İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

WINDOWS İŞLETİM SİSTEMİNDE ÇALIŞABİLEN UYGULAMALARDA DİNAMİK VERİ DEĞİŞ TOKUŞUNUN KULLANIMI

YÜKSEK LİSANS TEZİ

Müh. Mustafa İŞBİLEN

Tezin Enstitüye Verildiği Tarih : 13 Haziran 1994

Tezin Savunulduğu Tarih : 10 Ekim 1994

Tez Danışmanı : Doç. Dr. Nadia ERDOĞAN

Diğer Jüri Üyeleri : Prof. Dr. Nadir YÜCEL

: Doç. Dr. Bülent ÖRENCİK

EKİM 1994

ÖNSÖZ

Günümüzde dördüncü seviye dillerinin geldiği düzeyi ilgiyle izlemekteyiz. Bilgisayar dünyasında, grafik ortamda çalışan arayüzler kullanıcının daha çok hoşuna gitmektedir. Yeni yazılım geliştirme ortamları (API = Application programming interface) nesne yönelimli yapılarıyla kullanımı kolay ve zevkli yazılımlar üretme olanağı veriyorlar. API kendi kodunu üretiyor. Kod üretme kolaylaşmış gibi görünse de kullanıcıların istekleride artmaktadır. Yazılımcının daha fazla nesnenin kontrolunu düşünmesi gerekmektedir.

Yazılım paketleri gittikçe özelleşmektedir. Kelime işlemciler, elektronik tablo programları, hizmet programları, vb. kendi aralarında gelişmekteler. Böyle özelleşen paketlerin hepsi ile haberleşebilen arayüz şeklinde uygulamalar geliştirmek mümkündür. MS Widows 3.1 işletim sistemi ve burada çalışan paket yazılımlar İstemci/Sunucu yapısında haberleşme olanağına sahipler. Bu sayede bir uygulama, tasarlandığı amacın dışında, özellikle zayıf bırakılmış yanını diğer uygulamayla bağlantı kurarak bir ölçüde kapatabilir. Bu haberleşmeyi sağlayan protokol sistem tarafından desteklenir. Alt düzeyli ileti komutları vardır. Komutları bu haliyle kullanmak zor olduğu için DDE'yi (Dynamic data exchange) kotaracak, içinde fonksiyon çağrıları olan DDEML.DLL kütüphanesi kullanılır.

Tezin içeriğinde; DDE'yi açıklayıp, yazılım geliştirenler için bunu pratiğe aktarmada yöntemler anlatılmaktadır. DDE konusunda Türkiye'de güncel kaynak bulmak pek kolay değil. Burada; yardımlarından dolayı tez hocam Doç. Dr. Nadia Erdoğan'a, tezim için yaptığım araştırmalarda beni kendi imkanlarından faydalandıran TRIO Çözüm Evi Bilişim Hizmetleri A.Ş.'den Tayfun Yiğit'e , SF Ses ve Işık Sistemleri'nden Kadri Çakmak'a teşekkürlerimi sunarım.

İÇİNDEKİLER

ÖZET

Tezin genel amacı DDE'yi (Dynamic data exchange) açıklayıp, yazılım geliştirenler için bunu pratiğe aktarmada yöntemler verebilmektir.

İlk olarak DDE'nin terminolojisini oluşturan kavram ve kelimeler verildi. DDE; Windows'da (MS Windows 3.1 işletim sistemi) prosesler arası haberleşmede desteklenen üç mekanizmadan biridir. Diğer iki mekanizma, Windows Clipboard ve dinamik bağlantı kütüphaneleriyle (DLL) paylaşımlı bellek kullanılmasıdır.

Windows'da çalışan iki program bir DDE oturumunda birbirlerine iletiler postalayarak çalışırlar. Bunlardan oturumu başlatan; İstemci, hizmet veren; Sunucu'dur. Windows Sunucu programı, diğer Windows programının yararlanacağı verilere erişen programdır. DDE İstemcisi ise verileri elde eden programdır.

Windows programı, bir programa göre istemci diğerine göre sunucu olarak davranabilir. Böyle olursa iki ayrı oturumun kurulması gerekir. Sunucu uygulama, birden fazla istemciye veri gönderebilir ya da bir istemci birden fazla sunucudan veri alabilir. O zaman birden fazla DDE oturumlarının kurulması gerekir. Bu oturumların biricik (unique) ve ayrık olmalarını sağlamak için istemci/sunucu üzerindeki her oturum farklı bir pencere kullanır.

DDE oturumunda çalışacak programların kodlarının özel bir biçimde yazılması gerekir. DDE sunucusunun kullandığı verinin biçimi ile (format) istemcinin istediği aynı veya dönüştürülebilir olmalı. Temel olarak üç çeşit DDE oturumu vardır; soğuk bağlantı, sıcak bağlantı, ılık bağlantı.

DDE'nin Visual Basic ve Excel üzerinde uygulanması açıklandı. Sonra buraya kadar anlatılanları kapsayan TEZDEMO.MAK/EXE isimli bir uygulama geliştirildi. 5. bölüm bu uygulamaya ayrıldı. Ayrıca program kodu içinde verilen açıklamalara ek olarak TEZDEMO'nun kullanımı için de akış diyagramı verildi. Yazılım geliştirenler , bu uygulamada kullanılan temel işlemlerden faydalanıp çeşitli yazılımlar hazırlayabilirler.

SUMMARY

USING OF DDE IN APPLICATIONS WHICH ARE RUNNABLE ON WINDOWS OPERATING SYSTEM

Interprocess communication between two Windows applications using DDE (Dynamic data exchange) is called a DDE conversation. The conversation is initiated by the DDE client application requesting that a DDE server application open a DDE channel over which to hold the conversation. DDE clients and servers are occasionaly referred to as DDE source and destination applications, respectively, paralleling OLE (Object linking & embedding) terminology.

The relationship between DDE clients and servers is similar to that between workstations and database servers. The primary difference is that both the DDE client and server applications must be running on the same computer, because the transfer of data between applications takes place through the Windows Clipboard. The single-computer restriction does not apply if you use the Network DDE features on Windows for Workgroups.

In addition to transferring data from and to DDE server applications, you can execute instructions, usually in the form of the menu commands in DDE servers that support the DDE execute statement.

DDE is one of three mechanisms of interprocess communication supported under Windows. The other two are the Windows clipboard and shared memory in dynamic link libraries.

DDE is based on the messaging system built into Windows. Two Windows programs carry on a DDE conversation by posting messages to each other. These two programs are known as the server and client. A DDE server is the program that has access to data that may be useful to other Windows programs. A DDE client is the program that obtains this data from the server.

A single Windows program can be both a client to one program and a server to another, but this requires two different DDE conversations. A server can deliver data to multiple clients, and a client can obtain data from multiple servers, but again, this requires multiple DDE conversations. To keep this conversations unique and separate, each conversation (on both the client and server sides) uses a different window. Generally, a program that supports DDE will create a hidden child window for each conversation it maintains.

Because DDE uses the messaging system built into Windows, it fits very naturaly in the environment. But this is not to say that DDE is easy to implement. The protocol has many options, and programs must be ready to deal with some rather tricky problems.

The Types of Conversation

There are three basic types of DDE conversations; cold link, hot link, warm link. These conversations use DDE messages defined in the DDE.H header file. The simplest of the three conversations is known as cold link.

1) The Cold Link

A cold link conversation begins when a client broadcast a WM_DDE_INITIATE message identifying the application and topic it requires. (The application and topic may be set to NULL to begin a conversation with any server application or any data topic.) A server application that supports the specified topic responds to the client with a WM_DDE_ACK (acknowledge) message.

The client then requests a particular data item by posting a WM_DDE_REQUEST message. If the server can supply this data item, it responds by posting a WM_DDE_DATA message to the client. The client can acknowledge to the server that it has received the WM_DDE_DATA message. The server indicates whether it wants this acknowledgement in a flag passed with the WM_DDE_DATA message. A flag passed with the WM_DDE_ACK message indicates a positive acknowledgement.

If the client posts a WM_DDE_REQUEST message to the server, and the server can not supply the request data item, then the server posts a negative WM_DDE_ACK message to the client.

The DDE conversation continues with the client posting WM_DDE_REQUEST messages to the server -for the sama data or different data items- and the server responding with WM_DDE_DATA or WM_DDE_ACK messages. The conversation is terminated when the client and server post each other WM_DDE_TERMINATE messages.

2) The Hot Link

One problem with the cold link is that the data the server has access to may change with the passing of time. In the cold link, the client does not know when the data changes. The hot link solves this problem.

Again, the DDE conversation begins with a WM_DDE_INITIATE message and a WM_DDE_ACK message. The client indicates the data item it requires by posting a WM_DDE_ADVISE message to the server. The server responds by posting a WM_DDE_ACK message indicating if it has access to this item.

A positive acknowledgement indicates that the server can supply the data; a negative acknowledgement indicates that it cannot. At this point, the server is obligated to notify the client whenever the value of the data item changes. This notification uses a WM_DDE_DATA message, to which the client (based on a flag set in the WM_DDE_DATA message) may or may not respond with a WM_DDE_ACK message.

When the client no longer wishes to be advised of updates to the data item, it posts a WM_DDE_UNADVISE message to the server, and the server acknowledges. The conversation is terminated with the posting of WM_DDE_TERMINATE messages.

The cold link and the hot link are not mutually exclusive. During a single DDE conversation, a client may ask for some data items by using WM_DDE_REQUEST (for a cold link) and ask for others by using WM_DDE_ADVISE (for a hot link).

3) The Warm Link

The warm link combines elements of the cold link and hot link. The conversation begins as normal. As with the hot link, the client posts a WM_DDE_ADVISE message to the server, and the server acknowledges either positively or negatively. However, a flag passed with the WM_DDE_ADVISE message indicates that the client wishes only to be informed of changes in data without immediately receiving the new data item. So the server posts WM_DDE_DATA messages with NULL data.

Now the client knows that a particular data item has changed. To obtain this item, the client uses a WM_DDE_REQUEST message, just as in cold link.

As in the hot link, a client can stop being advised of changes in data items by posting a WM_DDE_UNADVISE message to the server. The conversation is terminated with the posting of WM_DDE_TERMINATE messages.

These three types of conversations use all the DDE messages except two: WM_DDE_POKE (in which a client gives a server unsolicited data) and WM_DDE_EXECUTE (in which a client sends a command string to a server).

Initiating a DDE Conversation with Service and Topic Names in VB

The DDE server that participates in the conversation is identified by its Service name; in versions of Windows prior to 3.1, Service name was called Application name, Service name is usually, but not always, the name of the application's executable file, without the .EXE extension. Service name is the name by which the Windows Task Manager recognizes the application. If the client request can not obtain a DDE channel within a fixed time period, called the DDE timeout interval, you receive an error message.

If the server application is not running, you need to launch the application with Visual Basic's Shell() function before you attempt to initiate a DDE conversation. Setting the value of the LinkMode property to a positive integer (1,2 or 3) does not cause Windows to automatically launch the application.

You set the value of Visual Basic's LinkTopic property to the combination of the DDE Service name and the DDE Topic name, separated by a pipe character. If the document is a file, you need to specify a well-formed path to the file and the file name as the Topic name. The following two statements open a DDE channel from a control object (ctrDDELink) to an Excel worksheet, SHEET.XLS, in the C:\EXCEL\WRKSHEET directory:

```
ctrDDELink.LinkTopic = "Excel|C:\EXCEL\WRKSHEET\SHEET.XLS"
ctrDDELink.LinkMode = 2    'Manual
```

All DDE server applications are required to have a Topic name "System" that returns information about the server, including the names of valid topics. Testing the values returned by the System topic enable you to determine if the file that constitutes the topic is presently open in the server application.

If setting the value of the LinkMode property successfully opens a DDE channel, Visual Basic assignes the channel number to the control object. If the link fails, you receive an error message. The preceding code fragment returns the entire content of the worksheet to ctrDDELink.

An Example TEZDEMO for DDE

There is an example about DDE, at the back of this book. TEZDEMO includes the principal DDE techniques. This thesis gives a route with which reader can develope more complex applications. The comments in the program code, are enough to understand it easily.

BÖLÜM 1

GİRİŞ

Dinamik veri değiş tokuşunu (DDE) incelemek için, Microsoft Windows 3.1 (Türkçe) kullanıldı. DDE'nin açıklanabilmesinde Windows 3.1 üzerinde çalışabilen uygulamalara ihtiyaç vardır. Böyle uygulamalar üretmek için Microsoft'un Windows üzerinde çalışan program geliştirme ortamlarından yararlanıldı.

Dinamik veri değiş tokuşu, Windows işletim sistemi üzerinde çalışan uygulamaların haberleşmesini sağlamak amacıyla işletim sisteminin içinde verilmiş bir protokoldur. Windows'ta programların herbiri birer pencere içinde çalışırlar. Pencereler arasında İstemci/sunucu ilişkisini sağlayan bağlantı kurulması DDE iletileri kullanılarak yapılabilir.

DDE'yi açıklarken konuya ait terminolojinin bilinmesi gerekir. 2. bölümde önemli bulunan terimler seçilip açıklandı. Daha sonra 3. bölümde; pencereler içinde çalışan uygulamaların protokola ait iletileri kullanarak nasıl ve ne tür DDE oturumları başlattıkları anlatıldı. Ayrıca bölümün sonunda DDE iletileri listesi verildi.

Windows 3.1 üzerinde çalışan programlar üretmek için MS Visual Basic for Windows 3.0 ve MS Excel for Windows 5.0 kullanıldı. Visual Basic, genel uygulamalar geliştirmek için tasarlanmış bir dildir. Ekran tasarımı yaparken, kullanılan nesnelerin özelliklerinin iyi bilinmesi gerekir. Ekranları bir projenin parçaları gibi düşünüp gruplayarak, tek bir program meydana

getirilebilir. 4. bölümde; Visual Basic'in bu tez içindeki kullanımını anlayabilecek kadar bilgi verildi. Burada "olay-tetikli programlama" denince ne anlaşıldığı ve önemli bazı Visual Basic fonksiyonları örnekli olarak açıklandı.

Excel, Windows üzerinde çalışan bir elektronik tablodur. Çalışmalar iş sayfalarında yapılır. İş sayfasındaki verilerle bir grafik gösterim yapılabilir. Grafik, verilerle aynı yerdeyse "katılmış grafik", ayrı bir sayfada ise "grafik sayfası" hazırlanır. Veriler grafikle ilişki halindedir. 4. bölümde Excel Visual Basic'ten sonra anlatıldı.

Visual Basic programları arasında DDE kurulmasını, Visual Basic ve Excel arasında ya da Excel ile Visual Basic programları arasında DDE kurulmasını sağlayan yöntemler de işlem basamakları halinde 4. bölümün son konularını oluşturdu.

DDE'nin uygulamalarla gerçeklenmesi 5. bölüm içinde bir örnek program ile açıklandı. Burada, DDE'nin gerekliliği ve niçin Visual Basic ve Excel kullanıldığı belirtildi.

BÖLÜM 2

2. DİNAMİK VERİ DEĞİŞ - TOKUŞUYLA İLGİLİ KAVRAMLAR

DDE'nin anlaşılması için anahtar görevi gören kavramlar ve terminoloji vardır. Bunlardan en önemli olanları aşağıda açıklanmıştır.

2.1. İstemci, Sunucu, Oturum (Client, Server, Conversation)

Bir DDE oturumunda, iki uygulama dinamik veri değiş - tokuşuyla ilgili olur. Oturumu başlatan uygulama "istemci", istemci'ye cevap veren uygulama ise "sunucu" dur. Uygulama birden fazla oturumla aynı zamanda ilgili olabilir. Böyleyse, bazı uygulamalar için istemci, bazıları için de sunucu konumunda çalışır.

DDE oturumu iki pencere arasında olur. Pencerelerde uygulamalar vardır. Pencere; uygulamanın ana penceresi, bir belgenin penceresi, birden fazla uygulamayla arayüz oluşturan belgenin penceresi, amacı DDE iletilerini işlemek olan gizli (görünmeyen) bir pencere olabilir. Bir DDE oturumunda pencere çiftleri var ve bunlardan herhangi biri başka bir DDE oturumuyla da ilgilenemez. İstemci ve sunucu olan bu pencereler, ilgilenmek istedikleri her DDE oturumu için (bu oturumlardaki görevleri bazen istemci bazen sunucudur) farklı bir pencere oluşturmak zorundadırlar.

Uygulama istemci / sunucu çifti içinde yeralırken, diğer oturumlarda gizli pencere oluşturarak kullanılamaz.[1]

2.2. Uygulama, Konu, Parça (Application, Topic, İtem)

DDE'de istemci ve sunucu arasında kullanılan üç seviyeli veri geçirme birimleri tanımlanır; uygulama, konu, parça.

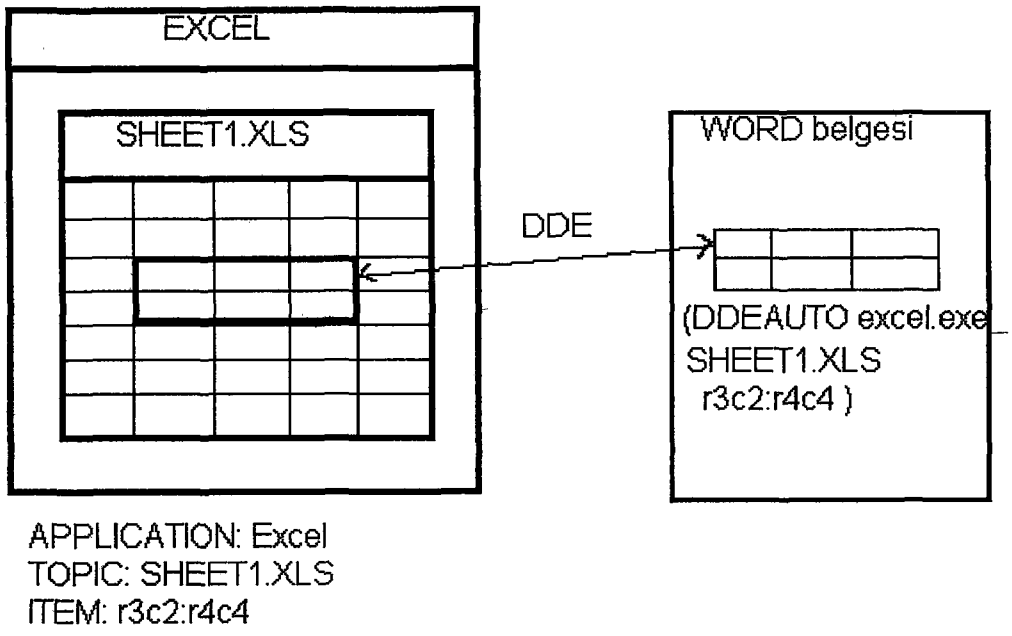
Her DDE oturumu, biricik olan uygulama ve konu ile belirlidir. Oturumun başlangıcında, istemci ve sunucu tarafından uygulama ismi ve konu saptanır.Uygulama ismi olarak genellikle sunucu uygulamanın adı kullanılır.Örneğin, bir uygulamada Microsoft Excel sunucu ise uygulama adı Excel'dir.

Konu; DDE oturumu sırasında değiş-tokuş edilecek verinin genel bir sınıfını belirtir.Dosya tabanlı belgelerle çalışan uygulamalar için, konu dosya adıdır. Diğerleri için uygulamaya özel bir isimdir.

İstemci ve sunucu DDE oturumunu belirten uygulama adı ve konu üzerinde anlaşmaya varırlar ve bunlar bir oturum zamanında aynı kalırlar, değiştirilemezler.

DDE veri parçası, konuyla ilgili bir bilgidir. Veri parçasının değeri istemci ve sunucu arasında hareket ettirilebilir. Veriler bir clipboard formatıyla aktarılır. Özellikle, "Link" isimli kayıtlı bir format, DDE oturumundaki parçayı tanıtmak için kullanılır.

Şekil 2.1 Microsoft Excel çalışma sayfasıyla, Microsoft Word belgesi arasında, DDE varken kurulan bağlantıyı gösterir. Bu örnekte uygulama adı "Excel", konu adı "çalışma sayfası" dır (SHEET1.XLS). Word belgesi tarafından istenen veri parçaları "r3c2:r4c4" hücreleridir.[1]



Şekil 2.1 Excel çalışma sayfası ve Word belgesi arasında kurulmuş DDE bağlantısı.

2.3. Veri bağlantıları hakkında ön bilgi

DDE oturumu bir kere başladıktan sonra, istemci sunucuyla bir ya da daha fazla sürekli bağlantı (link) kurabilir. Veri bağlantısı; veri parçalarında olan değişiklikleri, sunucunun istemciye bildirmesini sağlayan bir haberleşme mekanizmasıdır. Veri bağlantısının süresi, DDE oturumu bitene kadardır.

İki çeşit sürekli DDE bağlantısı vardır: "ılık" ve "sıcak". Bir ılık bağlantıda; sunucu, veri parçası değiştikçe istemciyi uyarır, fakat istemci istemedikçe sunucu değişen veriyi göndermez. Sıcak bağlantıda; sunucu değişen veri parçasını ivedilikle istemciye gönderir.

Windows'un "Edit" menüsünde "Copy" veya "Paste Link" komutlarıyla, kullanıcı uygulamalar arasında ılık ya da sıcak bağlantılar kurabilir. Bu uygulamalar standart clipboard formatı olan "Link" formatını

destekler. "Copy" ve "Paste Link" komutları bir uygulamayla kullanılırken, bu clipboard formatının kullanıcıya verdiği imkanla, kullanıcı sunucuya karşılık gelen uygulamadan veri parçasını kopyalar ve sonra istemciye bu veri parçasını geçirtir.

Kullanıcı "Edit" menüsündeki "Copy" komutunu seçince; sunucu uygulaması "Clipboard"a "Link" clipboard formatında bir ileti gönderir. Standart "Link" formatı şöyledir: "application\0topic\0item\0\0".

Tek "null" karakter isimleri birbirinden ayırır ve ifade iki "null" karakterle biter. İstemci ve sunucu uygulamaların her ikisinin de standart link formatını desteklemeleri gerekir.

