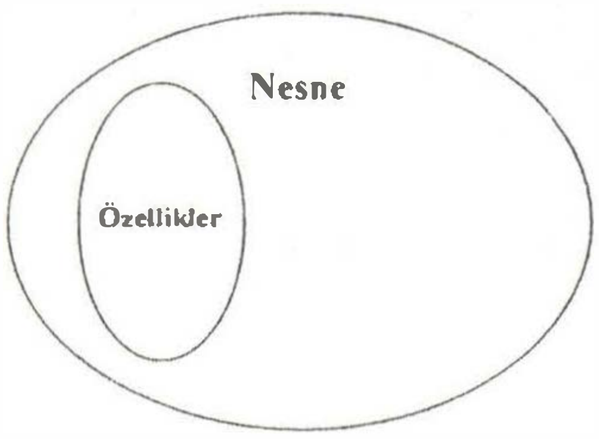
Bilgisayar ve otomobil imalatçıları, araba ve bilgisayarlar imaline her zaman yeniden baştan başlamazlar. Onlar, yeniden kullanılabilir parçayı raftan alır ve standart bir şekilde biraraya getirerek çok kısa zamanda çok daha fazla bilgisayar ve araba üretirler. Bu tekrar kullanılan bileşenler birçok farklı arabada kullanılan aynı motora veya birçok farklı bilgisayarda kullanılan aynı sürücü disklere kadar uzanır.

Visual FoxPro nesneleri benzerdir. Verinin bir tablodan giriş dosyasına nasıl ulaşacağının ayrıntısının gizlendiği karışık bir nesne oluşturulabilir. Bunun için nesnenin uygun bir ara kesit yüzlerini bilmek gerekir. Bir kere "nesneye çalıştır" denildiğinde onun ile ilgili ayrıntılar geride kalır. Yeniden kullanılabilir set oluşturulmuş olduğu için aynı nesne farklı programlarda uygulanabilir. Şekil 1.2 deki nesneye göz atalım. Onun bileşenlerini adım adım açıklayalım.



Şekil 1.2: Nesne Kapsamı

Nesneye dayalı programlama düşünüldüğünden de basittir. Şekilde elips cismin çevrelenmesini göstermektedir. Bu çevreleme nesneyi sınırlandırır. İlk bileşeni 1.3 de verilen özellikleridir.



Şekil 1.3: Nesne ile Özellikler

Özellikler nesnenin bildiği verilerdir. Özellikler:

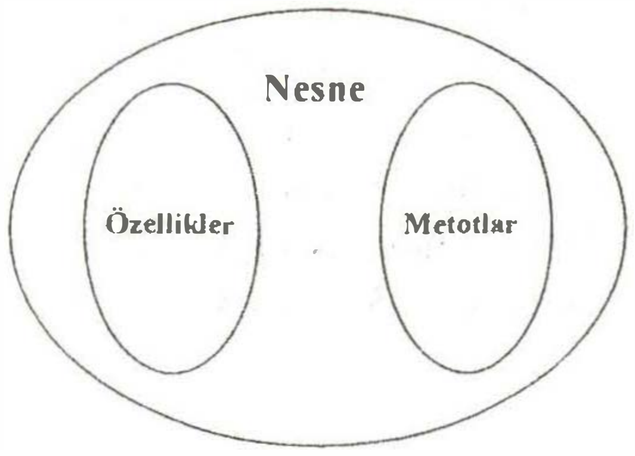
- İsimler
- Nesneyi tanımlayan davranışlar
- Aynı tip bellek değişken grupları
- Sayılar, karakter katarları, lojik ve diğer nesneler.

Bir kaç örnek özellikler ve değerler aşağıda verilmiştir:

• Ön renk (ForeColor)	rgb(128 , 128,0)
• Yükselik (Height)	47
• Font adı (FontName)	"Arial"
• Geçerli (Enabled)	.T.

Özellikler FoxPro 2.x de bellek değişkeninde depolanan aynı veri çeşitlerini içerir. Ancak Visual FoxPro da onlar nesne içinde çevrelenir. Çevreleme diğer nesnelere veya programlara karışmayı önler. Nesnenin özelliklerinin çevrelenmesi yoluyla onuu, diğer programlarda yapacağımız değişimlerle etkileşmesi önlenir. Bu problemleri kapsayan nedenlerden dolayı FoxPro 2.x kod birçok kere başarısız olmuştur.

Nesneleri oluşturmada blueprint kullanılır. Aynı sınıftan oluşturulan farklı cisimler, bazı farklı değerlere ya da tüm özelliklere sahiptir. Örneğin bir form daki farklı Command Butonları farklı başlıklara sahiptir. Birisi "Tamam" ise diğeri "İptal" dir.



Şekil 1.4: Nesne ile Mesajlar

Tüm bu özelliklerin nesneye yerleşmesi büyük bir iş olmakla beraber, işlenemez yetenekteki bilgi sınırlı kullanımdan doğar. Daha sonraki nesne bileşenleri Şekil 1.4 de gösterilmiştir.

Bir metot nesne tarafından oluşturulabilen bir fonksiyon olup aşağıdaki özelliklere sahiptir.