İstemci uygulamasının Link clipboard formatını desteklemesi; "Edit" menüsünde "Paste Link" komutunu kullanmasıyla olur. Bu komut kullanıcı tarafından seçildiğinde, istemci yukarıda anlatılan formatta gelen iletiyi ayrıştırır. Buradaki isimleri kullanarak belirtilen uygulama ve konu için (daha önceden böyle bir oturum açılmamışsa) DDE oturumunu başlatır. İstemci uygulaması sonra WM_DDE_ADVISE iletisini sunucuya gönderir. Bu iletinin içinde, Link formatlı Clipboard verisine ait parça ismi vardır.[3]

BÖLÜM 3

3. WİNDOWS 3.1'DE DDE YAPISI

DDE; Windows'da prosesler arası haberleşmede desteklenen üç mekanizmadan biridir. Diğer iki mekanizma, Windows Clipboard ve dinamik bağlantı kütüphaneleriyle (DLL) paylaşımlı bellek kullanılmasıdır.

DDE, Windows'un içinde bulunan bir mesajlaşma sistemidir. İki Windows programı bir DDE oturumunda birbirlerine iletiler postalayarak çalışırlar. Bu programlara istemci ve sunucu dendiği söylenmişti. DDE sunucusu asıl olarak, diğer Windows programının yararlanabileceği verilere erişen programdır. DDE istemcisi ise sunucudan verileri elde eden programdır.

Windows 3.1'de programlar seçimlik olarak DDEML'yi (DDE Management Library) kullanırlar. DDEML, DDE'yi kolaylaştırır. Bu kütüphane, yüksek seviyeli fonksiyon çağrı katmanıyla sağlanan DDE mesajlaşmasını kullanan bir programın içine sokulabilir (insulate). DDEML içeren programlar DDE iletilerine call-back fonksiyonlarla cevap verirler. Klasik DDE ve DDEML'li yöntemler arasında uyum problemi yoktur. Çünkü, DDEML daha alt seviyede kalan DDE mesajlaşmasının üzerine kurulmuştur ve tek amacı, programcıyı DDE'nin karmaşıklığından biraz olsun yalıtmaktır.

Bu bölümde daha çok klasik DDE ve biraz DDEML üzerinde durulacaktır.

DDE oturumunun istemci program tarafından başlatıldığı anlatılmıştı. İstemci, aktif olarak çalışan Windows programlarının hepsine bir ileti yayımlar (WM_DDE_INITIATE). Bu ileti, istemcinin ihtiyacı olan verilerin genel bir sınıfını belirtir. DDE sunucusu, yayımlanan iletiye cevap olabilecek uygun verileri elinde tutar (veya bunlara ulaşabilir). Oturum böylece başlar.

Windows programı, bir programa göre istemci diğerine göre sunucu olarak davranabilir. Böyle olursa iki farklı oturumun kurulması gerekir. Sunucu uygulama, birden fazla istemciye veri gönderebilir ya da bir istemci birden fazla sunucudan veri alabilir. O zaman birden fazla DDE oturumlarının kurulmaları gerekir. Bu oturumların biricik (unique) ve ayrık olmalarını sağlamak için istemci/sunucu üzerindeki her oturum farklı bir pencere kullanır. Genellikle, DDE'yi destekleyen her program, üzerinde DDE oturumunun kurulacağı bir çocuk pencere yaratır.

DDE oturumunda çalışacak programların kodlarının özel bir biçimde yazılması gerekir. DDE sunucusunun programcısı, verinin şeklini bildirir. DDE istemci olarak çalışan programın (Microsoft Excel gibi) kullanıcısı bu bilgiyle iki program arasında DDE oturumunu kurabilir.

Sadece biribirleriyle haberleşebilen Windows programları ailesi oluşturmak istenebilir. O zaman, özgün mesajlaşma protokolünün tanımlanması gerekir. Bunun yapılması tavsiye edilmez; çünkü Windows, gelecekteki sürümlerinde, proses haberleşmesi için DDE'den farklı mesaj tabanlı uygulamaları desteklemeyecek.

Windows 3.1'deki mesajlaşma sistemi kendi içinde olduğundan, DDE tarafından kullanılması doğaldır. Fakat bu, DDE'nin gerçeklenmesinin kolay olduğu anlamına gelmez. Bir protokolde birçok seçimlik (opsiyon) vardır ve programlarda birçok problemin çözümlenmesi gerekir.[2]

3.1. Önemli Kavramlar

İstemci sunucudan veri sorarken, ne tip veri istediğini belirtmek zorundadır. Bunu yapmak için şu üç karakter katarı kullanılır; "application" (uygulama), "topic" (konu), "item" (veri parçası).

Uygulama, konu, parça kavramlarını daha iyi açıklamak için bir örnek kullanılmalı. Bunun için, DDEPOP1 isminde bir sunucu programının nasıl yazıldığı açıklanacak. Bu program Amerika'nın 1970, 1980, 1990 nüfus verilerini kullanır, "quadratic extrapolation" ile herhangi bir eyaletin ya da tüm Amerika'nın anlık nüfusunu hesaplayabilir.

DDE sunucu programını yazan kişi uygulama, konu ve parçanın nasıl tanımlandıklarını belirtmek zorundadır:

a) Sunucu uygulama ismi: Bu örnek için "DDEPOP1". Her sunucu bir uygulama ismine ve bir program ismine sahip olmalıdır.

b) Konu ismi: DDE sunucuları en az bir konuyu destekler. DDEPOP1'de "US_Population" katarıyla belirtilen sadece bir konu vardır. Düşünce olarak; DDEPOP1 programına eyaletlerin yüz ölçümleri ($mile^2$) dahil edilirse "US_Area" adlı ikinci bir konu kullanılabilirdi.

c) Parça ismi: Her konunun içinde bir ya da daha fazla parça vardır. DDEPOP1'de eyaletlerin iki harfli posta kodu kısaltmaları parça olarak kullanılır. Örneğin, New York için "NY", California için "CA" olur. 52 parça vardır; 50 eyalet, DC (District of Columbia) ve tüm ülke.

Yukarıda verilenlerle; DDEPOP1, istemci olan başka bir Windows programıyla (MS Excel'de) sunucu olarak kullanılabilir.MS Excel ile DDEPOP1'i kullanmak için bir Excel sayfası hücresine şunu yazmak gerekir:

=DDEPOP1|US_Population!US

Bu üç katar uygulama, konu ve parçayı belirtir (yukarıdaki örnek tüm Amerika nüfusu içindir). DDEPOP1.EXE çalışır durumda değilse MS Excel onu çalıştırmaya uğraşır (DDEPOP1 aynı dizinde olmalı ya da PATH ortam değişkenindeki listenin içinde bulunmalıdır). Bu çaba başarılı olursa Excel, DDEPOP1'le bir DDE oturumu başlatır. Nüfus verilerini elde eder ve nüfusu bir hücrenin içinde sayı olarak gösterir. Böyle bulunan nüfus değerleriyle grafikler çizilebilir, başka işlemler yapılabilir.

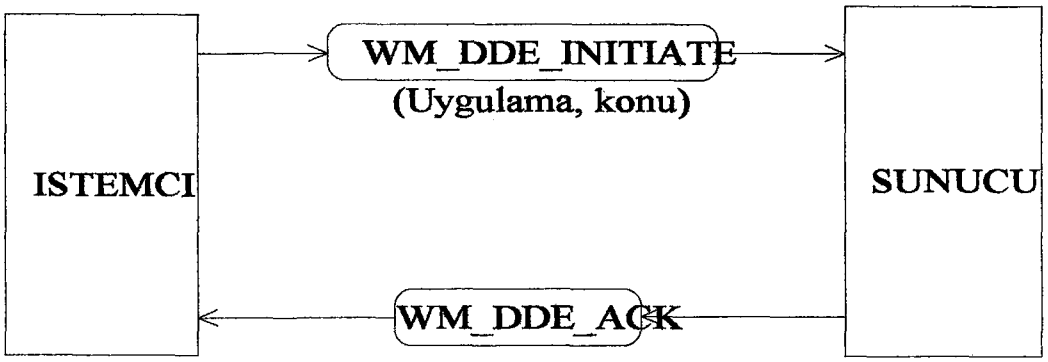
Nüfus sonuçları, periyodik olarak hücrelerde güncellenirler. Bu duruma "sıcak bağlantı" (hot link) ya da küçük bir farkla "ılık bağlantı" (warm link) denir. Beş saniyede bir DDEPOP1 sunucusu nüfus verilerini değerlendirip yeni bir sonuç üretir ve istemciyi uyarır. Örneğin, Amerika'nın nüfusu her 15 saniyede bir kişi artar.[3]

3.2. DDE Oturum Çeşitleri

Temel olarak üç çeşit DDE oturumu vardır; soğuk bağlantı, sıcak bağlantı, ılık bağlantı. Bu oturumlar, DDE.H başlık dosyasında tanımlanan DDE iletilerini kullanırlar.

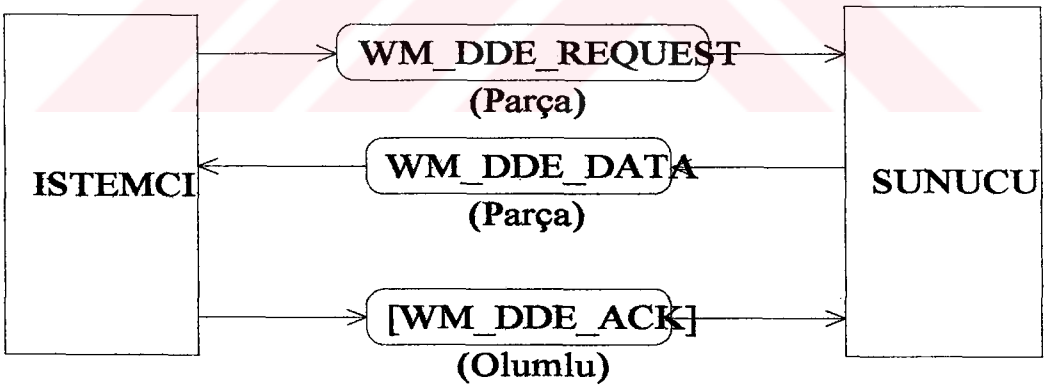
3.2.1. Soğuk Bağlantı

İstemci, içinde uygulama, konu ismi bulunan bir WM_DDE_INITIATE iletisi yayımlayarak soğuk bağlantı oturumunu başlatır. Burada uygulama ve konu "null" olursa, bir sunucu uygulamasıyla ya da herhangi bir konuyla oturum başlar. Sunucu, belirtilen konuda WM_DDE_ACK iletisiyle istemciyi cevaplar. Şekil 3.1.



Şekil 3.1

İstemci, WM_DDE_REQUEST iletisiyle özel veri parçası isteğinde bulunur. Sunucu, bu veri parçasını sağlayabilirse WM_DDE_DATA iletisiyle istemciye cevap verir (burada köşeli parantez içinde yazılan iletiler seçimliktir). Şekil 3.2.

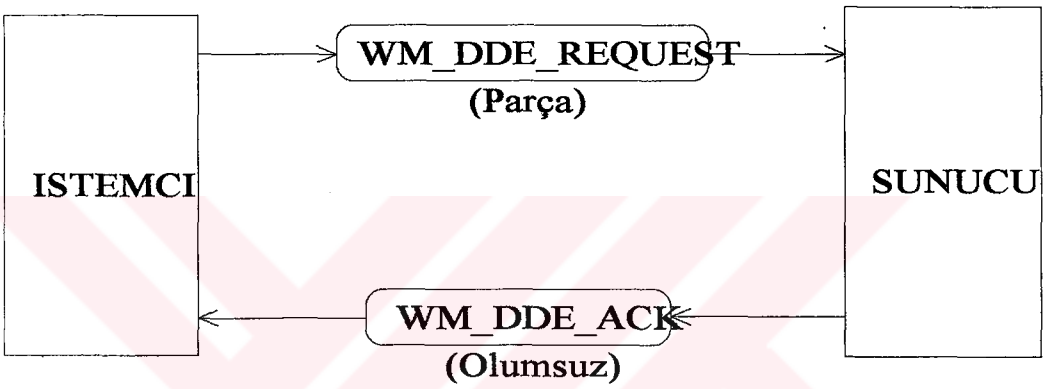


Şekil 3.2

Bu bağlantı türünde; istemci ne zaman birşeyler sorarsa o zaman birşeyler öğreniyor. İstemci, kendisine gelen WM_DDE_DATA iletisiyle veri parçasını aldığını sunucuya bildirebilir. Bunun yapılması seçimliktir; sunucu böyle bir mesaj isteyip istemediğini, kendi gönderdiği WM_DDE_DATA iletisinin içinde bir bayrağı geçirerek (bayrağı çeker ya da çekmez) belirtir.

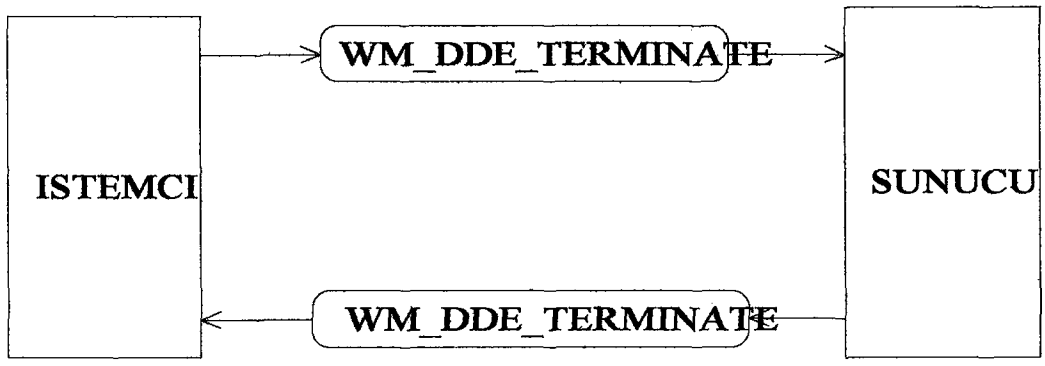
Eğer istenmişse, istemci WM_DDE_ACK iletisini (burada da bir bayrak var) gönderir, bu "olumlu" (positive) bir cevaptır.

İstemcinin WM_DDE_REQUEST iletisinde istenen veri parçası sunucu tarafından sağlanamıyorsa, o zaman sunucu "olumsuz" (negative) WM_DDE_ACK iletisini istemciye gönderir. Şekil 3.3.



Şekil 3.3

DDE oturumu istemcinin WM_DDE_REQUEST iletileri göndermesi ve bunları sunucunun WM_DDE_DATA veya WM_DDE_ACK iletileriyle cevaplamasıyla sürer. İstemci ve sunucu birbirlerine WM_DDE_TERMINATE iletileri yolladığında DDE oturumu biter.WM_DDE_TERMINATE iletisi ilk önce sunucudan de gelebilir. Şekil 3.4.

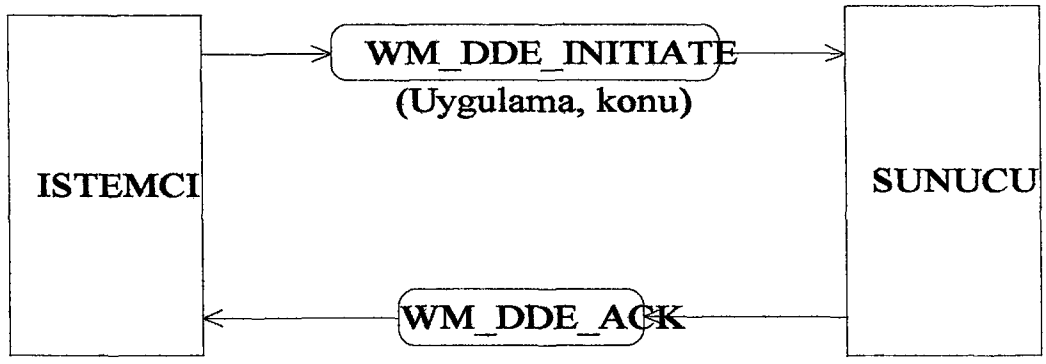


Şekil 3.4

3.2.2. Sıcak Bağlantı

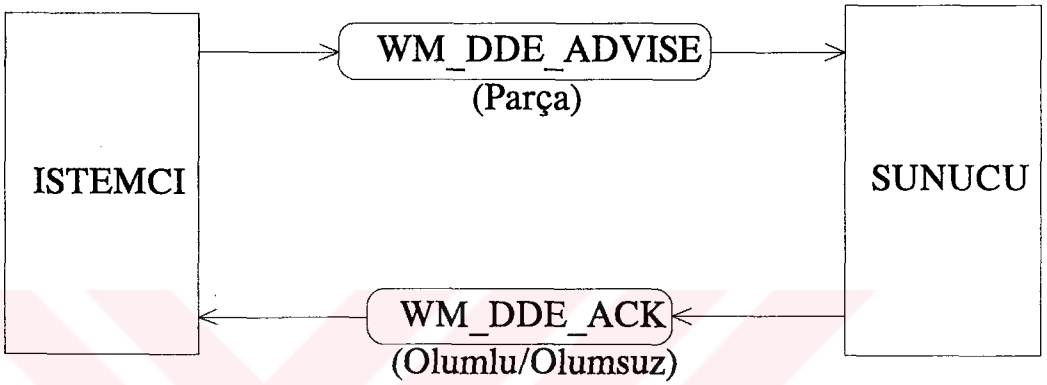
Soğuk bağlantıdaki sorun; veri, aktarılma süresi içinde güncelliğini kaybedebilir. Örneğin, DDEPOP1 anlık olarak hesapladığı sonucu gönderene kadar nüfus değişebilir. Soğuk bağlantıda, istemci verilerin ne zaman değiştiğini bilmez. Sıcak bağlantı bu problemi çözer.

DDE oturumu yine, WM_DDE_INITIATE ve WM_DDE_ACK iletileriyle başlar. Şekil 3.5.



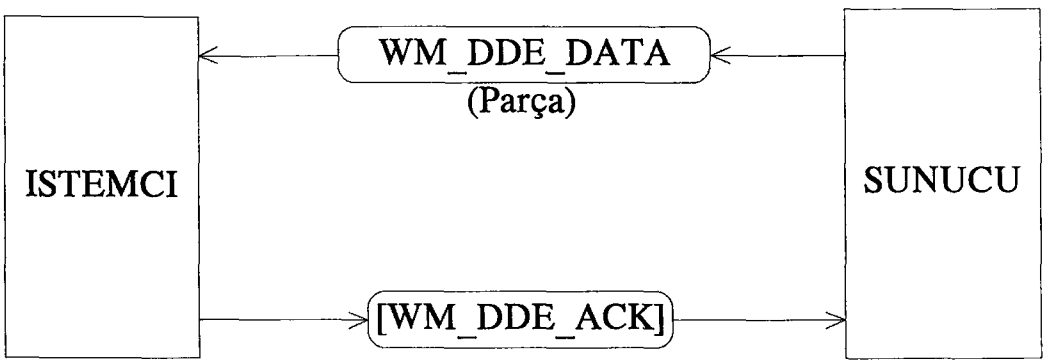
Şekil 3.5

Oturum başladıktan sonra istemci, çalışmak istediği veri parçasını WM_DDE_ADVISE iletisiyle sunucuya gönderir. Sunucu, istemciye WM_DDE_ACK iletisini gönderir. Bu ileti olumluysa sunucu istenen veri parçasını sağlayabiliyor, değilse sağlayamıyor demektir. Şekil 3.6.



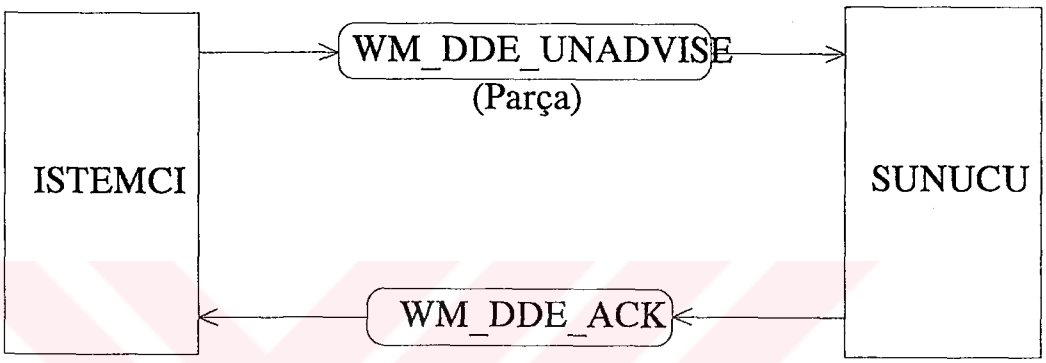
Şekil 3.6

Bu aşamada sunucu, veri parçasının değeri değiştiği zaman istemciyi uyarmakla yükümlüdür. Uyarmak için WM_DDE_DATA iletisini kullanır (iletinin içinde bir bayrak aktarılır, buna göre) istemci WM_DDE_ACK iletisini gerekirse yollar. Şekil 3.7.



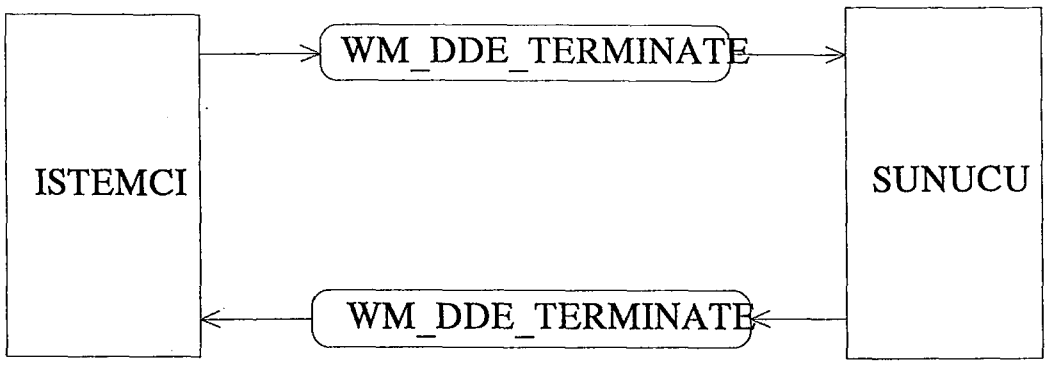
Şekil 3.7

Verinin güncellenmesi artık istemciyi ilgilendirmiyorsa, sunucuya WM_DDE_UNADVISE mesajı yollanır. Sonra sunucu, alındı mesajını gönderir. Şekil 3.8.



Şekil 3.8

Oturum karşılıklı WM_DDE_TERMINATE iletileriyle bitirilir. Şekil 3.9.