- Eylemler
- Davranışlar
- Programlar (kod, fonksiyonlar, prosedürler, sabrutinler)

Bunlar nesne özelliğini sağlar ve diğer cisimlerle iletişim sağlarlar

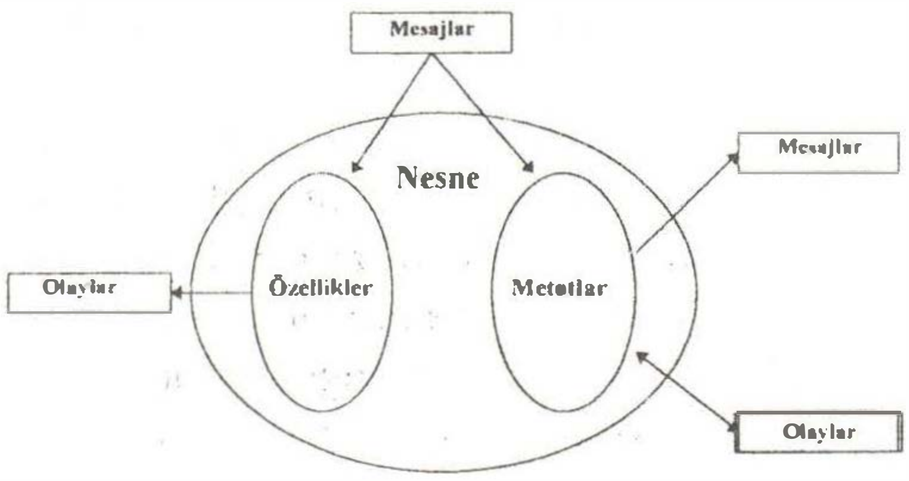
Bazı örnek metodlar:

- Click()
- Valid()
- Show()
- Move()
- InteractiveChange()

Aynı sınıftan oluşturulan farklı nesneler tamamen aynıı metotları paylaşacaktır.

Nesneleri kavramakta iyi bir yol onları yaşayan canlılar olarak incelemektir.

Diğer nesne bileşenleri mesajlar ve olaylar olup Şekil 1.5 de gösterilmiştir.



Şekil 1.5: Nesne Mesajları ve Olayları

Mesajlar, nesnelerle iletişim sağlayan yegane mekanizma olup, nesneyi kuşatan sınırı geçebilirler. Mesajlar özellikleri değiştirmede, geri döndürmede, metotları derlemede kullanılır.

Bazı örnek mesajlar:

- cmdOk.Top=0
- m.lnHeight = txtName.Height
- CustomerForrm.Refresh()

Mesajlar nesnenin özellikleri ve metotları ile etkileşir. Mesajlar yapılmış veri elemanlarına benzer ve isim parçalarını ayırmak için "." kullanır. Aynı zamanda fonksiyon çağırmaya benzer.

Bazıları bir nesne metotları çağırmak için mesaj terimini kullanmayı sınırlandırılar. Bu cismin sınırının dışından uygun değerlerin alınmasını sınırlandırır.

Bir olay; mesajın nesneye gönderilmesine yol açan kullanıcı eylemi veya sistem olayıdır. Olaylar Visual FoxPro uygulamalarında esas tahrik kuvveti olup kesilir ve belirli durum mesajları nesne metotları ile toplanır.

Bazı örnek olaylar:

- Sol mouse düğmesine basma
- T düğmesine basma ve bırakma
- Odakları azaltan bir giriş bölgesi
- 500 mili saniye zaman aralığı
- Nesnenin meydana gelişi.

den oluşur.

Nesneler diğer cisimlerden, .PRG koddan, ya da Command penceresindeki bir olaydan mesajları alır. Bu mesajlar özellik değerini değiştirebilir, özellik değerini geri gönderebilir ya da derleme için bir metota yol açar. Nesne ayrıca mesajı diğer cisme de gönderebilir. Olaylar uygulama için metota sebep olur ve bazı durumlarda metot içindeki kod diğer olaylara sebep olabilir. Ayrıca bu özelliklerin değişmesi oluşabilecek olayların sebebi de olabilir.

II.c. NESNEYE DAYALI ÜÇ TEMEL KAVRAM

Dil açısından nesne dayamada göz önüne alınacak husus; onun çevremele, polimorfizm ve kalıtsallık kavramlarını desteklemesidir. Çevreleme, nesneye giren veri ve kodu birleştirmektir. Polimorfizm benzer iç yapılarda farklı nesneler vermektir. Kalıtsallık ise diğer nesnelerden yeni nesneler yapmaktır. Aşağıda bu üç özellik ayrı ayrı ele alınmıştır.

Çevreleme: Bir nesnenin değişken takımı anlamında kendi içeriğini gösterir. Kullanıcı sadece, nesnenin nasıl kullanılacağını bilmesi gerekir. Nesnenin karakter ve fonksiyonelliği nesne tanımında verilir. Kod ilave ederek soyut veri tipine kadar genişler. Örneğin; komut butonu üzerinde Başlık özelliği yapıldğında nasıl bir katar depolanmış olduğunu bilmek gerekmez.

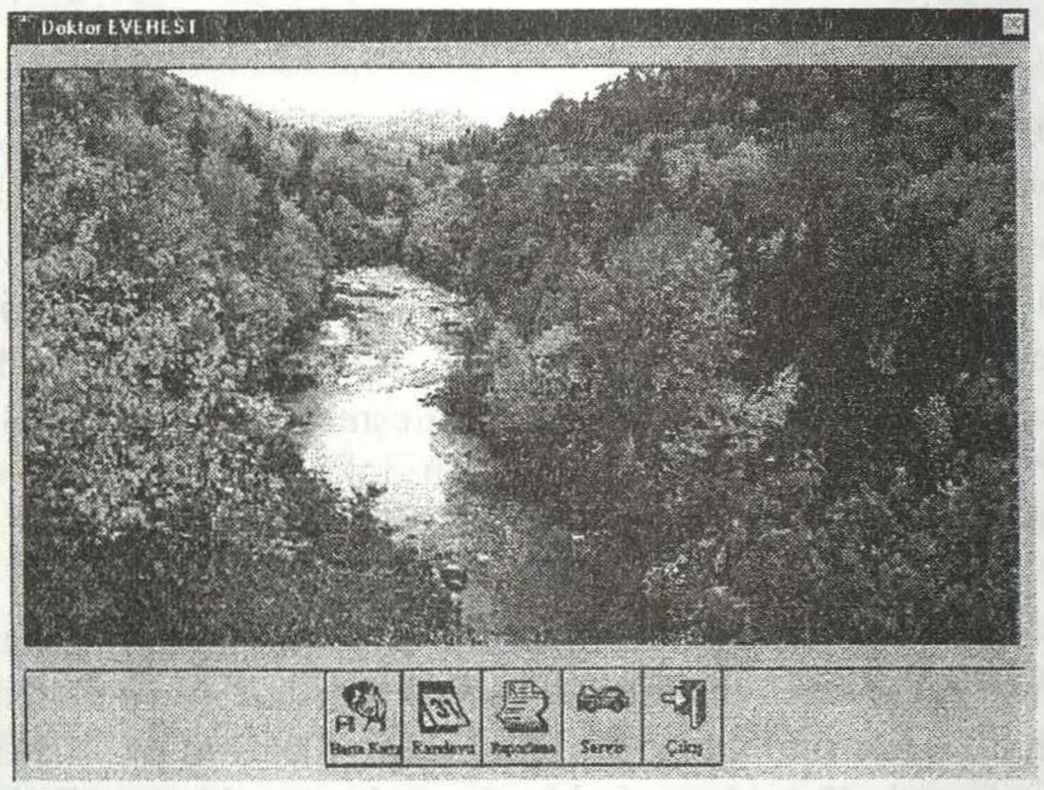
Polymorphism: Aynı isme sahip olma yeteneği olan polymorphism; nesnelerin aynı isimle metotları çağırabileceği, fakat davranışların nesnenin fonksiyonuna bağlı olmasını ifade eder ve ilgili sınıflar için içerikler farklıdır. Kulanım prosedürü, nesne sınıfı yoluyla çalışma zamanında tayin edilir. Bir parametre gibi nesneyi geçen prosedür, nesne parametresinin hangi tipte olduğunu bilmeye gerek duymadan çizim metodunu çağırabilir. Gerçek hayattan bir örnek olarak; siz üç kişiye depoya gitmesi söylendiğinizde, biri yürüyerek, diğeri bisiklete binerek ve diğeri arabaya binerek gidebilir fakat üçüde verilen görevi yerine getirmiş olması verilebilir.

Kalıtsallık: Nesneleri özelliklere ve sınıf metotlarına alma yoludur. Alt sınıflar taban sınıflardan gelen kalıtsal özelliklere sahiptir. Nesnelerde alt sınıflardan kalıtsal özelliklidir. Eğer ana sınıfta değişiklik yapılması istenirse ana sınıfa ait alt sınıf ve nesneler otomatik olarak değişmeyi yansıtır. Örneğin siz yeni özellik düzeltme kontroluna IsBold ilave ederseniz, sizin kontrolünüz altındaki herhangi bir alt sınıfta IsBold özelliği taşır.