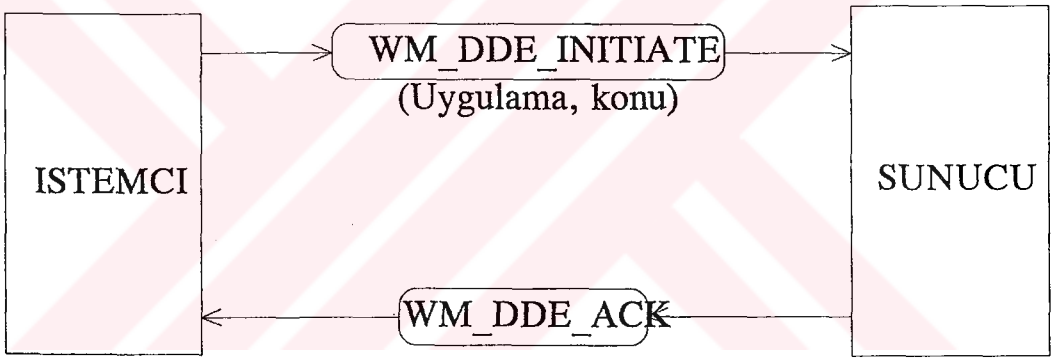


Şekil 3.9

Soğuk ve sıcak bağlantı birbirine benzer. Bir tek DDE oturumunda istemci veri parçası istemede, WM_DDE_REQUEST (soğuk bağlantı) veya WM_DDE_ADVISE (sıcak bağlantı) iletisini kullanır.[3]

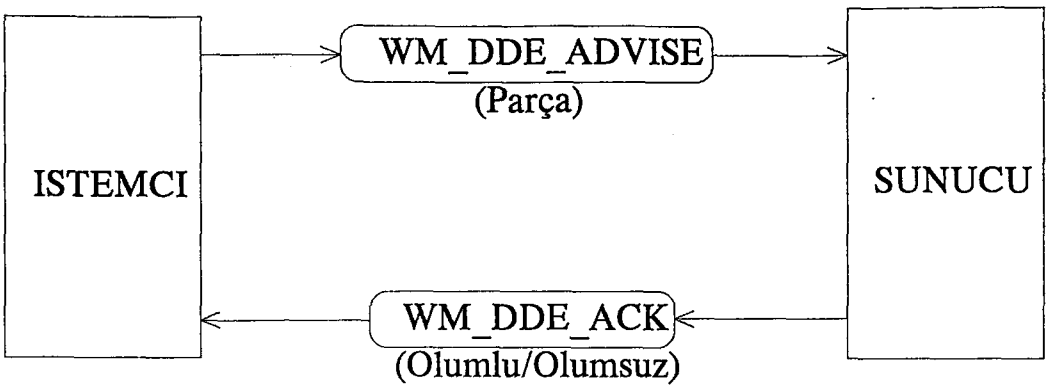
3.2.3. Ilık Bağlantı

Ilık bağlantı soğuk ve sıcak bağlantının karışımıdır. DDE oturumu her zamanki gibi başlar. Şekil 3.10.



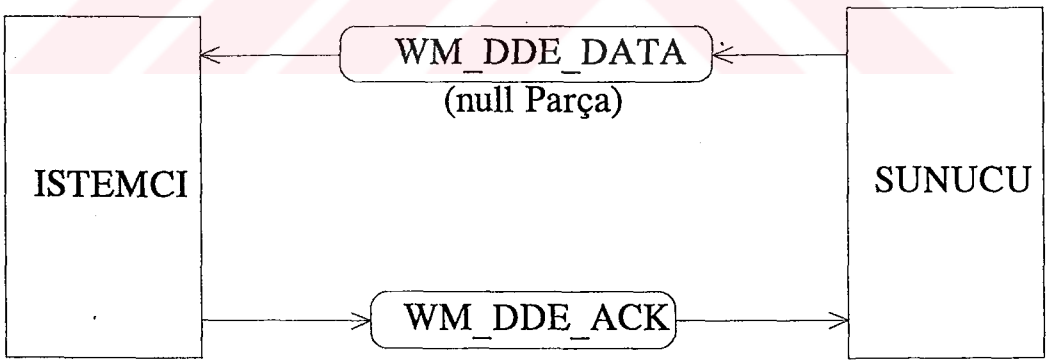
Şekil 3.10

Sıcak bağlantıda olduğu gibi, istemci sunucuya WM_DDE_ADVISE iletisi gönderir. Sunucu olumlu/olumsuz bir alındı iletisi yollar. Şekil 3.11.



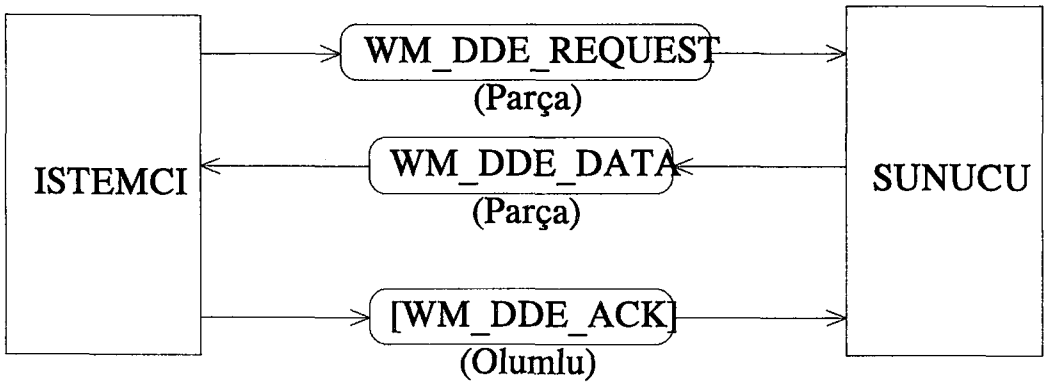
Şekil 3.11

WM_DDE_ADVISE iletisi içindeki bir bayrakla istemci, veri üzerindeki değişikliği (güncel veriyi hemen almadan) sadece bilmek istediğini belirtebilir. O zaman, değişiklik olduğunda, sunucu boş bir WM_DDE_DATA iletisi gönderir. Şekil 3.12.



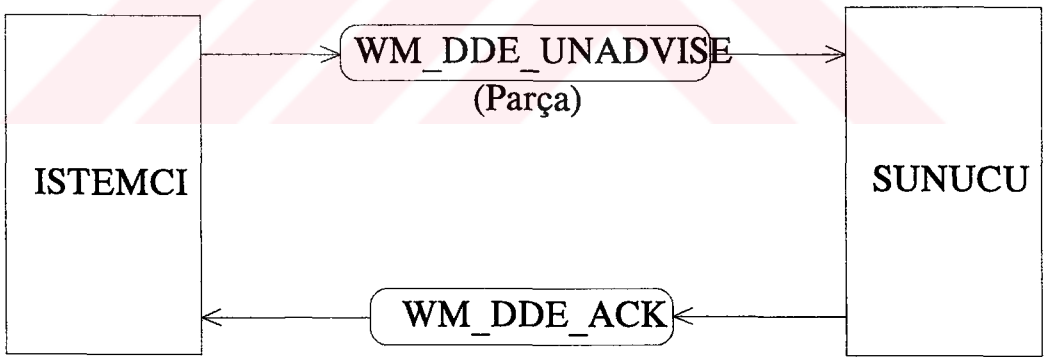
Şekil 3.12

Şimdi istemci kendi ilgilendiği veri parçasının değiştiğini bilir. Bu veri parçasını elde etmek için WM_DDE_REQUEST iletisini soğuk bağlantıdaki gibi kullanır. Şekil 3.13.



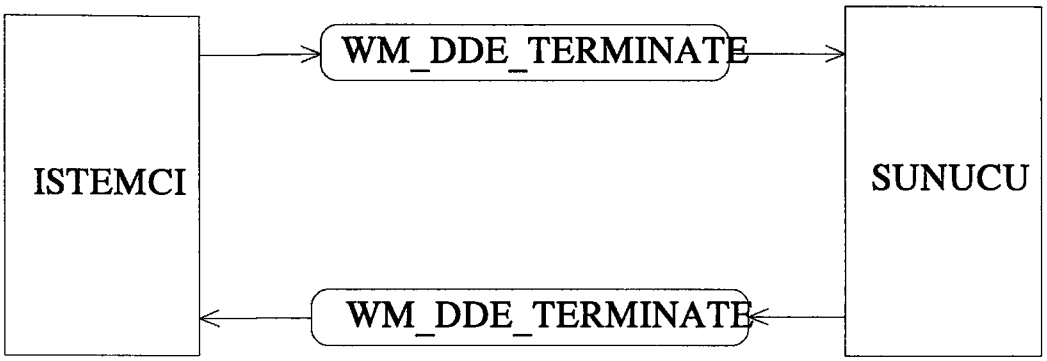
Şekil 3.13

İstemci artık verinin güncellenmesiyle ilgilenmiyorsa, bunu sunucuya sıcak bağlantıdaki gibi bildirebilir. Şekil 3.14.



Şekil 3.14

DDE oturumu karşılıklı WM_DDE_TERMINATE iletileriyle sona erdirilir. Şekil 3.15.



Şekil 3.15

Yukarıda anlatılan üç tür oturumda şu iletiler kullanılmadı: WM_DDE_POKE (istemci, sunucuya doğrudan veri gönderir), WM_DDE_EXECUTIVE (istemci, sunucuya yürütmesi için doğrudan bir komut katarı gönderir).

3.3. Karakter Katarları ve Atomlar

Şimdiye kadar DDE oturumunda istemci ve sunucu arasında verilerin nasıl gösterildiği (uygulama, konu, parça) anlatıldı. İstemci - Sunucu arasındaki veri iletiminde asıl rol oynayan "Atom" denen karakter katarlarıdır. Atomların anlatılması daha alt düzeyli bir konudur.

Atomlar "WORD" tipinde değerlerdir ve karakter katarlarına işaretçidirler. Bir programın atomları ona ait data segmentindedir ve sadece onun tarafından kullanılır. Bunlar bir tabloda (Atom tablosu) yer alırlar. Atom şöyle tanımlanabilir:

```
ATOM   aAtom ;
```

Şu fonksiyonla atom tablosuna bir katar eklenebilir:

aAtom = AddAtom (lpString) ;

Karakter katarı atom tablosunda değilse bu fonksiyon onu tabloya ekler ve biricik bir değer döndürür. Her atomun başvuru sayısı (reference count) vardır. AddAtom fonksiyonuyla aynı katarın çağrılma sayısıdır. Başvuru sayısı başlangıçta 1 olur. AddAtom aynı katar için her kullanılışında katarı belirten bir sayı döndürür ve başvuru sayısını bir arttırır. Başka bir fonksiyon:

DeleteAtom (aAtom) ;

Başvuru sayısını bir azaltır. Bu sayı sıfır olunca, atom ve karakter katarı atom tablosundan çıkarılır. Bir atomun işaret ettiği katarı bulabilmek için:

aAtom = FindAtom (lpString) ;

Fonksiyonu kullanılır. Bu katar, atom tablosunda yoksa sıfır döndürür. Başvuru sayısını değiştirmez. Atoma ait katarı elde etmek için:

nBytes = GetAtomName (aAtom, lpBuffer, nBuffersize) ;

Fonksiyonu kullanılır. Son parametre, ikinci parametreyle işaret edilen tamponunun boyunu gösterir. Bu fonksiyon başvuru sayısını etkilemez ve tamponda kaç byte alanın dolu olduğunu gösteren bir sayı döndürür.

Yukarıda anlatılan fonksiyonlar bir programın kendi atomlarının kullanılmasıyla ilgilidir. Atom tablosu programın veri segmentindeyse, o program için biriciktir (unique). Atomları DDE'de kullanmak için ek fonksiyonlar gerekir. Bunlar yukarıdakilere benzer olarak:

aAtom = GlobalAddAtom (lpString) ;
GlobalDeleteAtom (aAtom) ;
aAtom = GlobalFindAtom (lpString) ;
nBytes = GlobalGetAtomName (aAtom, lpBuffer, nBuffersize) ;

Windows'un içinde DLL (dynamic link library) ile, paylaşılan veri segmentine konulan atom tablosundaki atomlar, bu fonksiyonlar yardımıyla kullanılabilir. Bir program GlobalAddAtom ile atom tablosuna bir katar eklemek ve bu atomu başka bir programa geçirmek işini yapabilir. Karşısındaki program ise GlobalGetAtomName fonksiyonuyla o atoma ait karakter katarını elde edebilir. DDE'de atomların kullanılması önemlidir. Örneğin, Başka program tarafından atom tablosundan silinmiş bir atomun, bir program tarafıdan sürekli istenmesi olayı olmamalı. Benzer olarak, artık gerekmeyen atomların atom tablosunda silinmeden bırakılması da istenmez.

Atomlar, önceki konuda anlatılan DDE uygulama, konu ve veri parçası için kullanılır. Veriler (daha çok kalıplanmış veriler) bir Windows programından diğerine transfer edilirken bellekte bunlara bir yer ayrılabilmelidir. Bunun için GlobalAlloc GMEM_DDESHARE seçimliğiyle kullanılır. Böylelikle global bir bellek bloğu birden fazla Windows programı arasında paylaşılabilir. DDE kurallarıyla bu global bellek bloklarının tahsis işleminin hangi program tarafından yapılacağı belirlidir.[2]

3.4. DDE İletileri Listesi

DDE oturumları istemci ve sunucu pencereleri arasında belirli DDE iletilerini geçirerek çalışırlar. Dokuz tane DDE iletisi vardır.

Bu iletiler aşağıdaki gibi özetlenebilir. Detaylı tanımları için "Microsoft Windows Programmer's Reference, Volume 3" e bakılabilir.

WM_DDE_ACK: Bir iletiyi alıp almadığını bildirir.
WM_DDE_ADVISE: Bir veri parçasında değişiklik olduğunda, bunun sunucu tarafından istemciye bildirilmesini sağlatan iletidir. Sürekli veri bağlantısı kurar.
WM_DDE_DATA: İstemciye bir veri parçasını götürür.

WM_DDE_EXECUTE: Sunucu için, bir seri komutlar anlamına gelen karakter katarı taşır.
WM_DDE_INITIATE: İstemci ve sunucu arasında oturum başlatır.
WM_DDE_POKE: Sunucuya bir veri parçasını götürür.
WM_DDE_REQUEST: Sunucudan veri parçası isteğinde bulunur.
WM_DDE_TERMINATE: Oturumu bitirir.
WM_DDE_UNADVISE: Sürekli veri bağlantısından vazgeçildiğini bildirir.

Uygulama, WM_DDE_INITIATE iletisi için veya buna yanıt olan WM_DDE_ACK iletisi için "Windows SendMessage" fonksiyon çağrısını yapar. Diğer tüm iletiler "Windows PostMessage" fonksiyon çağrısıyla gönderilir. Bu çağrılarda, alan pencerenin pencere ilkeli (handle) ilk parametre olarak görünür. İkinci parametre; gönderilecek iletiyi, üçüncüsü; gönderen pencereyi belirtir. Dördüncü parametre; iletiye özel argümanları bulundurur.[2]

3.5. DDE İleti Akış Özeti

Tipik bir DDE oturumunda aşağıdaki olaylar olur:

1. İstemci uygulama oturumu başlatır ve sunucu uygulama yanıtlar.

2. İstemci, sunucuya doğrudan veri gönderebilir.

3. İstemci uygulama, veri değişikliğinde güncel veriyi göndermesi için sunucudan istekte bulunur (hot link).

4. İstemci uygulama, veri değişikliğinde uyarı göndermesi için sunucudan istekte bulunur (warm link).

5. Sunucu, istemcinin gönderdiği komutları anlar.

Bir uygulama penceresi istemci/sunucu'dan gelen iletileri, aldığı sırayla işlemelidir.

İstemci, birden fazla sunucuyla oturum kurabilir ya da tersi olabilir. Birden fazla kaynaktan gelen iletiler işlenirken, bir oturuma ait iletiler senkronize olarak (geliş sıralarına göre) işlenmeli ama diğer oturumlardan gelen iletilerin kendi aralarında senkron olarak işlenmeleri şart değildir.[3]

3.6. DDE Yönetim Kütüphanesi

DDE'yi kullanarak program yazmanın karmaşıklığı bilindiği için Microsoft yeni bir bakış açısı oluşturdu. DDE yönetim kütüphanesi (DDEML) DDE'nin karmaşıklığını gizlemek için yapıldı. Bu kütüphane, iletileri kılıflar, atom yönetimini ve bellek yönetimini bir fonksiyon çağrı arayüzünde yapar.

İlk anda DDEML zor görünebilir. DDEML.H başlık dosyasında 27 tane fonksiyon çağrısı "Dde" ön ekiyle tanımlanır. DDEML'yi kullanan program ayrıca bir call_back [ek1] fonksiyona ihtiyaç duyar. Bir call_back fonksiyon 16 değişik veri tipini kullanabilir. Bunlar, DDEML.H başlık dosyasında tanımlı "XTYP" ön ekiyle başlayan sabitlerdir.

DDEML bir DLL'dir. Windows altında çalışabilen uygulamalar bunu paylaşabilirler. DDEML ile birlikte, uygulama geliştirme arayüzü (API) vardır. Bununla, Windows tabanlı uygulamalara DDE yeteneği kazandırmak basitleşir. Bir uygulama; iletileri gönderme, işleme, alma işleriyle uğraşmak yerine, DDEML'nin DDE oturumlarını yönetmek için verdiği fonksiyonları kullanır. DDEML ayrıca DDE uygulamaları arasında paylaşılan verileri ve katarları yönetme özelliğini de sağlar. Paylaşılan bellek nesneleri için atomları ve işaretçileri kullanmak yerine, DDE uygulamaları; veri ilkelleri (global bellek nesnelerini belirtir), katar ilkelleri (katarları belirtir) yaratır, bunları değiş tokuş ettirir.

DDEML, sunucu uygulamanın belli servis isimlerine kayıt olmasına imkan verir. Bu isimler sistemdeki diğer uygulamalara duyurulur. Sistemdeki bir uygulama, servis ismini kullanarak sunucuyla bağlantı kurabilir. DDEML uygulamalara tutarlı bir DDE yapısı kurduttuğu için uyumluluk problemlerini de engellemiş olur. Klasik DDE veya DDEML ile çalışan uygulamalar arasında tam uyumluluk vardır.[3]

BÖLÜM 4

4. VISUAL BASIC VE EXCEL'DE DDE

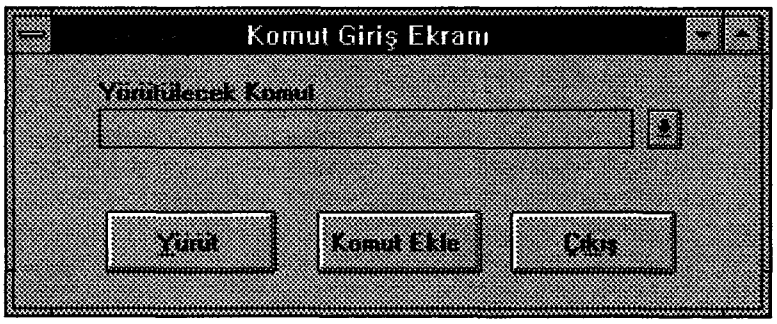
Bu bölümde, DDE olayının gerçeklenmesi için yapılması gerekenler anlatılacaktır. Uygulama geliştirme ortamları olarak; Visual Basic 3.0 for Windows, Excel 5.0 for Windows kullanıldı. Önce bu programlama ortamları anlatılacak.

4.1. Visual Basic

Visual Basic (VB) nesne yönelik olarak tasarlanmış olay-tetikli (event-driven) bir dildir. VB ile grafik ortamda çalışan uygulamalar geliştirilebilir. Uygulamalar genel anlamda bir proje gibi düşünülür (Şekil 4.1) ve birçok parçanın (ekranların) bir araya gelmesiyle oluşur. Proje dosyasının uzantısı .MAK olur. Proje içinde "Form" lar vardır.

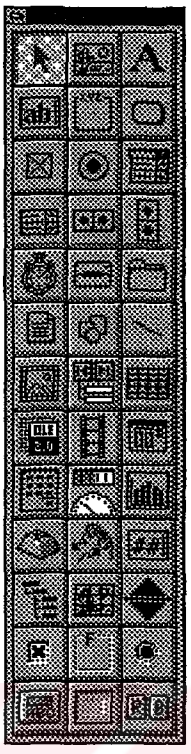
Şekil 4.1 Form ve kütüphane dosyaları bir proje içindedir.

Her form, içinde kontrolleri (kontrol amacıyla kullanılan nesneler), grafikleri ya da başka formları bulunduran bir penceredir (Şekil 4.2).



Şekil 4.2 İçinde bir kaç nesne bulunan Form'un yapısı.

Formlar geliştirilen uygulamanın arayüzleridir. Form penceresindeki nesneler tasarım sırasında (design-time) "Toolbox" denen (Şekil 4.3) yardımcı pencereden seçilirler. VB'nin esnek yapısı sayesinde çalışma sırasında da (run-time) bazı kontroller yaratılabilir.



Şekil 4.3 Toolbox

Formlar birçok şekilde kullanılabilir; uygulama başlangıcını gösteren bir ekran penceresi, resim, hata durumlarında ortaya çıkan diyalog penceresi, uygulama içinde bir belge, vb. olabilir.

Bir kod parçası proje içinde, birden fazla form arasında kullanılacak ise onu "Module" içine koymak daha pratik olur. "Module" ; ekranda görünmeyen içinde sadece görevler yazılı olan bir form gibi düşünülebilir.

4.1.1. Visual Basic'te Nesneler

VB'de kullanılan formlar tasarlanırken programın desteklediği çeşitli nesneler kullanılır. Önce, nesne denilince ne anlaşılması gerektiği, bu nesnelerin neler olduğu anlatılacaktır. VB'de pencere içinde bulunan her eleman bir nesnedir. Bu elemanların hangi durumda nasıl davranacaklarını belirleyen her birine ait görev kodları vardır. Görev kodları programcı tarafından tasarım sırasında yazılır ve istenirse tüm kodları içeren bir döküman hazırlanabilir. Yaygın olarak kullanılan nesnelerden bazıları şunlardır:

Picture box

İçinde grafiksel imajları gösteren bir alandır.