III. HASTA TAKİP UYGULAMASI

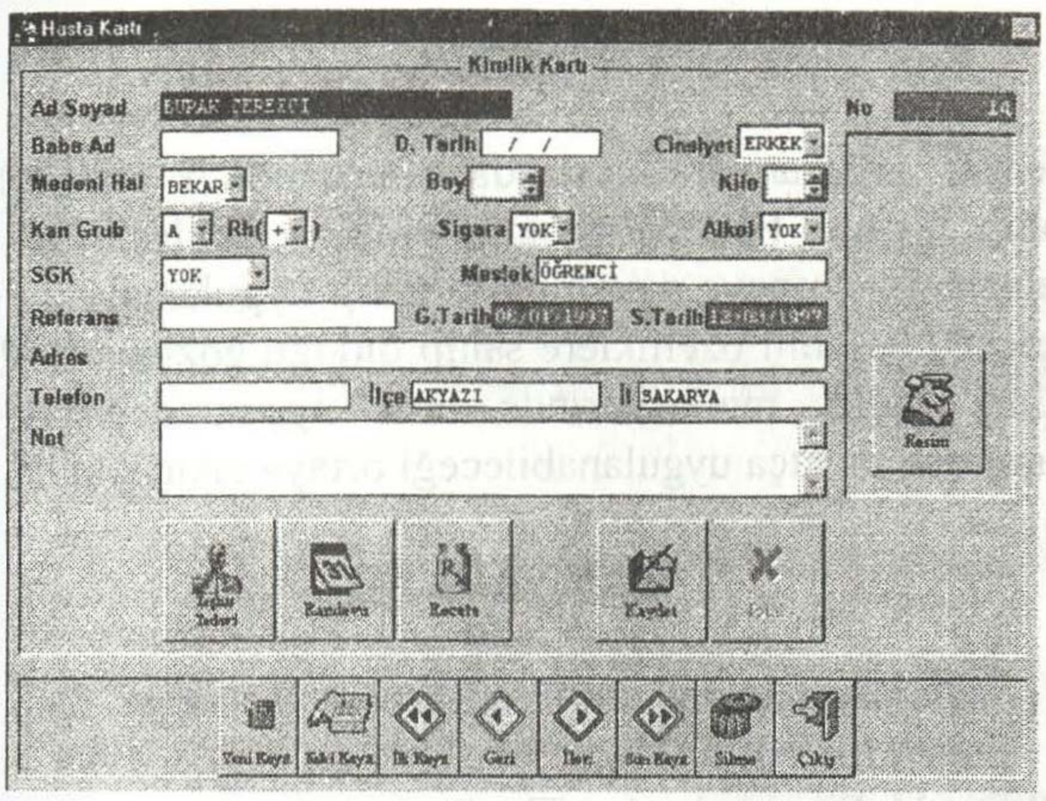
Visual Foxpro ile yazmış olduğumuz hasta takip uygulamasındaki ilişkiler aşağıdaki şekillerde gösterilmiştir.

Ana menü "Hasta Kartı", "Randevu", "Raporlama", "Servis" ve "Çıkış" nesnelerinden oluşmaktadır (Şekil 1.6)



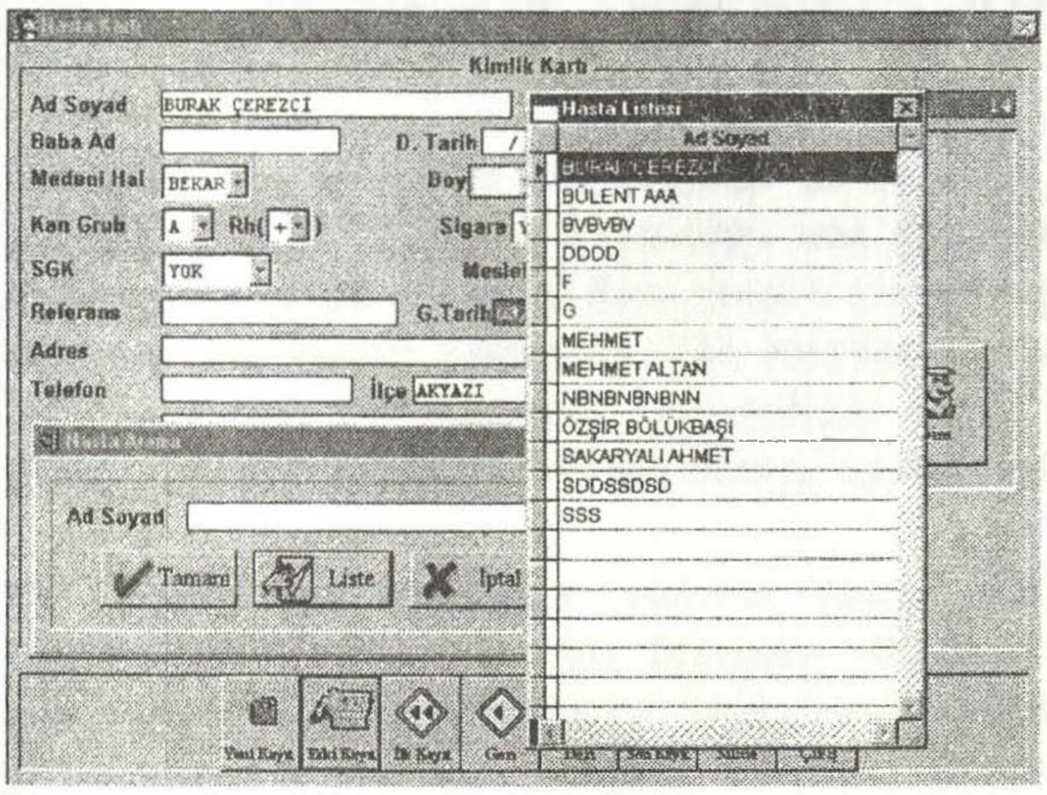
Şekil 1.6: Hasta Takip Programı Ana Menüsü

Hasta kartı formunda "Teşhis ve Tedavi", "Randevu" ve "Reçete" kartlarına geçilebilir. Eğer İlçe-İl girilirse daha sonraki hasta 'da daha önce veri tabanında depolanmış olan İlçe girildiğinde İl otomatik olarak ekrana gelmektedir. Resim butonu tıklatılırsa hastanın fotoğrafı kimlik kartına getirilebilir (Şekil 1.7).



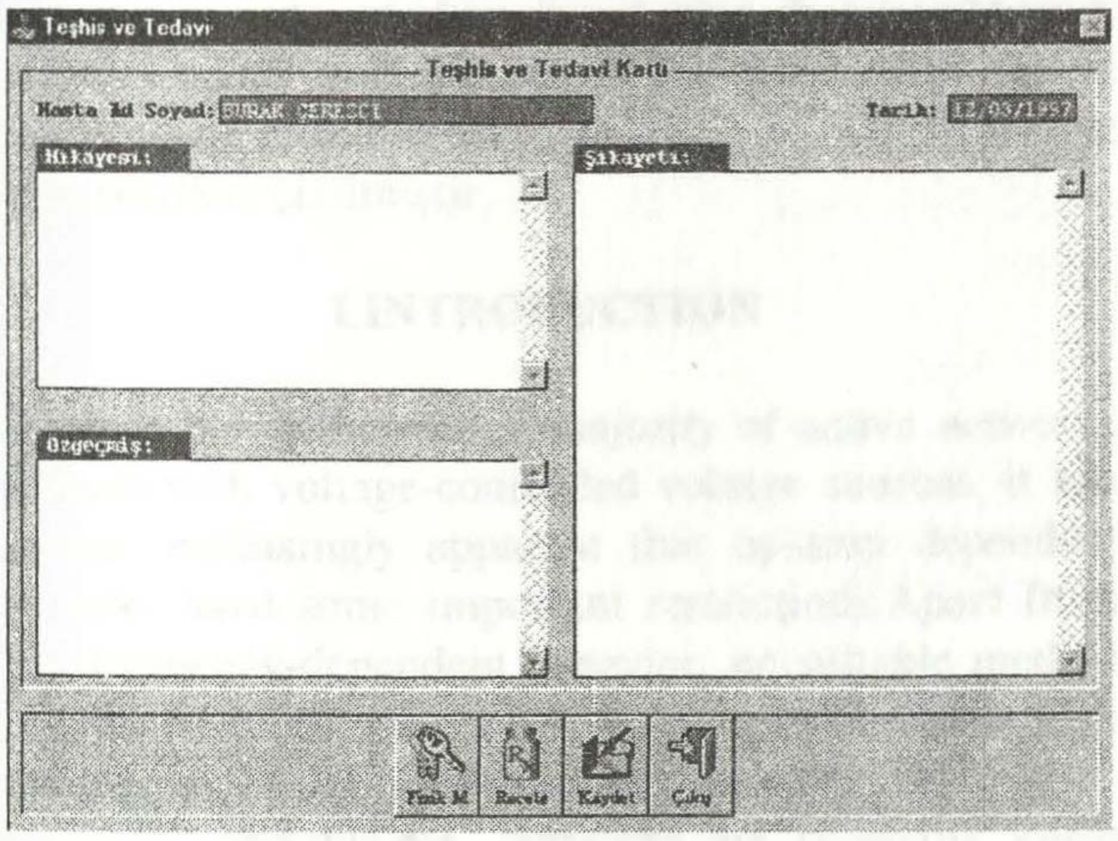
Şekil 1.7: Hasta Takip Programı Hasta Kartı Formu

Ayrıca "Yeni Kart" butonunu tıklatıp "Liste" seçilirse, hasta listesinin tümü ekrana gelir ve mouse 'un sağ tuşuyla seçim yapılabilir (Şekil 1.8).