Label

Kullanıcının değiştirmesi istenmeyen sabit yazı alanıdır. Ekranın içinde belli bir bölgeye isim vermek için kullanılabilir.

Text box

Kullanıcının içine yazı girebileceği veya var olan yazıyı değiştirebileceği nesnedir.

Frame

Nesneleri işlevlerine göre gruplamaya yarar. Bunu yaparken önce frame yerleştirilir sonra diğer nesneler buraya konulur.

Command button

Kullanıcının yaptığı seçimle çalışan ve bir dizi komut yürüten nesnedir.

Check box

Kullanıcı bu kutunun üzerine işaret koyarak birşeyi onayladığını belirtir. Kutular grup halindedir ve birkaç tanesi aynı zamanda seçilebilir.

Option button

Opsiyon butonları grup halindedir. "Check box" tan farklı olarak, bir anda sadece bir buton seçilebilir.

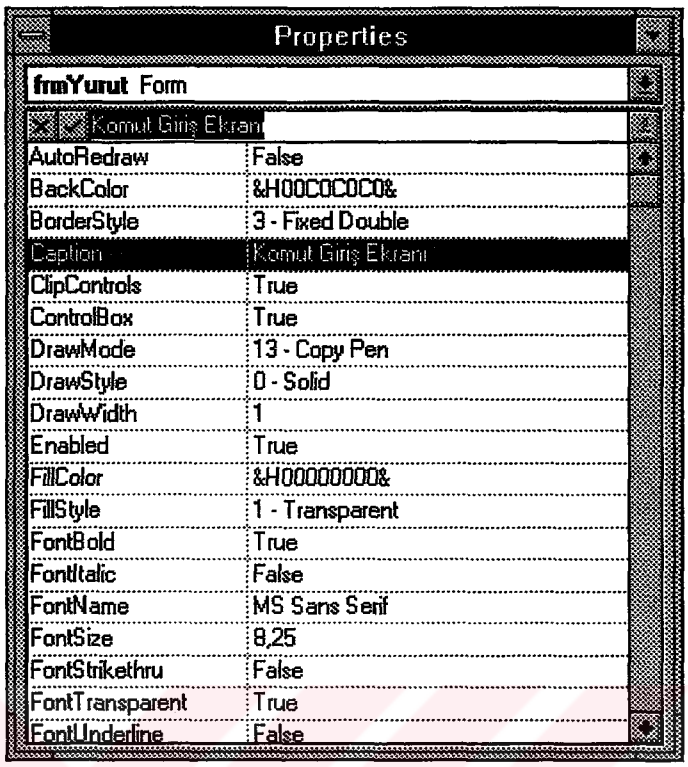
Combo box

Liste şeklinde bir kutu ve içine yazı girilebilen "Text box" nesnelerinin bitişik halidir. Kullanıcı isterse listeden bir eleman seçer ya da yeni bir eleman girer.

List box

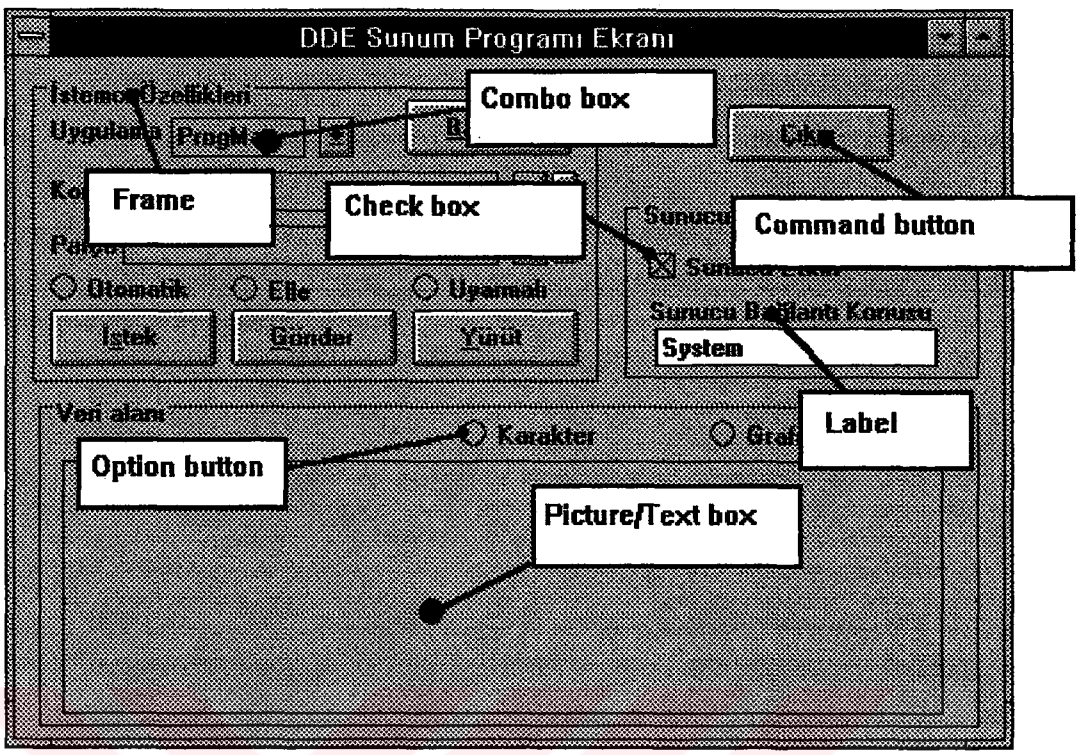
Kullanıcının içinden seçim yapabileceği elemanları liste şeklinde tutan bir kutudur.

Uygun nesneler seçilerek form doldurulur. Bu alanlar giriş/çıkış için kullanılır, formu daha anlaşılır yaparlar. Daha sonra her nesnenin kendisine ait olan "Properties" alanları düzenlenir (Şekil 4.4). "Form" un kendisi de bir nesnedir ve onun özelliklerini içeren "Properties" penceresi uygun şekilde doldurulmalıdır. Bu alanlarda nesnelerin şekil özelliklerini belirten bilgiler vardır. Nesne özellikleri değişken niteliktedir. Sonraki aşama, bu nesnelerin kod parçalarının onların görevlerine uygun olarak yazılmasıdır. Formların ve kontrolların kodları, sistem olaylarına cevap verecek şekilde yazılır. Kod yazılırken söz dizim kontrolu otomatik yapılmaktadır.



Şekil 4.4 Her nesne için bir özellikler penceresi vardır.

Uygulamalarda birçok komut girişi istenir. Her işlem komutu için kontrol koymak pratik değildir. Bunun yerine, komutları gruplayarak daha az yer kaplayan menüler tasarlanabilir. Menüler "Form" içinde yer alabilir. Yukarıda anlatılanlara örnek bir ekran aşağıdadır (Şekil 4.5).



Şekil 4.5 Bir form üzerinde nesneler

4.1.2. Olay-Tetikli Programlama

VB programlama ortamında olay-tetikli uygulamalar hazırlamak mümkündür. Klavyenin tuşuna basılması, farenin tıklatılması vb. olaylara karşı uygulama uyanık olur. Böylece görsel olarak anlaşılır, yönetimi kolay uygulamalar geliştirilebilir.

Geleneksel programlamada, kullanıcılar bilgi girerken programcının belirlediği sıraya uymak zorundadırlar. Olay-tetikli VB uygulamalarında, istenilen sırada bilgi girişi yapılır. Bir VB uygulaması formlar ve kontrollardan (herbiri bir nesnedir) oluşur. Her nesnenin (kontrolun) görevi, hareketleri anlayıp buna göre davranmaktır. Bu hareketlere "olay" (event) denir. Nesneyi yöneten "event procedure" (olay görevi) vardır. Olay görevinin adı; nesne adı

ve olay adından oluşur. Nesnenin adı, "properties" penceresinde "Name" alanı karşısında yazılıdır. Bir nesnenin kodu içinde, başka bir kontrol ile ilgilenmek gerekebilir. Örneğin; içine karakter katarları girilebilen bir "textbox" türü nesnede, bir yazı var mı, yok mu kontrolü yapılmak istensin. **if isimalanı.Text = " " then ....** yazılırsa, **isimalanı** nesnesinin **Text** özelliği (property) sınanmış olur. Olay görevi başka olay görevlerini de tetikleyebilir, altprogram çağrıları yapabilir, diğer nesnelerin özelliklerini değiştirebilir.

4.1.3. LinkMode Kullanımı

DDE oturumunun tipini belirler ve bağlatıyı kurar [Ek 2]. İstemci uygulamadaki bir kontrol DDE'yi başlatacaktır. Kontrolun "LinkTopic" ve "LinkItem" özellikleriyle oturum başlatılır. Sunucu; İstemciden gelen application|topic!item ifadesiyle tetiklenir. Kontrol nesnesinin türü; Label, Picturebox, Textbox olabilir. Yazımı:

form.kontrol.LinkMode = mode

İstemcideki kontrolun LinkMode değerleri:
0 = None, DDE etkileşimi yok.
1= Otomatik .
2= Manual, istemci ancak LinkRequest metodu çalıştırılınca güncellenir.
3= Notify, bağlantılı veri değiştiğinde LinkNotify olayı oluşur, istemcinin LinkRequest metodunu çalıştırması umulur.

Sunucu formdaki LinkMode kurma şekilleri:
0= None, DDE yok. Hiçbir istemci bu formla DDE oturumu başlatamaz, buna veri gönderemez. Tasarım sırasında LinkMode sıfır ise çalışma zamanında (run - time) bir yapılamaz.
1= Sunucudur. İstemcideki kontrol ile DDE oturumu kurulabilir. Tasarım sırasında LinkMode bir yapılmışsa, çalışma-zamanında sıfır veya bir yapılabilir.

İstemcideki kontrolda LinkMode sıfırdan farklı bir değere kurulursa o zaman VB, belirtilen LinkTopic ve LinkItem içeriklerine göre oturum kurmaya teşebbüs eder. Sunucu, bağlantının tipine göre istemcideki veri alanını günceller.

Eğer kontrol "Otomatik" olarak kurulmuşsa, o kontrola her veri girişi bir "Change" olayı yaratır. Sunucu oturumu bitirirse, istemcinin "kontrol" undaki LinkMode sıfıra döner. Bir formun sunucu olabilmesi için onun LinkMode'u tasarım sırasında 1 olmalı. Sonra istemci/sunucu olarak değiştirilebilir.

4.1.4. Shell() Fonksiyonu

Yürütülebilir bir programı çalıştırır. Yazımı:

Shell(Komut_katarı [,pencere türü])

Shell fonksiyonun argümanları şöyledir.

Komut_katarı: Yürütülebilir programın ismi komut satırındaki haliyle burada kullanılabilir. Program ismine uzantı olarak .COM, .EXE, .BAT, .PIF yazılmazsa, bunun .EXE olduğu varsayılır.

Pencere türü: Pencere türünü belirten bir sayıdır. Bu pencerenin içinde çağrılan program çalışacaktır. Sayı yazılmazsa; 2 olduğu kabul edilir.

1, 5, 9 = Erişimli normal.
2 = Erişimli küçültülmüş.
3 = Erişimli büyütülmüş.
4, 8 = Erişimsiz normal.
6, 7 = Erişimsiz küçültülmüş.

Shell fonksiyonu, programı başarıyla başlatabilirse, bunun "varlık ilkelini" döndürür. Varlık ilkeli; çalıştırılan programı tanımlayan bir jetondur. Shell fonksiyonu istenen programı çalıştıramazsa hata oluşur. Bu fonksiyon diğer programları asenkron çalıştırır. Yani, Shell() işine devam ederken bunu izleyen VB program komutları yürütülür.

ÖRNEK:

Bu örnek Shell() fonksiyonunu kullanır, aktif uygulamayı bırakarak Windows'un "Calculator programını çalıştırır. Kullanıcının isteğine bağlı olarak bazı sayıları ekleyen ya da Calculator programından çıkmaya yarayan tuş vuruşlarını gönderen "SendKeys" ifadesi vardır. "AppActivate", erimi VB ve Calculator arasına taşımak için kullanılır.

```
Sub Form_Click ()
    Dim K, X, Msg              ' Declare variables.
    AppActivate "Microsoft Visual Basic"     ' Change focus to Visual
Basic.
    SendKeys "%{ }{Down 3}{Enter}", True   ' Minimize Visual Basic.
    X = Shell("Calc.exe", 1)     ' Shell Calculator.
    For K = 1 To 100     ' Set up counting loop.
        SendKeys K & "{+}", True   ' Send keystrokes to Calculator

    Next K       ' to add each value of K.

    SendKeys "=", True ' Get grand total.
    AppActivate "Microsoft Visual Basic"      ' Return focus to Visual
Basic.
    Msg = "Choose OK to close the Calculator.
    MsgBox Msg ' Display OK prompt.
    AppActivate "Calculator"    ' Return focus to Calculator.
    SendKeys "%{F4}", True    ' Alt+F4 to close Calculator.
    AppActivate "Microsoft Visual Basic"      ' Return focus to Visual
Basic.
    SendKeys "%{ }{Enter}", True       ' Restore Visual Basic to size.
End Sub
```

4.2. Excel

Excel Windows üzerinde çalışan bir elektronik tablodur. Çalışmalar iş sayfalarında (worksheet) yapılır. Sayfalar çalışma kitabını oluşturur (workbook). Birkaç tane çalışma kitabı aynı anda açılabilir.

İş sayfasındaki verilerle bir grafik gösterim yapılabilir. Grafik, verilerle aynı yerdeyse "katılmış grafik" (embedded chart), ayrı bir sayfada ise "grafik sayfası" (chart sheet) hazırlanır. Çalışma kitabı içindeki grafik, gömülü ya da ayrı sayfada olsa bile; verilerini aldığı iş sayfasına bağlıdır. İş sayfası güncellendikçe grafik de güncellenir. Gömülü grafik raporlama için diğeri sunum hazırlamak için uygundur.

4.2.1. Excel'de Makro Yazımı

Microsoft Excel'de sınırlı sayıda tekrarlanan tuş ve fare hareketlerini kestirme yapmak için makrolar yazılır. Makro yazmak için Visual Basic veya Excel 4.0 makro dili kullanılır. VB'de makro yazmak için "VB Module" butonu "Tool bar" dan seçilip "Sub" ile başlayan altprogram yazılır. Excel'de aynı yöntemle kullanıcı tanımlı fonksiyon ve altprogramlar da yazılabilir.

Bir makro, kayıtı yapıldıktan sonra, Excel'de "Tools Menu" ye, bir butona ya da grafik nesnesine bağlanabilir. Böylece kullanımı daha kolay hale gelir. "Tools Menu" ye atılırsa her zaman kullanılabilecek demektir. İş sayfasındaki butona atanırsa, ancak o iş sayfası seçilmişse çalıştırılabilir. İş sayfasındaki butona makro atanması şöyle olur: "Tool bar" içinde "Drawing" butonu seçilir. Bu buton görünmüyorsa, menüden "Tool bars..." seçimliği kullanılarak görünür hale getirilebilir. Sonra, çizim butonlarını içeren pecere

gelir. Buradan buton oluşturmaya ait çizim butonu seçilir. Oluşturulan buton ekranda iş sayfasının bir yerinde dururken "Assign Macro" (makro atama diyalog penceresi) belirir. Buton için, var olan makrolardan biri seçilerek butona atama yapılır ya da yeni makro oluşturulup aynı işlem uygulanır.

4.3. Visual Basic Uygulamaları Arasında DDE Kurulması

Bu kısımda, biri VB istemcisi diğeri VB sunucusu olan iki uygulama arasında DDE kurmak için gerekli adımlar anlatılacaktır. İş adımları:

1) DDE sunucusu olarak çalışacak VB uygulamasını yarat.

2) DDE istemcisi olarak çalışacak VB uygulamasını yarat.

3) Manual DDE bağlantısını iki uygulama arasında kur (istemcinin isteğiyle güncelleme bilgisi alınır).

4) "LinkRequest" kullanarak sunucudaki güncel bilgiyle istemciyi güncelle.

5) Otomatik DDE bağlantısı kur (sunucunun güncel bilgiyi hemen ilettiği durum).

6) "LinkPoke" kullanarak istemciden sunucuya veri gönder.

7) Otomatik ve Manual arasında "LinkMode" kullanarak geçiş yap.

İstemci uygulama DDE aracılığıyla sunucuya iletiler (komutlar, emirler) göndererek bir bağlantı oluşturur. DDE oturumunda, sunucu istemciye bilgiler sağlar veya istemciden bilgi alır.

ÖRNEK:

İki VB uygulaması arasında DDE oturumunun nasıl kurulduğuna dair adımlar aşağıdadır.

Önce sunucu (kaynak) uygulama yaratılır:

1. Visual Basic başlat. "Form1" isimli bir form gelir.

2. Form1'in "Caption" özelliğini (yani formun ismi) "Source" (sunucudur) olarak değiştir.

3. Bir "Text Box" (Text1) nesnesini formun içine koy.

4. Tek formlu projeyi "SOURCE" adıyla sakla.

5. "File" menüsünden, projeyi .EXE hale getir.

İkinci olarak istemci (varış) uygulama yaratılır:

1. "File" menüsünden "New Project" seçilir.

2. Form1 olarak gelen formun "caption" özelliği değiştirilip "Destination" yapılır.

3. Form1 üzerinde şu kontroller bulunmalıdır:

Default Name	Caption	Name
Text1	(İşlevsel değil)	Text1
Option1		Manual Link
ManualLink		
Option2		Automatic Link
AutomaticLink		
Command1	Poke	Poke
Command2	Request	Request

4. Form1'in "General Declaration" kısmına şu kodu ekle:

```
Const  AUTOMATIC = 1
Const  MANUAL = 2
Const  NONE = 0
```

5. Form1'in "Load event" (sadece yüklenirken çalışan kısım) görevi olarak şu yazılmalı:

```
Sub Form_Load()
    'Bu görev VB sunucu uygulamasını başlatır.
    z% = Shell("SOURCE", 1)
    z% = DoEvents()          ' Windows Shell komutunun çalışmasını bitirir.
    Text1.LinkMode = NONE    'DDE bağlantısı zaten varsa kaldırır.
    Text1.LinkTopic = "Source|Form1"    ' VB sunucusuyla bağlantı kurar.
    Text1.LinkItem = "Text1"            'Sunucudaki text kutusuna bağlanır.
    Text1.LinkMode = MANUAL         'Manual DDE bağlantısı kurar.
    ManualLink.Value = TRUE         'Uygun opsiyon butonunu kurar.
End Sub
```

6. Aşağıdaki kodu "ManualLink" isimli butonun "Click event" görevi olarak yaz

```
Sub ManualLink_Click()
    Request.Visible = TRUE          'Request isimli butonu geçerli yapar.
    Text1.LinkMode = NONE           'Varsa DDE temizler.
    Text1.LinkMode = MANUAL      'Yeniden LinkMode kurar.
End Sub
```

7. Aşağıdaki kodu "AutomaticLink" isimli butonun "Click event" görevi olarak yaz.

```
Sub AutomaticLink_Click()
    Request.Visible = FALSE     'Automatic Link isimli butona ihtiyaç yok.
    Text1.LinkMode = NONE           'Varsa DDE temizler.
    Text1.LinkMode = AUTOMATIC      'Yeniden LinkMode kurar.
End Sub
```

8. Aşağıdaki kodu "Request" isimli butonun "Click event" görevi olarak yaz.

```
Sub Request_Click()
    'Manual DDE bağlantısında bu buton görünür olacak ve seçildiği zaman
    'sunucudan güncelleme bilgisi isteyecek.
    Text1.LinkRequest
End Sub
```

9. Aşağıdaki kodu "Poke" isimli butonun "Click event" görevi olarak yaz.

```
Sub Poke_Click()
    'DDE bağlantısının türünden bağımsız olarak bu buton görünür kalır
    'seçildiği zaman istemciden sunucuya veri gönderir.
    Text1.LinkPoke
End Sub
```

Son aşama deneme aşamasıdır. Program .EXE hale getirilip VB'nin dışında ya da doğrudan VB içinde çalıştırılabilir. İstemci uygulama çalıştırılınca sunucu da çalışacaktır. Deneme sırasında şunlar yapılabilir [5]:

a) Sunucuda text kutusu içine birşeyler yazılmaya çalışılır sonra "Request" butonu tıklanır. Bu yazı istemcinin text kutusu içinde belirir.

b) "Automatic Link" butonu tıklanır ve sunucunun text kutusuna birşeyler yazılır. Yazı hemen istemciye gönderilir.

c) İstemcide text kutusuna birşeyler yazılır ve "Poke" butonu tıklanır. Yazı sunucunun text kutusunda belirir.

4.4. Visual Basic'ten Excel'e DDE Kurulması

Bu kısımda istemci VB uygulaması, sunucu Excel uygulamasıdır. İki uygulama arasında DDE oturumunun nasıl kurulacağı anlatılacak. Bunun için işlem basamakları şöyledir:

1) Aktif DDE için bir Excel belgesi hazırla.

2) Manual DDE bağlantısını VB ve Excel arasında kur (istemcinin isteğine göre güncel veri alınacak).

3) "LinkRequest" ile Excel'den VB uygulamasını güncelle.

4) Otomatik DDE bağlantısını VB ve Excel arasında kur (sunucu istemciye güncel veriyi hemen gönderecek)

5) "LinkPoke" ile VB'den Excel uygulamasına bilgi gönder.

6) "LinkMode" otomatik ve manual arasında değiştir.

İstemci uygulama DDE aracılığıyla sunucuya iletiler (komutlar, emirler) göndererek bir bağlantı oluşturur. DDE oturumunda, sunucu istemciye bilgiler sağlar veya istemciden bilgi alır.[7]

ÖRNEK:

VB uygulaması ve Excel arasında DDE oturumunun nasıl kurulduğuna dair adımlar aşağıdadır.

Önce sunucu (kaynak) uygulama (iş sayfası) yaratılır:

1. Excel'i başlat. İş sayfası "SHEET1" ismiyle gelir.

2. Belgeyi "SOURCE.XLS" olarak saklayıp ismini değiştir.

3. Excel'den çık.

Sonraki aşama istemci uygulamanın VB'de yaratılmasıdır:

İstemci uygulama, bağlantı işlerini yapar. Sunucuya doğrudan veri gönderir ya da veri alır.

1. Visual Basic başlatılır.

2. Form1 olarak gelen formun "caption" özelliği değiştirilip "Destination" yapılır.

3. Form1 üzerinde şu kontroller bulunmalıdır:

Default Name	Caption	Name
Text1	(İşlevsel değil)	Text1
Option1		Manual Link
ManualLink		
Option2		Automatic Link
AutomaticLink		
Command1	Poke	Poke
Command2	Request	Request

4. Form1'in "General Declaration" kısmına şu kodu ekle:

```
Const AUTOMATIC = 1
Const MANUAL = 2
Const NONE = 0
```

5. Form1'in "Load event" (sadece yüklenirken çalışan kısım) görevi olarak şu yazılmalı:

```
Sub Form_Load()
    'Bu görev Excel sunucu uygulamasını başlatır.
    z% = Shell("EXCEL SOURCE.XLS", 1)
    z% = DoEvents()          ' Windows olaylarını işler. DDE çalışmadan
    'Excel'in yürütülmesini engeller.

    Text1.LinkMode = NONE    'DDE bağlantısı zaten varsa kaldırır.
```

```
    Text1.LinkTopic = "Excel |source.xls"
     ' Excel sunucusuyla bağlantı kurar.
    Text1.LinkItem = "R1C1"              'Sunucudaki ilk hücreye bağlanır.
    Text1.LinkMode = MANUAL        'Manual DDE bağlantısı kurar.
    ManualLink.Value = TRUE          'Uygun opsiyon butonunu kurar.
End Sub
```

6. Aşağıdaki kodu "ManualLink" isimli butonun "Click event" görevi olarak yaz

```
Sub ManualLink_Click()
    Request.Visible = TRUE           'Request isimli butonu geçerli yapar.
    Text1.LinkMode = NONE            'Varsa DDE temizler.
    Text1.LinkMode = MANUAL       'Yeniden LinkMode kurar.
End Sub
```

7. Aşağıdaki kodu "AutomaticLink" isimli butonun "Click event" görevi olarak yaz.

```
Sub AutomaticLink_Click()
    Request.Visible = FALSE      'Automatic Link isimli butona ihtiyaç yok.
    Text1.LinkMode = NONE            'Varsa DDE temizler.
    Text1.LinkMode = AUTOMATIC       'Yeniden LinkMode kurar.
End Sub
```

8. Aşağıdaki kodu "Request" isimli butonun "Click event" görevi olarak yaz.

```
Sub Request_Click()
    'Manual DDE bağlantısında bu buton görünür olacak ve seçildiği zaman
    'sunucudan güncelleme bilgisi isteyecek.
    Text1.LinkRequest
End Sub
```

9. Aşağıdaki kodu "Poke" isimli butonun "Click event" görevi olarak yaz.

```
Sub Poke_Click()
    'DDE bağlantısının türünden bağımsız olarak bu buton görünür kalır
    'seçildiği zaman istemciden sunucuya veri gönderir.
    Text1.LinkPoke
End Sub
```

Son aşama deneme aşamasıdır. Program (istemci) .EXE hale getirilip VB'nin dışında ya da doğrudan VB içinde çalıştırılabilir. İstemci uygulama çalıştırılınca sunucu da çalışacaktır. Excel'de "Excel Workspace" diyalog kutusunda "Ignore Remote Requests" opsiyonu seçilmiş olmamalı yoksa DDE bağlantısı kurulamaz. Deneme sırasında şunlar yapılabilir:

a) R1C1 hücresine birşeyler yazılır. İstemci tarafında "Request" butonuna basılır. Hücrede yazılanlar text kutusunda belirmeli. Şunlara dikkat edilmeli: Excel hücresine veri girildikten sonra ENTER tuşuna basılmış olmalı sonra VB uygulamasındaki "Request" butonuna basılmalı. Böyle yapılmazsa TEXT1.LINKREQUEST ifadesinden dolayı "Timeout while waiting for DDE response" hatası gelir. Çünkü, hücreye yazarken, Excel bunu bir yoklama çevrimi (polling loop) içinde takip eder, aslında ENTER tuşuna basılana kadar hücreye veri transfer edilmez. Visual Basic istek butonuna basılmasıyla oluşan isteğine devam eder, fakat Excel bu isteği, kendi yoklama çevriminden çıkana kadar dikkate alamaz, o zaman DDE'de bir zaman aşımı hatası oluşur.

b) "Automatic Link" butonu seçilir. Excel'de R1C1 hücresine birşeyler yazılır. Yazı otomatik olarak VB'deki istemci uygulamasına gönderilir.

c) VB istemci uygulamasında text kutusuna birşeyler yazılır. Sonra "Poke" butonu seçilir. Yazı Excel iş sayfasında R1C1 hücresinde belirir.

4.5. Excel'den Visual Basic'e DDE Kurulması

Bu kısımda Excel makrosuyla VB uygulaması arasında DDE oturumunun kurulması anlatılacaktır. İşlem basamakları şunlardır:

1) DDE'yi başlatacak Excel makrosunu hazırla.

2) Manual DDE bağlantısını kur.

3) "Poke" kullanarak Excel'den VB'ye bilgi gönder.

İstemci uygulama DDE aracılığıyla sunucuya iletiler (komutlar, emirler) göndererek bir bağlantı oluşturur. DDE oturumunda, sunucu istemciye bilgiler sağlar veya istemciden bilgi alır.[6]

ÖRNEK:

Excel makrosu ve VB uygulaması arasında DDE oturumunun nasıl kurulduğuna dair adımlar aşağıdadır.

DDE için VB'yi hazırlama:

1. Visual Basic başlat.

2. Form1'e text kutusu ekle.

3. "Properties" düzenlemesi şöyle olur:

Control	Name	Caption	Properties
Form1	Form1	Form1	LinkMode : 1-Source LinkTopic : Form1
Text Box	Text1	Text1	LinkItem : Text1

4. Projeyi DDE.MAK adıyla sakla. Formu DDE.FRM adıyla sakla.

5. Programı DDE.EXE dosyası haline getir .

6. VB'yi kapat.

DDE için Excel'i hazırlama:

1. Excel çalıştığında Seet1.xls hemen yaratılır. C2 hücresine, VB uygulamasına gidecek veriyi yaz. İş sayfasını Sheet1 olarak sakla.

2. Excel'de "File" menüsünden "New" seçildikten sonra "Macro Sheet" seçilerek yeni bir makro sayfası yaratılır.

3. Şu makroyu gir:

```
Record1 (a)
chan = INITIATE("dde", "Form1")
=POKE(chan, "text1", "C:\SHEET1.XLS' ! C2)
=TERMINATE(chan)
=RETURN(chan)
```

"Initiate" fonksiyonu VB uygulamasını başlatır. "Poke" fonksiyonu Sheet1 sayfasındaki C2 hücresinin içeriğini Form1'deki Text1 kutusuna gönderir. "Terminate" fonksiyonu bağlantıyı bitirir. "Return" fonksiyonu makronun sonlanmasını sağlar.

Önemli noktalar:

a) Bağlantı kurmaya çalışmadan önce iş sayfası saklanmış olmalı.
b) Uygulamanın çalışma dizini PATH içinde olmalıdır.
c) "Initiate" fonksiyonunda uygulama için yol belirtilmez.

4. "Macro" menüsünden "Run" seçilir.

5. "Run Macro" diyen bir mesaj görünür. Gerekli yere 'Macro1.xlm!Record1' yazılıp "OK" butonuna basılır.

6. DDE.EXE çalışmıyorsa, mesaj kutusunda "Remote data not acceptable. Start application DDE.EXE" yazacaktır. "YES" seçilir.

7. VB uygulamasında text kutusu içinde veri görünmüyorsa Excel'e dönüp Sheet1 yeniden saklanmalıdır.

BÖLÜM 5

5. DDE'NİN UYGULAMALARLA GERÇEKLENMESİ

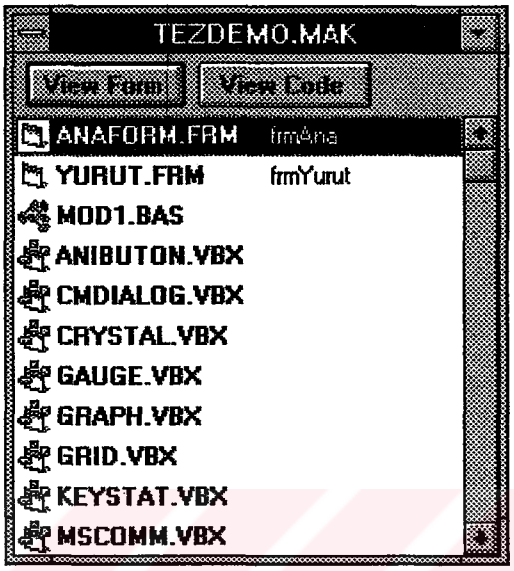
DDE'nin çalışma mantığını anlayabilmek için buradaki olayların bilgisayar ortamında ifade edilmesi gerekir. Önce DDE'nin niçin gerekli olduğunu ve program geliştirme ortamlarının seçiminin nasıl yapıldığını hatırlatmakta yarar var.

DDE'nin gerekliliğini şöyle açıklayabiliriz: Günümüzde yazılım paketleri gittikçe özelleşmektedir. Kelime işlemciler, elektronik tablo programları, vb. kendi alanlarında gelişmekteler. Böyle özelleşen paketlerin hepsi ile haberleşebilen arayüz şeklinde uygulamalar geliştirmek mümkündür.

Burada kullanılan işletim sisteminde (Microsoft Windows 3.1) çalışan uygulamalar İstemci/Sunucu yapısında haberleşme olanağına sahip olabiliyor. Böylece bir uygulama, tasarlandığı amacın dışında, özellikle zayıf bırakılmış yanını diğer uygulamayla bağlantı kurarak bir ölçüde kapatabilir. Bu haberleşmeyi sağlayan protokol sistem tarafından desteklenir. Alt düzeyli ileti komutları vardır. Komutları bu haliyle kullanmak zor olduğu için DDE olayını kotaracak, içinde fonksiyon çağrıları olan DDEML.DLL kütüphanesi kullanılır.

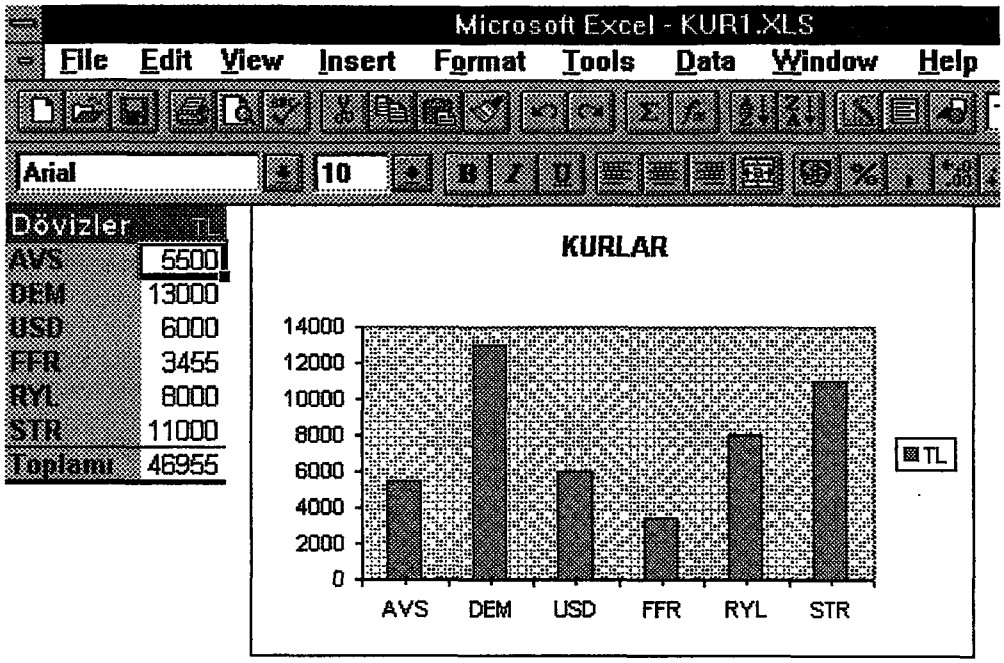
Visual Basic (VB) ve Excel 5.0 (EXCEL) program geliştirme ortamları Windows altında çalışabiliyorlar ve DDE'yi kullanabiliyorlar. Bunun için bu ortamların kullanılması uygun bulundu. Önce VB ile TEZDEMO.MAK programı yazıldı. TEZDEMO.MAK, İstemci tarafı temsil eder. Bölüm 4'te

anlatıldığı gibi TEZDEMO.MAK programı da proje yapısındadır. Proje içindeki formlar; ANAFORM.FRM, YURUT.FRM, MOD1.BAS dir. Ayrıca kullanılan nesnelerin yapısını tanımlayan .VBX uzantılı dosyalar vardır.



Şekil 5.1 TEZDEMO.MAK içindeki form ve kütüphane dosyaları.

EXCEL tarafında içinde bir veri kümesi barındıran "KUR1.XLS" dosyası vardır (Şekil 5.2). EXCEL, Sunucu tarafı temsil eder. Örnek dosya tasarlanırken EXCEL'in makro komutları kullanılmıştır. KUR1.XLS dosyasının oluşturulma amacı, bir veri kümesini kullanıp (burada döviz isimleri ve TL tutarları var, verilerin ne olduğunun önemi yok) onun üzerinde çalışmaktır. Ayrıca veri değerleriyle grafik çizdirildi. Döviz verileri hücreler içinde ve bunların ifadesi olan grafik hemen yanındadır.



Şekil 5.2 KUR1.XLS'nin EXCEL içinde görüntüsü.

EXCEL'de KUR1.XLS'yi hazırlatan makro aşağıdadır. VB ile yazılan makro bir prosedür şeklindedir. VB bilgisi olmadan bu kod anlaşılmaz, okuyucu isterse VB ile EXCEL'de nasıl makro yazıldığını öğrenebilir. Burada bilgi amacıyla makrolar da verildi. TEZDEMO.MAK programı ise alt bölümlerde anlatılacaktır.

5.1. Excel'de Günlük Döviz Kurlarını Gösteren Grafiği Çizdiren Makro (VB ile yazıldı).

```
Sub Macro1()
    Workbooks.Open Filename:="KUR1.XLS"
    Range("A1:B11").Select
    With Toolbars(8)
        .Position = xlTop
        .Left = 484
        .Top = 42
```

```
    End With
    ActiveSheet.ChartObjects.Add(140, 4, 350, 150).Select
    Application.CutCopyMode = False
    ActiveChart.ChartWizard Source:=Range("A1:B11"), Gallery:=xlColumn, _
        Format:=6, PlotBy:=xlColumns, CategoryLabels:=1, SeriesLabels _
        :=1, HasLegend:=1, Title:="Günlük Kurlar", CategoryTitle:= _
        "Dövizler", ValueTitle:="", ExtraTitle:=""
    ActiveSheet.ChartObjects("Chart 1").Activate
    ActiveChart.Axes(xlCategory).Select
    Selection.TickLabels.Orientation = xlHorizontal
    ActiveChart.SeriesCollection(1).Select
    With Selection.Border
        .Weight = xlThin
        .LineStyle = xlAutomatic
    End With
    Selection.InvertIfNegative = False
    With Selection.Interior
        .ColorIndex = 3
        .Pattern = xlSolid
    End With
    ActiveChart.ChartArea.Select
    ActiveWindow.Visible = False
    Windows("KUR1.XLS").Activate
    ActiveSheet.DrawingObjects("Chart 1").Select
    Selection.Height = 180
End Sub
```

5.2. Aynı makronun Excel 4.0'da yazılmış hali.

```
Kurmac1
=OPEN("KUR1.XLS")
=SELECT("R1C1:R11C2")
=CREATE.OBJECT(5;"R1C3";13,5;3;"R14C9";0,75;9;1;TRUE)
=CHART.WIZARD(TRUE;"R1C1:R11C2";3;6;2;;;1;"Günlük
Kurlar";"Dövizler";"";"";1;1)
=VPAGE(2)
=VLINE(-29)
=WINDOW.MAXIMIZE()
```

```
=FORMAT.SIZE(;3,75;"R16C9")
=UNHIDE("[KUR1.XLS]KURLAR Chart 1")
=SELECT("S1")
=PATTERNS(1;1;1;2;;0;1;2;3;FALSE;FALSE;TRUE)
=SELECT("Chart")
=HIDE()
=SELECT("R1C1:R11C2")
=FORMAT.AUTO(8;TRUE;TRUE;TRUE;TRUE;TRUE;TRUE)
=VLINE(4)
=SELECT("R15C1")
=RETURN()
```

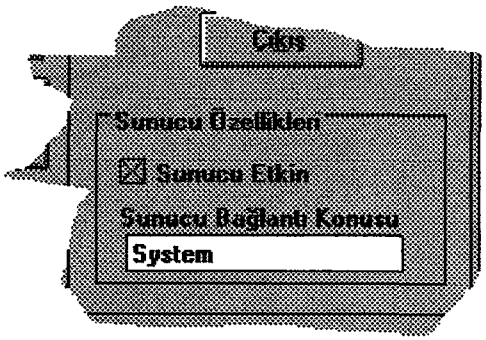
5.3. DDE İçin Hazırlanan TEZDEMO.MAK Programı Hakkında

TEZDEMO.MAK programı ve KUR1.XLS dosyasının hazırlanma nedenleri yukarıda açıklanmıştı. Burada, istemci konumdaki TEZDEMO.MAK programının yapısı ve çalışma ilkeleri anlatılacaktır.

Program başlıca üç kısımdan oluşmaktadır: Temel işlemlerin bulunduğu ANAFORM.FRM ekranı, yardımcı ekran YURUT.FRM, sabitlerin tanımlandığı MOD1.BAS modülü. Programın çalıştırılması için; TEZDEMO.MAK Visual Basic 3.0 ortamına çağırılabilir (veya bu ortam kullanılmadan çalışmak için, önceden .EXE hale getirilebilir) buradan başlatılır. Sistemde "C:\EXCEL" altında sunucu olarak kullanılacak olan EXCEL.EXE bulunmalı. "KUR1.XLS" dosyası "C:\ITU_TEZ\EXCELWRK" dizini altındadır. "PATH" değiştirilerek dosyalar ve uygulamalar farklı yerlerden başlatılabilir. DDE uygulamalarında konu ismi olarak genellikle dosya isimleri kullanılır. Buradaki örnekte "KUR1.XLS" kullanıldı. Program kodu aşağıda verilmiştir, içerisinde yer yer açıklayıcı satırlar vardır. Ana ekran ve yardımcı ekranın pencere şeklinde görünüşleri her kod kısmının başında verilmiştir.

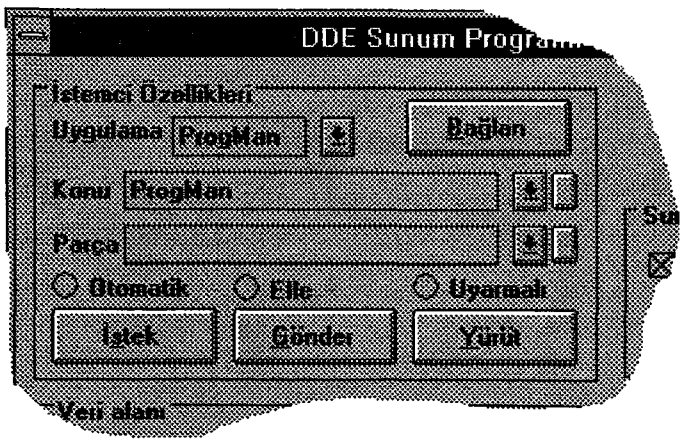
VB'de nesnelerin "Frame" denen bir nesne kullanılarak gruplanabileceği anlatılmıştı. ANAFORM.FRM ekranında böyle "Frame" ler vardır. Eğer ekrandaki "Frame" lerin görevleri içindeki nesnelerle birlikte tek tek açıklanırsa, tüm ekranın çalışması da kendiliğinden ortaya çıkacaktır.

Frame "Sunucu Özellikleri" :



TEZDEMO.MAK uygulaması hem istemci hem de sunucu olarak çalışacak şekilde tasarlandı. "Check box" işaretlenirse sunucu olarak, işaretlenmezse istemci olarak çalışır. "Sunucu bağlantı konusu", Sunucu Etkin durumlu çalışmada önemlidir. İlk değer olarak "System" konusu kabul edilmiştir. Çünkü "System" konulu oturumda istemci olan karşı taraf, sunucunun desteklediği standartları öğrenebilmektedir (örneğin, hangi formatları desteklediği gibi).

Frame "İstemci Özellikleri" :



Ana ekranda temel işlerin yaptırıldığı kısımdır. Burada; Uygulama, Konu, Parça isimlerinin seçilebileceği veya yenilerinin girilebileceği "Combo box" türünde 3 tane nesne vardır. Konu ve Parça "Combo box" larının sağında küçük butonlar vardır. Bu butonların görevi; ekranda yeni girilen Konu veya Parça isimleri varsa bunları Konu ya da Parça listesine ekletmeyi sağlamaktır.

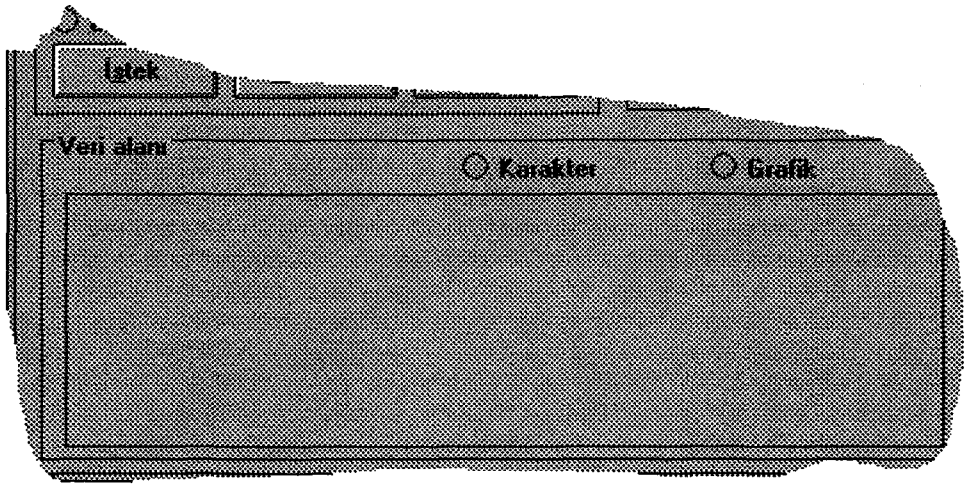
"Otomatik", "Elle", "Uyarmalı" isminde bir opsiyon buton grubu var. Bunlardan bir anda sadece birisi seçilebilir. "Otomatik" seçeneği "Sıcak bağlantı" kurma isteğini, "Elle" seçeneği "Soğuk bağlantı" kurma isteğini, "Uyarmalı" seçeneği "Ilık bağlantı" kurma isteğini belirtir.