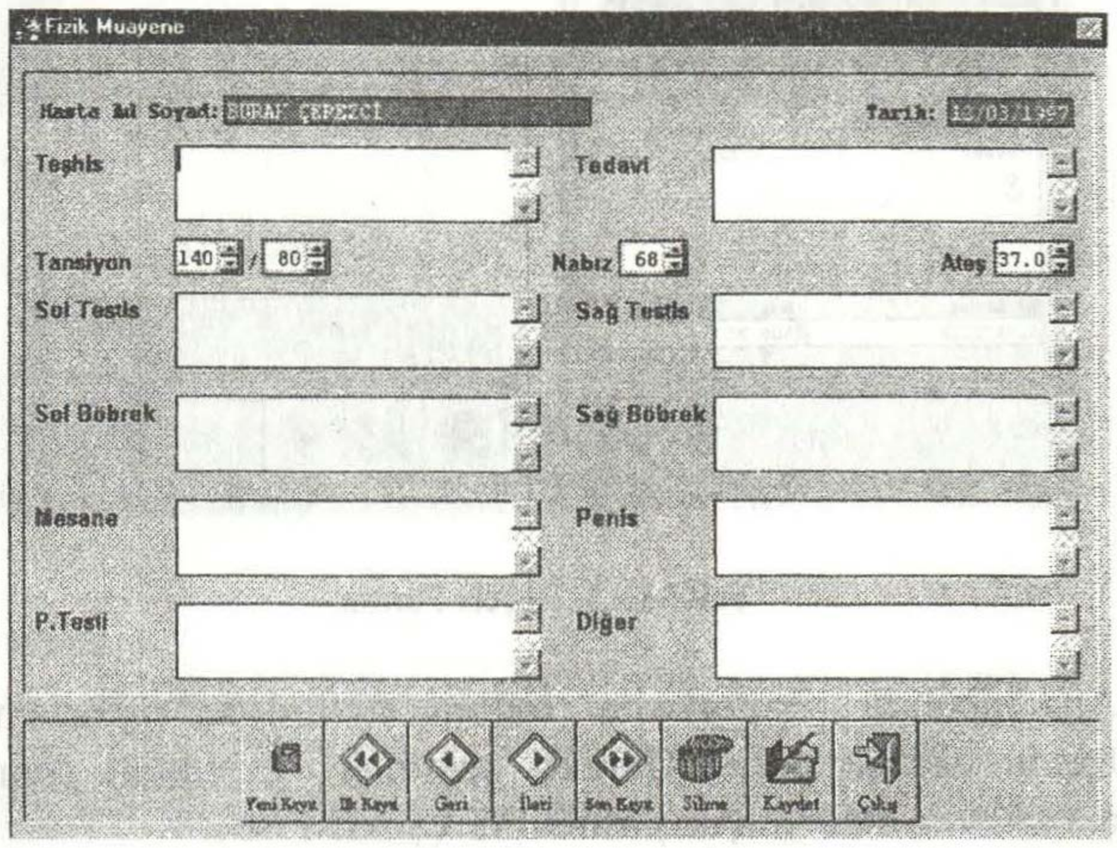
Şekil 1.8: Hasta Kartı Formunda Yeni Kart 'ta Liste Seçeneği

Hasta kartı formunda "Teşhis ve Tedavi" butonu tıklatılırsa teşhis ve tedavi formu ekrana gelir (Şekil 1.9).



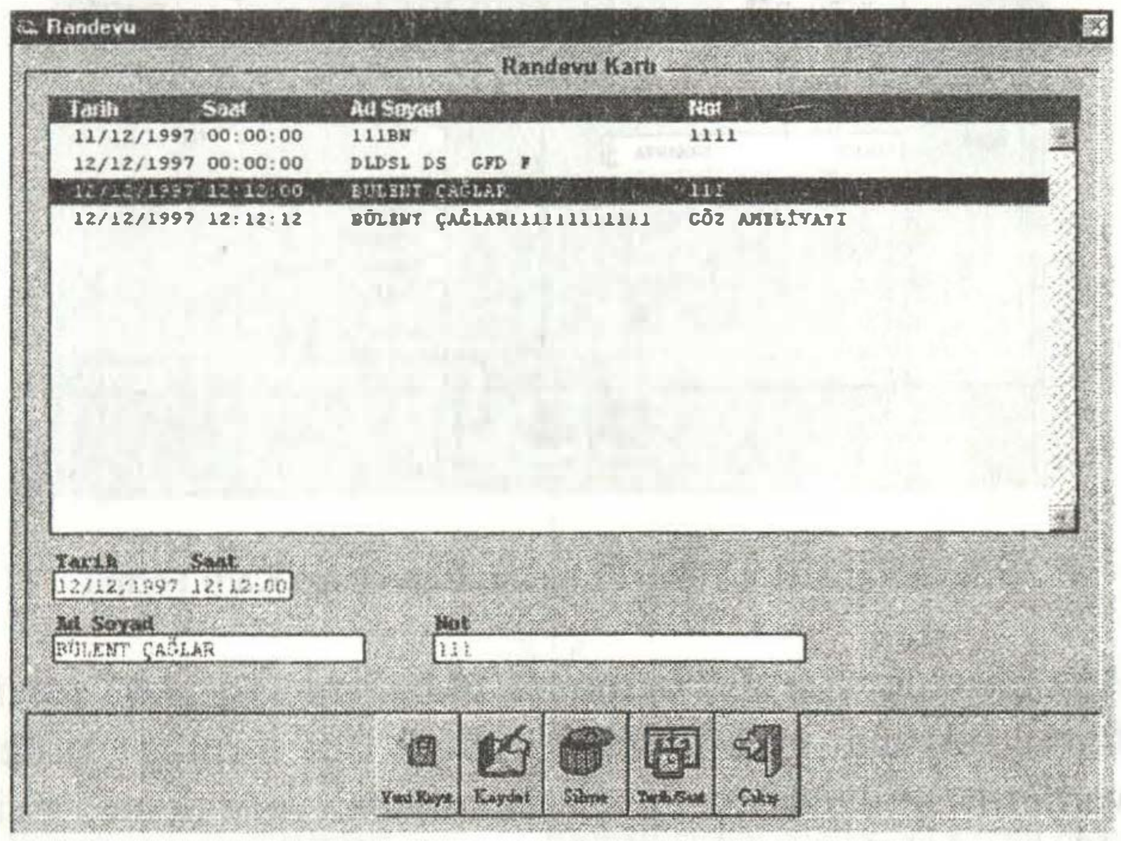
Şekil 1.9: Teşhis ve Tedavi Formu

Teşhis ve tedavi formunda "Fizik M" butonu tıklatılırsa fizik muayene kartı ekrana gelir (Şekil 1.10). Ayrıca bu formda hasta daha öncede gelmişse alttaki butonlarla eski kayıtlarıda izlenebilir.



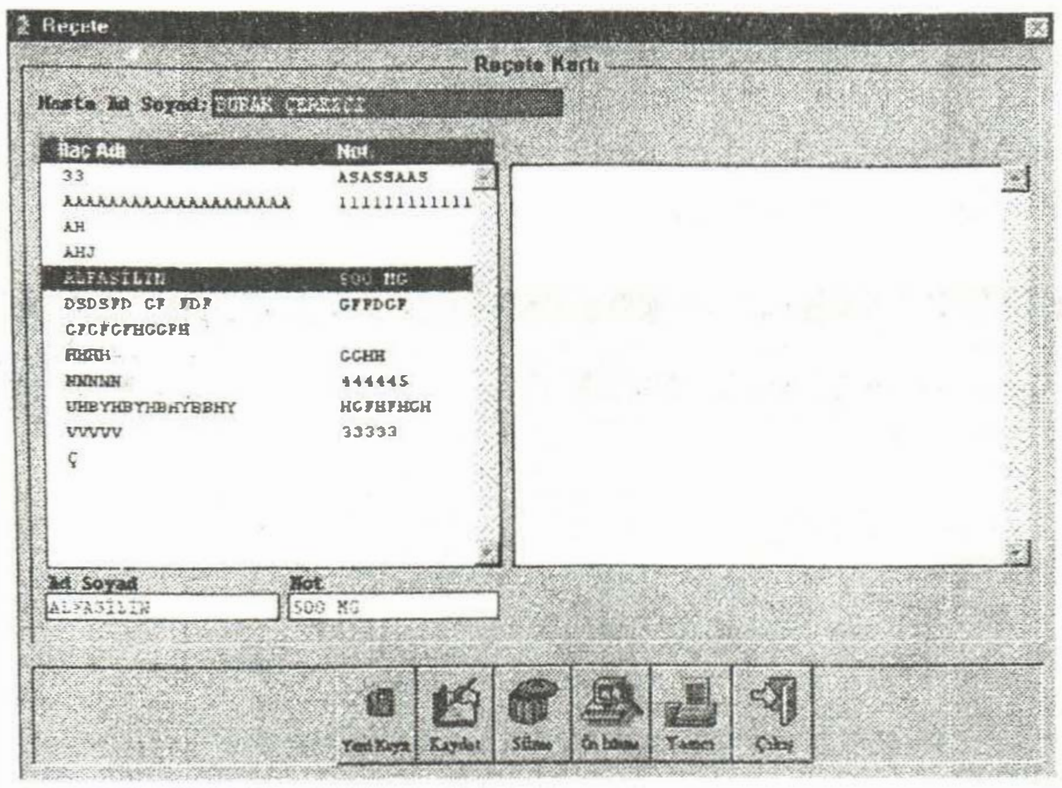
Şekil 1.10: Fizik Muayene Formu

Ana menü veya hasta kartında "Randevu" butonuna tıklatılırsa randevu formu ekrana gelir (Şekil 1.11).



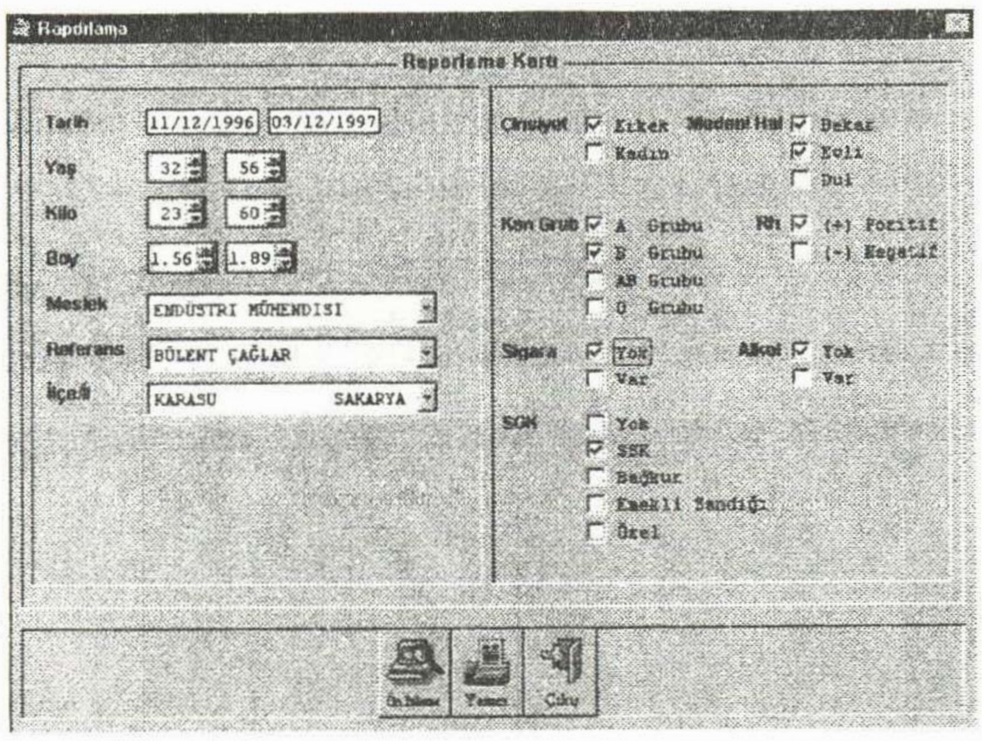
Şekil 1.11: Hasta Takip Programı Randevu Formu