"İstek", "Gönder", "Yürüt" butonları seçilen bağlantı türüne göre aktif veya pasif olurlar. Aktif durumda renkleri normal, pasif olunca açık gri rengindedir. "Otomatik" seçilmişse "İstek" pasiftir. "Elle" ve "Uyarmalı" seçilmişse bütün butonlar aktiftir.

Uygulama kutusundan istenen uygulama ismi seçilir. Konu kutusundan istenen konu (genellikle bir dosya ismidir) seçilir. Opsiyon butonlarından bağlantı türü "Elle" seçilir. Sonra "Bağlan" butonuna basılır. Uygulama, çalışır durumda değilse çalışmaya başlar. Dikkat edilirse Parça olarak birşey seçilmedi bunun için bağlantı daha kurulmamış oldu, sadece uygulama (Konuda belirtilen dosyayı da açarak) başlatıldı.

İstemci taraf sunucu hakkında bilgiler edinmek isteyebilir. Bunu yapabilmek için; Konu adı "System" seçilir ve Uygulama için daha önceki isim tekrar seçilir. Parça kutusunda "SysItems" görünmeli. Uygulamanın desteklediği konular ve parçalar uygun kutular tıklanarak anlaşılabilir.

Frame "Veri alanı" :



DDE bağlantısı ana ekranı oluşturan nesnelerden biri ile sunucudaki (Excel olabilir) nesnelerden biri arasındadır. İstemci ve sunucu arasında, nesne sayısı kadar bağlantı kurulabilir. Ekranda "Veri alanı" nesnesi, DDE bağlantısında gönderilen ya da alınan verilerin tutulduğu yerdir. Veriler, karakter veya grafik şeklinde olabilecekleri için bu durumu ayırt etmeye yarayan "Karakter" ve "Grafik" opsiyon butonları kullanılmıştır.

Yukarıda sözü edilen "İstek" butonu, sunucudaki bir nesneyle DDE bağlantısı kurulduktan sonra bunun içeriğini öğrenmek için (veri alanına taşımakta) kullanılır. "Gönder" butonu ise tam tersini yaparak veri alanı içine girilen veriyi sunucu taraftaki (DDE bağlantısı kurulmuş) nesneye gönderir. "Yürüt" butonu, sunucuyu yönetecek şekilde basit komutlar gönderten bir yardımcı ekran çağırır (Şekil 5.3).

Şekil 5.3 TEZDEMO.MAK içinde yardımcı ekran

Kullanıcı sunucuya yürüttürmek istediği komutları "Combo box" içine girer. Sonra "Yürüt" butonuna basarak bunu onaylar. "Komut Ekle" butonuyla "Combo box" listesine komutlar eklenebilir. "Çıkış" butonu ana ekrana dönüşü sağlar.

5.3.1. TEZDEMO Program Kodu

MOD1.BAS

```
==========================================
Genel değişken tanımlamalarının yapıldığı modüldür. Buradaki değişkenler
tüm formlar için globaldir.
==========================================

Option Explicit
Global Const NONE = 0
' Clipboard formatları
Global Const CF_LINK = &HBF00
Global Const CF_TEXT = 1
Global Const CF_BITMAP = 2
Global Const CF_METAFILE = 3
Global Const CF_DIB = 8
Global Const MODAL = 1
'Hata numaraları (LinkError)
Global Const WRONG_FORMAT = 1
```

```
Global Const DDE_SOURCE_CLOSED = 6
Global Const TOO_MANY_LINKS = 7
Global Const DATA_TRANSFER_FAILED = 8
'Fare işaretçisi
Global Const DEFAULT = 0
Global Const HOURGLASS = 11
'Bağlantı kipi  (form ve kontroller için)
Global Const LINK_SOURCE = 1
Global Const LINK_AUTOMATIC = 1
Global Const LINK_MANUAL = 2
'Run-time hatalar
Global Const NO_APP_RESPONDED = 282
Global Const DDE_REFUSED = 285
'Fare butonu maskeleri
Global Const LEFT_BUTTON = 1
Global Const RIGHT_BUTTON = 2

Global Const MB_YESNO = 4
Global Const MB_ICONQUESTION = 32
Global Const IDYES = 6
```

YURUT.FRM

==========

İstemci tarafında "Yürüt" butonuna basıldığında bu ekran gelir. "Combo box" içine sunucunun yürütmesi istenilen komut girildikten sonra bu hareket "Yürüt" butonuna basılarak onaylanır. "Komut Ekle" butonu, "Combo box" içine komutlar eklemek için kullanılabilir. "Çıkış" butonuyla ana ekrana dönülür.

==========

```
==========================================
```
"Combo box" üzerine elle yeni bir komut satırı girildiğinde veya yazılan alan üzerinde değişiklik yapıldığında çalışır. Yardımcı ekrandaki "Yürüt" butonunu aktif yapar.
```
==========================================
```

```
Sub cboYurutStr_Change ()
   cmdOK.Enabled = (Len(cboYurutStr.Text) > 0)
End Sub
```

```
==========================================
```
"Combo box" üzerine fareyle tıklayınca çalışır. Yardımcı ekrandaki "Yürüt" butonunu aktif yapar.
```
==========================================
```

```
Sub cboYurutStr_Click ()
   cmdOK.Enabled = (Len(cboYurutStr.Text) > 0)
End Sub
```

```
==========================================
```
"Çıkış" butonunu fareyle tıklayınca çalışır, yardımcı pencerenin formunu kapatır.
```
==========================================
```

```
Sub cmdCancel_Click ()
   Unload frmYurut
End Sub
```

```
==========================================
```
"Komut Ekle" tıklanınca "Combo box" ın listesine burada girilmiş olan komut satırını ekler.
```
==========================================
```

```
Sub cmdEkle_Click ()
   cboYurutStr.AddItem cboYurutStr.Text
End Sub
```

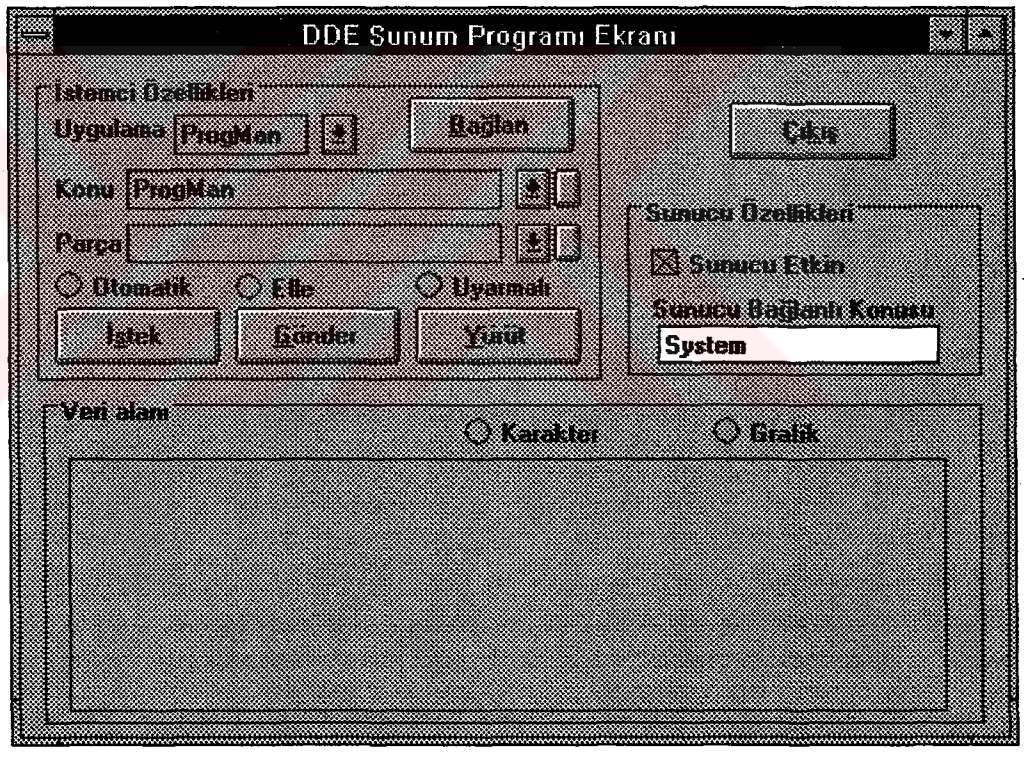
==========

"Yürüt" butonu tıklanınca çalışır. Seçilen veya girilmiş olan komut satırını sunucuya icra ettirir.

==========

```
Sub cmdOK_Click ()
On Error Resume Next
   frmAna.txtVeri.LinkExecute cboYurutStr.Text
   frmAna.picVeri.LinkExecute cboYurutStr.Text
End Sub
```

ANAFORM.FRM




```
Option Explicit
Option Compare Text
Dim UygDegisBay As Integer, KonuDegisBay As Integer
Dim Baglandi As Integer, UyarBay As Integer
Const DEST_TEXT = 0, DEST_PIC = 1
Const MNU_COPY = 0, MNU_PASTE = 1, MNU_PASTELINK = 2
```

==========

"Bağlan" butonuna basıldığında çağrılan görevlerden biridir. "Durum" değişkeni True ise bağlantı zaten vardır ve butonun üzerindeki yazı "Kopar" olarak kalır. "Durum" False ise "Veri alanı" boşaltılır, butonun üzerindeki yazı "Bağlan" olarak değişir. "İstek", "Gönder", "Yürüt" butonlarının aktif pasif durumları ayarlanır.

==========

```
Sub BaglanDurum (Durum As Integer)
Dim i As Integer
   If Durum Then
      cmdBaglan.Caption = "K&opar"
   Else
      txtVeri.Text = " "
      picVeri.Picture = LoadPicture()
      cmdBaglan.Caption = "&Bağlan"
   End If
   Baglandi = Durum
   cmdIstek.Enabled = Durum
   cmdGonder.Enabled = (optBaglanKip(LINK_MANUAL).Value And Durum)
   cmdYurut.Enabled = Durum
   cboUyg.Enabled = Not Durum
   cboKonu.Enabled = Not Durum
End Sub
```

==========

"Bağlan" butonuna basılınca çalışan fonksiyonlardan biridir. Uygulama ismini parametre olarak alır ve bu uygulamayı (çalışır durumda değilse) başlatır. Başlatamazsa sıfır döndürür.

==========

```
Function Baslat (appname As String) As Integer
Dim app As String
app = appname
On Error Resume Next
   Select Case appname
      Case "Excel"
         If cboKonu.Text <> "System" Then
            app = "C:\EXCEL\EXCEL.EXE " & cboKonu.Text
```

```
        Else
          app = "C:\EXCEL\EXCEL.EXE"
        End If
      Case "FoxPro"
        app = "C:\FOXPROW\FOXPROW.EXE"
  End Select
  Baslat = (Shell(app) > 31)
  If Err Then
    MsgBox appname & " uygulaması başlatılamıyor!"
    Baslat = 0
  End If
End Function
```

==========

"Konu" isimli "Combo box" üzerinde değişiklik yapınca çalışır. Konunun değiştiğini belirten bir bayrak çeker. "SistemKonuKontrol" görevini çağırır.

==========

```
Sub cboKonu_Change ()
  KonuDegisBay = True
  SistemKonuKontrol
End Sub
```

==========

"Konu" isimli "Combo box" üzerinden "TAB" tuşuna basarak geçince çalışır. Konu değişti bayrağı çekilmişse bunu False yapar.

==========

```
Sub cboKonu_LostFocus ()
  If KonuDegisBay Then
    KonuDegisBay = False
    If Baglandi Then cmdBaglan.Value = True
    SistemKonuKontrol
  End If
End Sub
```

```
==========================================
```
"Parça" isimli "Combo box" üzerinde değişiklik yapınca çalışır. Ana ekrandaki veri alanına, kutuya yazılan parçayla (sunucu tarafındadır) DDE bağlantısı kurulacağını belirtir.
```
==========================================
```

```
Sub cboParca_Change ()
On Error Resume Next
   picVeri.LinkItem = cboParca.Text
   txtVeri.LinkItem = cboParca.Text
End Sub
```

```
==========================================
```
"Parça" isimli "Combo box" üzerine tıklayarak bir seçim yapılınca çalışır. Ana ekrandaki veri alanına, kutuya yazılan parçayla (sunucu tarafındadır) DDE bağlantısı kurulacağını belirtir.
```
==========================================
```

```
Sub cboParca_Click ()
   picVeri.LinkItem = cboParca.Text
   txtVeri.LinkItem = cboParca.Text
End Sub
```

```
==========================================
```
"Uygulama" isimli "Combo box" üzerine tıklayarak bir seçim yapılınca çalışır.
```
==========================================
```

```
Sub cboUyg_Click ()
   If Baglandi Then cmdBaglan.Value = True
   'Bağlanmış durumda Kopar promptu vardır
   'yeni uygulama seçildiği için önceki bağlantı koparılır,
   'tekrar Bağlan seçilebilecek duruma gelinmiş olunur.
   KonuListeDoldur
   'Seçilen uygulamanın konuları Yurut penceresinde Combo box'a konulur.
   If cboUyg.Text = "Excel" Then
      cboKonu.AddItem "C:\ITU_TEZ\EXCELWRK\KUR1.XLS"
   End If
End Sub
```

==========

"Uygulama" isimli "Combo box" üzerinden "TAB" tuşuna basarak geçince çalışır. Uygulama değişti bayrağı çekilmişse bunu False yapar.

==========

```
Sub cboUyg_LostFocus ()
   If UygDegisBay Then
      UygDegisBay = False
      If Baglandi Then cmdBaglan.Value = True
      KonuListeDoldur
   End If
End Sub
```

==========

"Sunucu Etkin" check box tıklanınca çalışır. Ana ekranın istemci veya sunucu olarak çalışmasını sağlar.

==========

```
Sub chkSunucuKip_Click ()
'Başka bir uygulama tarafından sunucu olarak kullanılacağı zaman
'sunucu etkin yapılır. Bu diğer uygulama VB veya EXCEL uygulaması
olabilir.
   LinkMode = Abs(chkSunucuKip.Value)
   txtSunucuKonu.Enabled = chkSunucuKip.Value
End Sub
```

==========

"Bağlan" butonu tıklanınca çalışır. İstenen bağlantı kipine uygun bağlantıyı ilgili uygulamayla kurar. Bu uygulama çalışmıyorsa onu çalıştırır. Uygulamanın çalıştırılmasına engel bir durum varsa gerekli hata mesajlarını bir pencere içinde ekrana yansıtır.

==========

```
Sub cmdBaglan_Click ()
Dim istemciLinkKip As Integer
   If Not Baglandi Then
      For istemciLinkKip = 1 To 3
         If optBaglanKip(istemciLinkKip).Value Then Exit For
      Next
      picVeri.Picture = LoadPicture()
      txtVeri.Text = " "
      Select Case LinkYap(istemciLinkKip)
```

```
        Case 0
          BaglanDurum True
        Case 1001
          Exit Sub
        Case NO_APP_RESPONDED
          If MsgBox(cboUyg.Text & " uygulaması çalışır durumda değil.
Başlatılsın mı?", MB_YESNO + MB_ICONQUESTION) = IDYES Then
            If Baslat((cboUyg.Text)) Then
              Select Case LinkYap(istemciLinkKip)
                Case 0  'Bağlantı kuruldu
                'Buton durumları ayarlanıyor
                  BaglanDurum True
                Case NO_APP_RESPONDED
                  MsgBox "Şu anda bağlantı yok!"
              End Select
            End If
          End If
      End Select
    Else
      LinkBoz txtVeri
      LinkBoz picVeri
      BaglanDurum False
    End If
End Sub


==========================================
"Konu" combo box içine yazılan konu ismini konular listesine ekletir.
==========================================
Sub cmdEk1_Click ()
'cboKonu'ya text eklenir
   cboKonu.AddItem cboKonu.Text
End Sub
```

```
==========================================
"Parça" combo box içine yazılan konu ismini konular listesine ekletir.
==========================================
Sub cmdEk2_Click ()
'cboParca'ya text eklenir
   cboParca.AddItem cboParca.Text
End Sub


==========================================
Ana ekran üzerindeki "Çıkış" butonu tıklanınca çalışır. Tüm programı bitirir.
==========================================
Sub cmdExit_Click ()
   Unload frmAna
   End
End Sub


==========================================
"Gönder" butonunu tıklayınca çalışır.  Veri alanında karakter opsiyon butonu
seçilmişse karakter verisini, grafik opsiyon butonu seçilmişse grafik verisini
sunucunun DDE ile bağlı nesnesine gönderir.
==========================================
Sub cmdGonder_Click ()
On Error Resume Next
   If optVeriTipi(0).Value Then
      txtVeri.LinkPoke
   Else
      picVeri.LinkPoke
   End If
   If Err Then MsgBox Error & Str(Err)
End Sub
```

==========

"İstek" butonunu tıklayınca çalışır. Veri alanında karakter opsiyon butonu seçilmişse karakter verisini, grafik opsiyon butonu seçilmişse grafik verisini sunucunun DDE ile bağlı nesnesinden getirtir.

==========

```
Sub cmdIstek_Click ()
On Error Resume Next
   txtVeri.LinkRequest
   picVeri.LinkRequest
   UyarBay = False
End Sub
```

==========

"Yürüt" butonunu tıklayınca çalışır. Yardımcı ekranı çağırır. Bu ekranın çalışması yukarıda anlatıldı.

==========

```
Sub cmdYurut_Click ()
'YURUT.FRM deki combox bo_ gelir
'Yürütülebilecek komutlardan bazıları şimdi yüklenecek
   frmYurut.cboYurutStr.Clear  'Burada YURUT.FRM gelir
   'Sunucu uygulamaya uygun komutlar cboYurutStr'ye yüklenir
   Select Case cboUyg.Text
      Case "ProgMan"
         frmYurut.cboYurutStr.AddItem "[CreateGroup(Tez Demo Grubu)]"
         frmYurut.cboYurutStr.AddItem
"[AddItem(C:\ITU_TEZ\WORKS\TEZDEMO.EXE, Tez Demo Program²)]"
         frmYurut.cboYurutStr.AddItem "[ShowGroup(Tez Demo Grubu, 7)]"
      Case "Excel"
         frmYurut.cboYurutStr.AddItem "[SELECT(" & Chr(34) & "R2C1:R7C2"
& Chr(34) & ")]"
         frmYurut.cboYurutStr.AddItem "[NEW(2,2)]"
         frmYurut.cboYurutStr.AddItem "[GALLERY.3D.PIE(4)]"
         frmYurut.cboYurutStr.AddItem "[CLOSE()]"
      Case "FoxPro"
         frmYurut.cboYurutStr.AddItem "[use TEZDEMO.DBF]"
         frmYurut.cboYurutStr.AddItem "[browse last]"
         frmYurut.cboYurutStr.AddItem "[use]"
         frmYurut.cboYurutStr.AddItem "[quit]"
   End Select
```

```
        'Case "Winword"
            'frmYurut.cboYurutStr.AddItem "[StartOfLine][EndOfLine 1]"
            'frmYurut.cboYurutStr.AddItem "[InsertBookMark.Name = " & Chr(34)
& "DDE1" & Chr(34) & "]"
            'frmYurut.cboYurutStr.AddItem "[LineDown 1]"
    frmYurut.Show MODAL
End Sub


==========================================
ANAFORM.FRM yüklenirken yapılacak işler buradadır. Nesnelerin ilk
değerleri atanır.
==========================================
Sub Form_Load ()
    cboUyg.AddItem "ProgMan"
    cboUyg.AddItem "Excel"
    cboUyg.AddItem "FoxPro"
    'Sunucu kipindeyken form ile hangi konuda konuşulacak
    LinkTopic = txtSunucuKonu.Text
    'İstemcinin veri haberleşmesinde kullanabileceği alanların listesi
    Topics.Caption = "Topics" & Chr(9) & "picVeri" & Chr(9) & "txtVeri" &
Chr(13) & Chr(10)
End Sub


==========================================
ANAFORM.FRM biterken yapılacak işler buradadır. Mevcut DDE bağlantıları
kaldırılır.
==========================================
Sub Form_Unload (Cancel As Integer)
    LinkBoz txtVeri
    LinkBoz picVeri
End Sub
```

```
==========================================
"Konu" isimli combo box içine uygulamanın desteklediği konuları doldurur.
==========================================
Sub KonuListeDoldur ()
   cboKonu.Clear
   cboKonu.Text = " "
   If cboUyg.Text = "ProgMan" Then
      cboKonu.Text = "ProgMan"
   Else
   'Uygulamanın desteklediği konuları öğrenecek bir oturum başlatır
      Screen.MousePointer = HOURGLASS
      lblSysLink.LinkMode = NONE
      lblSysLink.LinkTopic = cboUyg.Text & "|" & "System"
      lblSysLink.LinkItem = "Topics"
      On Error Resume Next
      lblSysLink.LinkMode = LINK_MANUAL
      If Err Then
         cboKonu.AddItem "System"
      Else
         lblSysLink.LinkRequest
         'Konular tab ile ayrılmış olarak caption'a gelir.
         ListeDoldur cboKonu, lblSysLink     'Konu combox dolar
         cboKonu.Text = "System"     'Seçilir
      End If
      Screen.MousePointer = DEFAULT
   End If
   cboKonu.Refresh
End Sub


==========================================
DDE bağlantılarını koparma işleri.
==========================================
Sub LinkBoz (Ctl As Control)
Dim tempTimeOutVal
'ProgMan ile bağlantı koparma timeout hatasına neden olur
On Error Resume Next
'Timeout en kısa yapılıp sonra kontrol nesnesinin
' timeout zamanı restore ediliyor
```

```
   tempTimeOutVal = Ctl.LinkTimeout
   Ctl.LinkTimeout = 1
   Ctl.LinkMode = NONE
   Ctl.LinkTimeout = tempTimeOutVal
End Sub


==========================================
"Control" isimli bir nesne üzerinde Uygulama (appname), Konu (topic), Parça
(item) ve bağlantı tipine (LinkTipi) göre bağlantıyı kurar.
==========================================
Function LinkKur (Ctl As Control, appname As String, topic As String, item
As String, LinkTipi As Integer) As Integer
On Error Resume Next
   Ctl.LinkMode = NONE
   Ctl.LinkTopic = appname & "|" & topic
   Ctl.LinkItem = item
   Ctl.LinkMode = LinkTipi
   LinkKur = Err
   If Err = 0 And LinkTipi <> LINK_AUTOMATIC Then
   'Otomatik için istekte bulunmaya gerek yok
      Ctl.LinkRequest
   End If
End Function


==========================================
LinkKur() fonksiyonunu kullanarak bağlantı kurulnmasını sağlar.
==========================================
Function LinkYap (clientLinkMode As Integer) As Integer
Dim BagTxt As Integer, BagPic As Integer
   BagPic = LinkKur(picVeri, (cboUyg.Text), (cboKonu.Text),
(cboParca.Text), clientLinkMode)
   BagTxt = LinkKur(txtVeri, (cboUyg.Text), (cboKonu.Text), (cboParca.Text),
clientLinkMode)
   If BagPic = NO_APP_RESPONDED And BagTxt =
NO_APP_RESPONDED Then
      LinkYap = NO_APP_RESPONDED
```

```
  ElseIf BagTxt = 0 Then  'Uygun durum, bağlantı kurulmuş.
    LinkYap = 0
    optVeriTipi(DEST_TEXT).Value = True   'Basıldı etkisi yaratır
    'Click event ile pic veya txt alanının görünürlüğünü ayarlar
  ElseIf BagPic = 0 Then
    LinkYap = 0
    optVeriTipi(DEST_PIC).Value = True
  Else
    LinkYap = BagPic
  End If
End Function
```

==========

"Control" değişkeniyle temsil edilen nesneyi (combo box) "Lbl" isimli karakter tipli bir nesnenin içinde bulunan elemanlarla doldurur. Bu elemanla TAB karakterleriyle ayrılmış bir karakter katarı şeklindedir.

==========

```
Sub ListeDoldur (cbo As Control, lbl As Control)
Dim i As Integer, lasti As Integer
'lbl.Caption içinde, tablar ile ayrılmış bir string var.
' Bunu Combo box'ın listesine ayrımlayarak ekler.
  Do
    i = i + 1
    lasti = i
    i = InStr(lasti, lbl.Caption, Chr(9))
    If i = 0 Then
      cbo.AddItem Mid(lbl.Caption, lasti)
      Exit Do
    Else
      cbo.AddItem Mid(lbl.Caption, lasti, i - lasti)
    End If
  Loop
End Sub
```

==========

Bağlantı kurulmuşken, bağlantı tipi değiştirilmek istenirse bu görev çalışır. Opsiyon butonu tıklanınca (bağlantı kipini belirtmek için kullanılan buton grubunun bir elemanıdır) "Bağlan" butonunu iki kere tetikler. Bunun nedeni; önce var olan DDE bağlantısı koparılır sonra seçilen bağlantı tipine göre yeni bir DDE oturumu başlatılır.

==========

```
Sub optBaglanKip_Click (Index As Integer)
'Opsiyon butonu seçilince Bağlan butonuna basmadan
'hemen bağlantı kuruyor.
   If Baglandi Then    'Kopar prompt var
      cmdBaglan.Value = True  'Bağlan prompt var
      cmdBaglan.Value = True  'Kopar prompt var
   End If
End Sub
```

==========

"Veri alanı" nın karakter/grafik tipinde veriyi temsil ettiğini belirtmek için "Karakter" ya da "Grafik" opsiyon butonlarında birinin tıklanması sırasında çalışır.

==========

```
Sub optVeriTipi_Click (Index As Integer)
   If Index = DEST_TEXT Then
      txtVeri.Visible = True
      picVeri.Visible = False
   ElseIf Index = DEST_PIC Then
      txtVeri.Visible = False
      picVeri.Visible = True
   End If
End Sub
```

==========

Form unload sırasında; "Kopar" butonuna basılınca, Bağlantı kipini belirten opsiyon (Otomatik/Elle/Uyarmalı) butonuna basılınca çalışır. "BaglanDurum" görevinin ne yaptığı yukarıda açıklandı.

==========

```
Sub picVeri_LinkClose ()
   BaglanDurum False
End Sub
```

```
==========================================
```
"Uyarmalı" bağlantı kipinde çalışırken "Veri alanı" (grafik durumdayken) nesnesinin, sunucu tarafta DDE bağlantısı kurduğu hücrenin içeriğinin değiştiğini, sunucunun bir mesajla bildirmesini sağlar.
```
==========================================
```

```
Sub picVeri_LinkNotify ()
   If Not UyarBay Then
      MsgBox "DDE sunucusunda yeni veri var, İstek butonuyla güncelleme
yapılabilir."
      'Güncel veri için mesaj alındı demek.
      'Kullanıcı isterse bu veriyi alacaktır.
      UyarBay = True
   End If
End Sub
```

```
==========================================
```
"Veri alanı" grafik olarak çalıştığında üzerinde rastgele renkleri olan çizgiler çizilebiliyor. Bu görev, farenin sağ/sol butonuna basıldığında çalışır. Sol butona basılıp (Button = 1 olur) fare işaretçisinin olduğu yer işaretlenir. Sağ butonuna basılınca ise rastgele bir renk seçilir. "Button" değişkeni (parametresi); farenin hangi butonuna basıldığını, "Shift" değişkeni; buton basık durumdayken ne kadar gidildiğini, "X", "Y" değişkenleri; fare işaretçisinin anlık konumunu belirtirler.
```
==========================================
```

```
Sub picVeri_MouseDown (Button As Integer, Shift As Integer, X As Single, Y
As Single)
   If Button And 1 Then
      PSet (X, Y)
   Else
      picVeri.ForeColor = QBColor(Rnd * 16)
   End If
End Sub
```

==========

"Veri alanı" grafik olarak çalıştığında üzerinde rastgele renkleri olan çizgiler çizilebiliyor. Bu görev, farenin sol butonuna basıldığında çalışır. Fare butonu basık tutularak fare işaretçisi hareket ettirilirse veri alanında çizgi çizilir. "Button" değişkeni (parametresi); farenin hangi butonuna basıldığını, "Shift" değişkeni; buton basık durumdayken ne kadar gidildiğini, "X", "Y" değişkenleri; fare işaretçisinin anlık konumunu belirtirler.

==========

```
Sub picVeri_MouseMove (Button As Integer, Shift As Integer, X As Single, Y
As Single)
   'Sol fare butonuna basarak çizgi çizdirir.
   If Button And 1 Then picVeri.Line -(X, Y)
End Sub
```

==========

"Veri alanı" grafik olarak çalıştığında üzerinde rastgele renkleri olan çizgiler çizilebiliyor. Bu görev, farenin sol butonuna basıldığında çalışır. Fare butonu serbest bırakılırken eğer "Otomatik" bağlantı kurulmuş ise veri alanındaki çizimi sunucu taraftaki nesneye gönderir. "Button" değişkeni (parametresi); farenin hangi butonuna basıldığını, "Shift" değişkeni; buton basık durumdayken ne kadar gidildiğini, "X", "Y" değişkenleri; fare işaretçisinin anlık konumunu belirtirler.

==========

```
Sub picVeri_MouseUp (Button As Integer, Shift As Integer, X As Single, Y As
Single)
   If Button And 1 Then picVeri.LinkSend   'Sadece resim verisini
gönderirken End Sub
```

==========

"Konu" combo box içinde "System" veya "Progman" seçilmişse; sunucunun desteklediği parçaları öğrenip "Parça" combo box içine doldurur (SistemParcaDoldur görevi kullanılır) ve kendiliğinden "Elle" butonuna basılmış etkisi yaratır.

==========

```
Sub SistemKonuKontrol ()
Dim i
   If cboKonu.Text = "System" Or cboKonu.Text = "ProgMan" Then
      SistemParcaDoldur
      optBaglanKip(1).Enabled = False
      optBaglanKip(3).Enabled = False
      optBaglanKip(2).Value = True      'Elle butonuna basılmış etkisi yaratır
```

```
    Else
        For i = 1 To 3
            optBaglanKip(i).Enabled = True
        Next
        cboParca.Clear
        cboParca.Text = " "
    End If
End Sub


==========================================
Kullanılabilecek parçaları cboParca combo box içine koyar
==========================================
Sub SistemParcaDoldur ()
    cboParca.Clear
    'Bu proses çalışırken fare işaretçisi form üzerine
    ' getirilirse kum saati belirir
    Screen.MousePointer = HOURGLASS
    'Gizli bir kontroldur
    lblSysLink.LinkMode = NONE
    lblSysLink.LinkTopic = cboUyg.Text & "|" & "System"
    lblSysLink.LinkItem = "Sysitems"
    On Error Resume Next
    lblSysLink.LinkMode = LINK_MANUAL
    'Uygulamanın System oturumunda desteklediği parçalar
    ' bağlantı kurulunca tablı string şeklinde döner
    If Err = 0 Then     'Uygun durumdur
        lblSysLink.LinkRequest
        ListeDoldur cboParca, lblSysLink
        'Dönen string cboParca combox'a ayrımlanarak konur
        cboParca.Text = "Systems"
    End If
    cboParca.Refresh
    Screen.MousePointer = DEFAULT
End Sub
```

```
==========================================
ANAFORM.FRM  sunucu olarak çalışıyorsa bunun konusunu değiştirir.
==========================================
Sub txtSunucuKonu_Change ()
   LinkTopic = txtSunucuKonu.Text
End Sub


==========================================
```

"Veri alan" ı karakter kipinde çalışırken DDE oturumu bitirilirse, ekranın bağlantı olmayan duruma ayarlanmasını sağlar. "BaglanDurum" görevi bu işleri yapar.

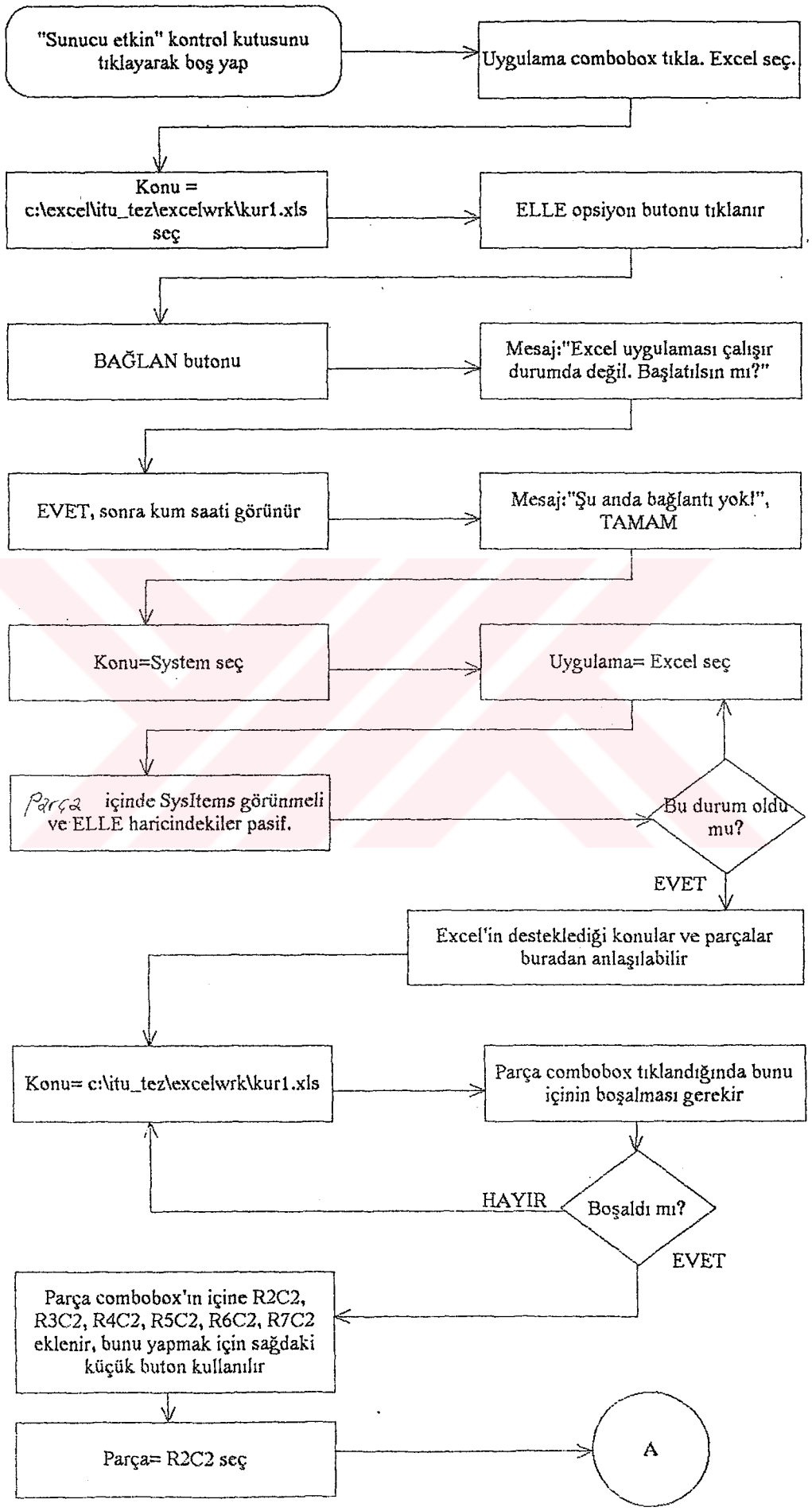
```
==========================================
Sub txtVeri_LinkClose ()
   BaglanDurum False
End Sub


==========================================
```

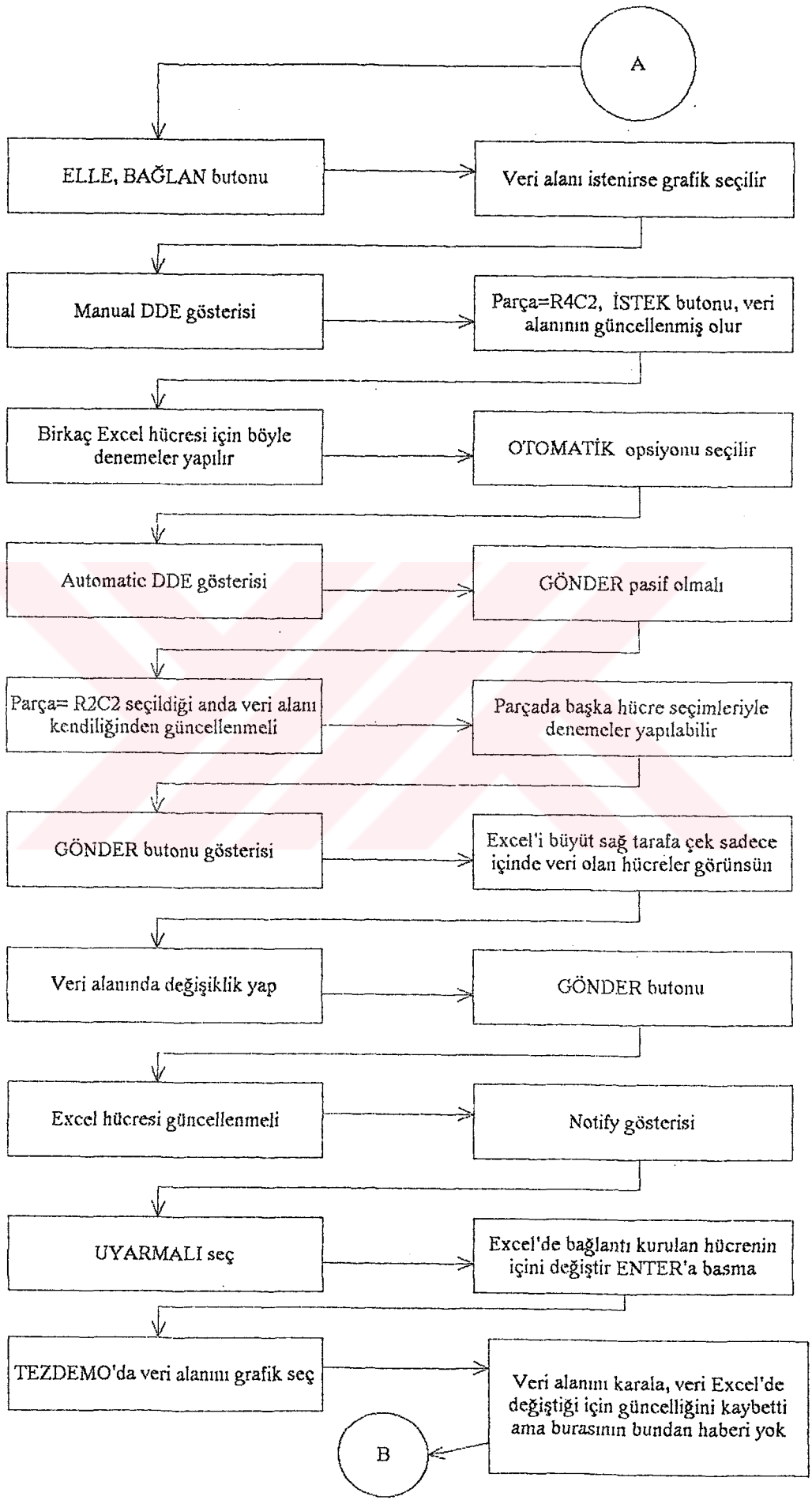
"Uyarmalı" bağlantı kipinde çalışırken "Veri alanı" (karakter durumdayken) nesnesinin, sunucu tarafta DDE bağlantısı kurduğu hücrenin içeriğinin değiştiğini, sunucunun bu durumu bir mesajla bildirmesini sağlar.

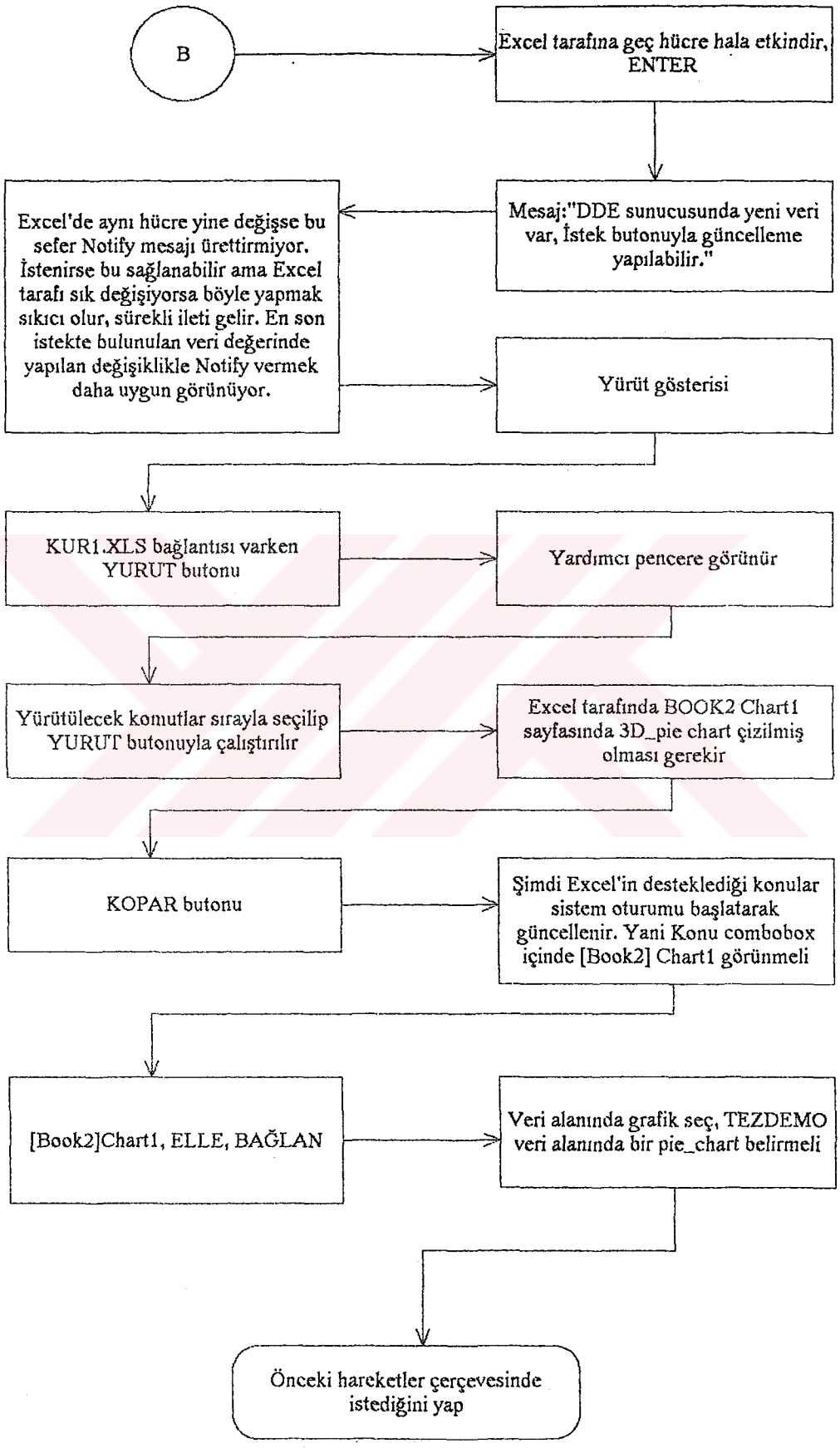
```
==========================================
Sub txtVeri_LinkNotify ()
   If Not UyarBay Then
      MsgBox "DDE sunucusunda yeni veri var, İstek butonuyla güncelleme
yapılabilir."
      'Güncel veri için mesaj alındı demek.
      'Kullanıcı isterse bu veriyi alacaktır.
      UyarBay = True
   End If
End Sub
```

5.3.2. DDE'nin İyi Şekilde Anlaşılması İçin TEZDEMO'nun Tavsiye Edilen Kullanım Prosedürü

Visual Basic ile hazırlanan TEZDEMO programı İstemci, Excel 5.0 Sunucu konumdadır. Amaç; İstemci ve Sunucu arasındaki etkileşim çeşitlerini görmektir. Ayrıca, TEZDEMO.MAK istenirse Sunucu olarak da çalışabilecek şekilde tasarlanmıştır. Bunun için ekrandaki "sunucu etkin" kontrol kutusunu tıklamak yeterlidir. Öyle yapıldığında, karşısında çalışan programında doğal olarak İstemci olması gerekir.