Hasta kartında "Reçete" butonuna basılırsa reçete formu ekrana gelir. Mouse 'un sağ tuşuna basılırsa ilaçlar ekranın sağına aktarılır. Buradan da yazıcıya çıkış alınabilir. Ayrıca bu formda ilaçta ilave edilebilir.



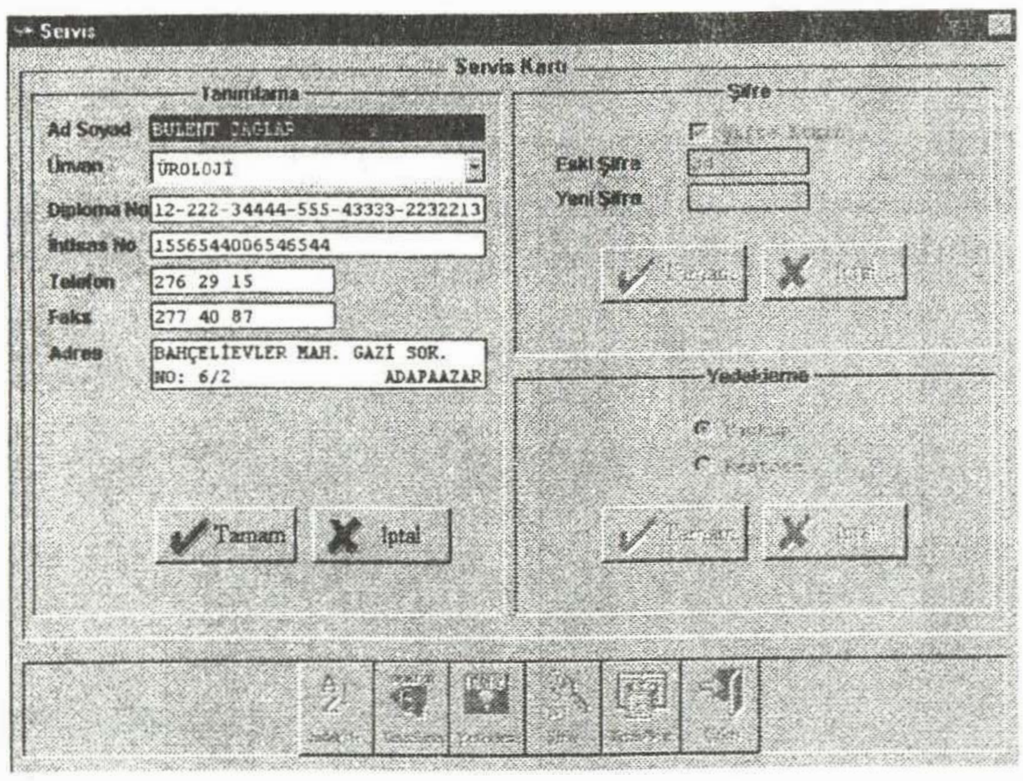
Şekil 1.12: Reçete Formu

Ana menüde "Raporlama" butonu tıklatılırsa raporlama formu ekrana gelir. Gerekli seçimleri yaptıktan sonra ekrana ve yazıcıya çıkış alınabilir (Şekil 1.13).



Şekil 1.13: Hasta Takip Programı Raporlama Kartı Formu

Ana menüde "Servis" butonuna tıklanırsa aşağıdaki servis formu ekrana gelir (Şekil 1.14). Bu formda girilen verilerin sıralanması, genel tanımlama, yedekleme, giriş şifre değişikliği ve tarih/saat ayarı yapılabilir.



Şekil 1.14: Hasta Takip Programı Servis Formu

4. SONUÇ

Benzer programlara göre daha ergonomik özelliklere sahip Visual FoxPro programında gerçekleştirilen bu çalışmada, nesneye dayalı hasta takip uygulamasında son derece kapsamlı özelliklere sahip olduğu gözlenmiş olup, bütün sağlık merkezlerinde bazı uygun değişiklikler yapılarak rahatça uygulanabileceği ortaya çıkmaktadır.

KAYNAKLAR

[1] "Microsoft Visual FoxPro 3.0 For Windows Step By Step", Microsoft Press, Washington, ABD, 1995.

[2] David, Frankenbach, "The Pros Talk Microsoft Visual FoxPro 3.0", Microsoft Press, Washington, ABD, 1996.