"Sunucu etkin" kontrol kutusunu tıklayarak boş yap
Uygulama combobox tıkla. Excel seç.
Konu = c:\excel\itu_tez\excelwrk\kur1.xls seç
ELLE opsiyon butonu tıklanır
BAĞLAN butonu
Mesaj:"Excel uygulaması çalışır durumda değil. Başlatılsın mı?"
EVET, sonra kum saati görünür
Mesaj:"Şu anda bağlantı yok!", TAMAM
Konu=System seç
Uygulama= Excel seç
Parça içinde SysItems görünmeli ve ELLE haricindekiler pasif.
Bu durum oldu mu?
EVET
Excel'in desteklediği konular ve parçalar buradan anlaşılabilir
Konu= c:\itu_tez\excelwrk\kur1.xls
Parça combobox tıklandığında bunu içinin boşalması gerekir
HAYIR
Boşaldı mı?
EVET
Parça combobox'ın içine R2C2, R3C2, R4C2, R5C2, R6C2, R7C2 eklenir, bunu yapmak için sağdaki küçük buton kullanılır
Parça= R2C2 seç
A


A
ELLE, BAĞLAN butonu
Veri alanı istenirse grafik seçilir
Manual DDE gösterisi
Parça=R4C2, İSTEK butonu, veri alanının güncellenmiş olur
Birkaç Excel hücresi için böyle denemeler yapılır
OTOMATİK opsiyonu seçilir
Automatic DDE gösterisi
GÖNDER pasif olmalı
Parça= R2C2 seçildiği anda veri alanı kendiliğinden güncellenmeli
Parçada başka hücre seçimleriyle denemeler yapılabilir
GÖNDER butonu gösterisi
Excel'i büyüt sağ tarafa çek sadece içinde veri olan hücreler görünsün
Veri alanında değişiklik yap
GÖNDER butonu
Excel hücresi güncellenmeli
Notify gösterisi
UYARMALI seç
Excel'de bağlantı kurulan hücrenin içini değiştir ENTER'a basma
TEZDEMO'da veri alanını grafik seç
Veri alanını karala, veri Excel'de değiştiği için güncelliğini kaybetti ama burasının bundan haberi yok
B


B
Excel tarafına geç hücre hala etkindir, ENTER
Mesaj:"DDE sunucusunda yeni veri var, İstek butonuyla güncelleme yapılabilir."
Excel'de aynı hücre yine değişse bu sefer Notify mesajı ürettirmiyor. İstenirse bu sağlanabilir ama Excel tarafı sık değişiyorsa böyle yapmak sıkıcı olur, sürekli ileti gelir. En son istekte bulunulan veri değerinde yapılan değişiklikle Notify vermek daha uygun görünüyor.
Yürüt gösterisi
KUR1.XLS bağlantısı varken YURUT butonu
Yardımcı pencere görünür
Yürütülecek komutlar sırayla seçilip YURUT butonuyla çalıştırılır
Excel tarafında BOOK2 Chart1 sayfasında 3D_pie chart çizilmiş olması gerekir
KOPAR butonu
Şimdi Excel'in desteklediği konular sistem oturumu başlatarak güncellenir. Yani Konu combobox içinde [Book2] Chart1 görünmeli
[Book2]Chart1, ELLE, BAĞLAN
Veri alanında grafik seç, TEZDEMO veri alanında bir pie_chart belirmeli
Önceki hareketler çerçevesinde istediğini yap

SONUÇ VE ÖNERİLER

20. yüzyılın sonları, sanayi ötesi bir toplumun belirtisi olan bilgi teknolojisi devriminin yaşandığı zaman dilimi olmaktadır. Mikro ve makro evrene doğru devamlı ilerleme azminde olan insanoğlu, bunu yaparken ortaya çıkardığı teknolojiyi artık bilgiyi kontrol etmek için kullanmak istiyor. Bunun için iletişimin en üst seviyeye çıkması gerekiyor.

Sanayi devrimi sonrasında insanlar, aralarındaki iletişimi arttırmak için bir araya gelerek kentleri oluşturdular. Şimdi; gün geçtikçe gelişen iletişim ortamlarında ağlar sayesinde kullanıcılar pek çok bilgi bankasına erişebiliyorlar, haberleşebiliyorlar ve hayatlarını kolaylaştıran birçok etkinliği buralarda gerçekleştiriyorlar. İletişim ortamları geliştikçe kentlerin dağılma süreci hızlanacaktır.

Bu tezin konusu olarak incelenen DDE, bence, global iletişim çabasının, kullanılan uygulamalar şeklinde, Windows 3.1'deki bir mikro yansımasıdır. Sistemdeki çeşitli uygulamalar temeli kurulmuş bir protokol sayesinde birbirleri ile haberleşebilmektedirler. Bir uygulama, tasarlandığı amacın dışında, özellikle zayıf bırakılmış yanını diğer uygulamayla bağlantı kurarak bir ölçüde kapatabilir.

İçinde, başka uygulamalarla bağlantılı çalışan çeşitli nesneler bulunduran uygulamaların geliştirilmesi mümkün olabilmiştir. DDE'nin imkanlarından en iyi derecede faydalanan karmaşık uygulamalar yapılmaktadır. Bunlar için harcanan çabalara bu tezin de katkısı olacağına inanıyorum.

M. İŞBİLEN

KAYNAKLAR

[1] Microsoft Corporation (c) 1992, 1993. Article in CD.
Dynamic - Data - Exchange Concepts.

[2] PETZOLD, Charles. Microsoft Corporation (c) 1988,1992. Article in CD.
Chapter 17,
Dynamic - Data - Exchange (DDE),
Basic Concepts.

[3] Microsoft Corporation (c) 1992, 1993. Article in CD.
Dynamic - Data - Exchange Messages,
Dynamic - Data - Exchange Message Flow,
DDEML.

[4] Microsoft Corporation (c) 1992, 1993. Article in CD.
Exchanging Data with windows-Based Applications,
DDE from the User's Point of View,
Uses of Windows DDE,
Excel 5.1 Function Categories, DDE Formats.

[5] Visual Basic 3.0 for Windows. Article in VB 3.0 Help. Microsoft Corp. (c) 1992, 1993.
Example of Client/Server DDE Between Visual Basic Applications.

[6] Visual Basic 3.0 for Windows. Article in VB 3.0 Help. Article last modified on 20/10/1993.PSS ID number: Q105447. Microsoft Corp. (c) 1992, 1993.
How to Use DDE Between Excel and Visual Basic.

[7] Visual Basic 3.0 for Windows. Article in VB 3.0 Help. Microsoft Corp. (c) 1992, 1993.
DDE from Visual Basic to Excel for Windows.

[8] Visual Basic 3.0 for Windows. Document in VB 3.0 Help. Microsoft Corp. (c) 1992, 1993.

[9] Visual Basic 3.0 for Windows. Article in VB 3.0 Help.
Article ID: Q76551, Microsoft Corp. (c) 1992, 1993.
DDE Example Between VB and Windows Program Manager.

[10] Visual Basic 3.0 for Windows. Article in VB 3.0 Help.
Article ID: Q74862, Microsoft Corp. (c) 1992, 1993.
DDE Example Between VB and Word for Windows.

[11] Visual Basic 3.0 for Windows. Article in VB 3.0 Help.
Article ID: Q93160, Microsoft Corp. (c) 1992, 1993.
How to Estabilish a Network DDE Link Using VB 3.0.

[12] Microsoft Developer Network (MSDN)
Development Library CD April Disc 5, 6, 7

[13] Microsoft TechNet, Technical İnformation Network
CD Volume 2, İssue 6, June 1994

EK 1

ÖNEMLİ KELİMELER

Application = Uygulama.
Application programming interface= Uygulama programlama arayüzü.
Arayüz = İnterface.
Call - back function = Run time programda dışarıdan gelen bir sinyale göre tetiklenen ve o anda çalışmaya başlayan prosedürdür.
Client / Server = İstemci / Sunucu.
Clipboard = Geçici pano.
Cold link = Soğuk bağlantı.
Controls = Genel anlamda nesne demektir.
Conversation = Oturum.
DDE = Dynamic data exchange; Dinamik veri değiş - tokuşu.
Document = Belge; İçinde, özel bir uygulamayla arayüz oluşturulmuş uygulama, pencere şeklindedir; Uygulama.
Dynamic link libraries = Dinamik bağlantı kütüphaneleri.
Event-driven = Olay-tetikli.
Forms = İçinde nesnelerin bulunabildiği VB pencereleri.
Hot link = Sıcak bağlantı.
İtem = Parça.
MDI = Multiple document interface; Belge içinde birden fazla arayüz oluşturan, pencereler vardır.
Mesaj = İleti.
Modules = VB'de yardımcı program parçaları, tüm formlar kullanabilir.
Process = Görev.
Projects = Formlardan oluşan VB uygulaması.
Sheet = Excel sayfası.
Topic = Konu.
Unique = Biricik.
VB = Visual Basic.
Warm link = Ilık bağlantı.

DDE Formats

Microsoft Excel supports several DDE formats. The formats are listed in the following table in the order of precedence defined by Microsoft Excel, from highest precedence (XlTable) to lowest precedence (CF_METAFILEPICT). Clipboard formats that begin with CF_ are formats that are already defined in WINDOWS.H. Clipboard formats without CF_ must be registered before use. For more information about registering formats, see "Registering Clipboard Formats" on page 434.

Clipboard format	Description
XlTable	Microsoft Excel fast table format. For more information, see "Fast Table Format" on page 435.
Biff4	Binary interchange file format (BIFF) for Microsoft Excel version 4.0. For more information, see Chapter 4, "Binary Interchange File Format for Worksheets," which begins on page 191.
Biff3	BIFF for Microsoft Excel version 3.0. For more information, see Chapter 4, "Binary Interchange File Format for Worksheets," which begins on page 191.

Clipboard format	Description
Biff	BIFF for Microsoft Excel version 2.x. For more information, see Chapter 4, "Binary Interchange File Format for Worksheets," which begins on page 191.
CF_SYLK	Microsoft symbolic link (SYLK) format. Microsoft Excel for the Apple Macintosh was originally designed to use SYLK format, and this format is now supported by Microsoft Excel on both Windows and Macintosh platforms.
Wk1	Lotus 1-2-3 Release 2.01 and Release 2.2 format.
Csv	Comma-separated values format, commonly used in BASIC language I/O. It is similar to CF_TEXT format, except that Csv uses commas to separate fields.
CF_TEXT	The simplest form of clipboard data. It is a null-terminated string containing a carriage return and linefeed at the end of each line.
Rich Text Format	A method of encoding formatted text and graphics for easy transfer between applications. Rich Text Format (RTF) is commonly used by document processing programs such as Microsoft Word for Windows and Microsoft Word for the Macintosh.
CF_DIF	An ASCII format used by the VisiCalc spreadsheet program. The format is under the control of Lotus Development Corporation.
CF_BITMAP	A Windows version 2 compatible bitmap.
CF_METAFILEPICT	A metafile picture structure. For complete information, see the documentation for the Microsoft Windows Software Development Kit.

DDE in Other Microsoft Applications

Microsoft Excel and Microsoft Word for Windows both support DDE and provide statements in their macro languages that allow users to write macros that perform DDE. If you have written macros in one of these Microsoft applications, you may find Table 20.1 helpful. It compares the DDE statements and functions in the Microsoft Excel and Microsoft Word macro languages with the equivalent DDE properties and methods in Visual Basic.

Table 20.1 DDE Statements in Microsoft Excel, Word, and Visual Basic

Microsoft Excel	Microsoft Word	Microsoft Visual Basic
INITIATE	DDEInitiate	*LinkMode* = Automatic LinkMode = Manual LinkMode = Notify
REQUEST	DDERequest	**LinkRequest** method
POKE	DDEPoke	**LinkPoke** method
EXECUTE	DDEExecute	**LinkExecute** method
TERMINATE	DDETerminate	LinkMode = None

DDE in Other Microsoft Applications

EK 2

Service Name

Every application that can be a DDE source has a unique service name. Usually this is the application's executable file name without any extension. The following table shows the service names of a few Microsoft applications.

Application	DDE Service Name
Microsoft Excel	Excel
Microsoft Word for Windows	WinWord
Microsoft Project	Project
Microsoft Access	MSAccess
Microsoft FoxPro for Windows	FoxPro

When you set up your own application to act as a DDE server, you can give it its own service name based on the executable file name.

Service Name

EK 2

LinkRequest Method

Asks the source in a dynamic data exchange (DDE) conversation to update the contents of a control.

Syntax

control.LinkRequest

Remarks

The *control* is the name of a text box, picture box, or label involved in a DDE conversation as a destination. **LinkRequest** causes the source application to provide fresh data to the *control*, updating the Text property if the *control* is a text box, the Picture property if the *control* is a picture box, or the Caption property if the *control* is a label.

If the LinkMode property of the *control* is set to Automatic (1), the source application will automatically update the *control* and **LinkRequest** is not needed. If the LinkMode property of the *control* is set to Manual (2), the source application will only update the *control* when requested to with the **LinkRequest** method. If the LinkMode property of the control is set to Notify (3), the source will notify the destination that data has changed by causing the LinkNotify event to occur. The destination must then use **LinkRequest** to actually update the data.

EK 2

LinkRequest Method Example

The example uses LinkRequest to update the contents of a text box with the values in a Microsoft Excel worksheet. To try this example, you must have Microsoft Excel running on your computer. Place some data in the first cells in the first column in the default worksheet (called Sheet1.XLS). Paste the code shown below into the Declarations section of a form that has a text box control called Text1. Then press F5 and click the form.

```
Sub Form_Click ()
    Const NONE = 0, LINK_MANUAL = 2          ' Declare constants.
    If Text1.LinkMode = NONE Then            ' Test link mode.
        Text1.LinkTopic = "Excel|Sheet1"     ' Set link topic.
        Text1.LinkItem = "R1C1"              ' Set link item.
        Text1.LinkMode = LINK_MANUAL         ' Set link mode.
        Text1.LinkRequest                    ' Update text box
    Else
        If Text1.LinkItem  = "R1C1" Then
            Text1.LinkItem = "R2C1"
            Text1.LinkRequest                ' Update text box.
        Else
            Text1.LinkItem = "R1C1"
            Text1.LinkRequest                ' Update text box.
        End If
    End If
End Sub
```

EK 2

LinkPoke Method

Transfers the contents of a control to the source application in a dynamic data exchange (DDE) conversation.

Syntax

control.LinkPoke

Remarks

The *control* is the name of a text box, picture box, or label involved in a DDE conversation as a destination. If *control* is a text box, LinkPoke transfers the current contents of the Text property to the source. If control is a picture box, LinkPoke transfers the current contents of the Picture property to the source. If control is a label, LinkPoke transfers the current contents of the Caption property.

Typically, information in a DDE conversation flows from source to destination. However, LinkPoke allows a destination control to supply data to the source. Not all source applications will accept information supplied in this way; if the source application does not accept the data, an error occurs.

LinkPoke Method Example

The example establishes a DDE link with Microsoft Excel, places some values into cells in the first row of a new worksheet, and charts the values. LinkPoke sends the values to be charted to the Microsoft Excel worksheet. To try this example, Microsoft Excel must be installed and in your path. Paste the code into the Declarations section of a form that has a text box with the default name Text1. Then press F5 and click the form.

```
Sub Form_Click ()
   Const NONE = 0, LINK_MANUAL = 2          ' Declare constants.
   Dim Cmd, I, Q, Row, Z                    ' Declare variables.
   Q = Chr(34)                              ' Define quote marks.
   ' Create a string containing Excel macro commands:
   Cmd = "[ACTIVATE(" & Q &"SHEET1" & Q & ")]"
   Cmd = Cmd & "[SELECT(" & Q & "R1C1:R5C2" & Q & ")]"
   Cmd = Cmd & "[NEW(2,1)][ARRANGE.ALL()]"
   If Text1.LinkMode = NONE Then
      Z = Shell("Excel", 4)                 ' Start Excel.
      Text1.LinkTopic = "Excel|Sheet1"      ' Set link topic.
      Text1.LinkItem = "R1C1"               ' Set link item.
      Text1.LinkMode = LINK_MANUAL          ' Set link mode.
   End If
   For I = 1 To 5
      Row = I                               ' Define row number.
      Text1.LinkItem = "R" & Row & "C1"     ' Set link item.
      Text1.Text = Chr(64 + I)              ' Put value in Text.
      Text1.LinkPoke                        ' Poke value to cell.
      Text1.LinkItem = "R" & Row & "C2"     ' Set link item.
      Text1.Text = Row                      ' Put value in Text.
      Text1.LinkPoke                        ' Poke value to cell.
   Next I
   Text1.LinkExecute Cmd                    ' Execute Excel commands.
End Sub
```

ER 2

LinkExecute Method

Sends a command string to the other application in a dynamic data exchange (DDE) conversation.

Syntax

control.LinkExecute *cmdstring*

Remarks

The LinkExecute method has these parts:

Part	Description
control	Name of the Visual Basic control involved in a DDE conversation. The *control* can be any label, picture box, or text box.
cmdstring	String expression containing a command recognized by the other application.

The actual value of *cmdstring* will vary depending on the source applications. For example, Microsoft Excel and Microsoft Word for Windows will accept command strings that consist of their macro commands enclosed by square brackets. To see what command strings a source application will accept, consult documentation for that application.

LinkExecute Method Example

The example establishes a DDE link with Microsoft Excel, places some values into cells in the first row of a new worksheet, and charts the values. LinkExecute sends Microsoft Excel the command to activate a worksheet, select some values, and chart them. To try this example, Microsoft Excel must be installed and in your path. Paste the code into the Declarations section of a form that has a text box with the default name Text1. Then press F5 and click the form.

```
Sub Form_Click ()
   Const NONE = 0, LINK_MANUAL = 2          ' Declare constants.
   Dim Cmd, I, Q, Row, Z                    ' Declare variables.
   Q = Chr(34)                              ' Define quote marks.
   ' Create a string containing Excel macro commands:
   Cmd = "[ACTIVATE(" & Q &"SHEET1" & Q & ")]"
   Cmd = Cmd & "[SELECT(" & Q & "R1C1:R5C2" & Q & ")]"
   Cmd = Cmd & "[NEW(2,1)][ARRANGE.ALL()]"
   If Text1.LinkMode = NONE Then
      Z = Shell("Excel", 4)                 ' Start Excel.
      Text1.LinkTopic = "Excel|Sheet1"      ' Set link topic.
      Text1.LinkItem = "R1C1"               ' Set link item.
      Text1.LinkMode = LINK_MANUAL          ' Set link mode.
   End If
   For I = 1 To 5
      Row = I                               ' Define row number.
      Text1.LinkItem = "R" & Row & "C1"     ' Set link item.
      Text1.Text = Chr(64 + I)              ' Put value in Text.
      Text1.LinkPoke                        ' Poke value to cell.
      Text1.LinkItem = "R" & Row & "C2"     ' Set link item.
      Text1.Text = Row                      ' Put value in Text.
      Text1.LinkPoke                        ' Poke value to cell.
   Next I
   Text1.LinkExecute Cmd                    ' Execute Excel commands.
End Sub
```

EK 2

LinkTimeout Property

Applies To

Label, picture box, text box.

Description

Determines the amount of time a control waits for a response to a DDE message.

Usage

[*form.*]{*label*|*picturebox*|*textbox*}.**LinkTimeout**[= *duration*]

Remarks

By default, the LinkTimeout property is set to 50 (equivalent to 5 seconds). You can specify other settings in tenths of a second.

DDE response time from source applications varies. Use this property to adjust the time a destination control waits for a response from a source application. This helps avoid generating an error by Visual Basic if a given source application takes too long to respond.

Note The maximum length of time that a control can wait is 65535 tenths of a second, or about 1 hour 49 minutes. Setting LinkTimeout to -1 tells the control to wait the maximum length of time for a response in a DDE conversation. The user can force the control to stop waiting by pressing the Esc key.

Data Type

Integer

LinkItem Property

Applies To

Label, picture box, text box.

Description

Specifies the data passed to a destination control in a DDE conversation with another application; corresponds to the *item* argument in the standard DDE syntax, with *application*, *topic*, and *item* as arguments.

Usage

[*form*.]{*label*|*picturebox*|*textbox*}.LinkItem[= *stringexpression*]

Remarks

To set this property, specify a recognizable unit of data in an application as a reference—for example, a cell reference such as "R1C1" in Microsoft Excel.

Use LinkItem in combination with the LinkTopic property to specify the complete data link for a destination control to a source application. To activate this link, set the LinkMode property.

You set LinkItem only for a control used as a destination. When a Visual Basic form is a source in a DDE conversation, the name of any label, picture box, or text box on the form can be the *item* argument in the *application*|*topic*!*item* string used by the destination. For example, the syntax: =*VizBasicApplication*|*MyForm*!*TextBox1* represents a valid reference from Microsoft Excel to a Visual Basic application. You would enter this syntax for a destination cell in the Microsoft Excel formula bar.

Avoiding infinite loops A DDE control can potentially act as destination and source simultaneously, causing an infinite loop if a destination-source pair is also a source-destination pair with itself. For instance, a text box may be both a source (through its parent form) and destination of the same cell in Microsoft Excel. When data in a Visual Basic text box changes, sending data to Microsoft Excel, the cell in Microsoft Excel changes, sending the change to the text box, and so on, causing the loop.

To avoid such loops, use related but not identical items for destination-source and source-destination links in both directions between applications. For example, in Microsoft Excel, use related cells (precedents or dependents) to link a worksheet with a Visual Basic control, thus avoiding using a single item as both destination and source. You should document any *application*|*topic* pairs you establish if you include a Paste Link command for run-time use.

Note Setting a permanent data link at design time with the Paste Link command from the Edit menu also sets the LinkMode, LinkTopic, and LinkItem properties. This creates a link that is saved with the form. Each time the form is loaded, Visual Basic attempts to reestablish the conversation.

Data Type

String

EK 2

LinkTopic Property

Applies To

Form, label, picture box, text box.

Description

For a destination control—determines the source application and the topic. You use LinkTopic with the LinkItem property to specify the complete data link.

For a source form—determines the topic that the source form responds to in a DDE conversation.

Usage

[*form.*][*label.*|*picturebox.*|*textbox.*]**LinkTopic**[= *link*]

Remarks

The LinkTopic property consists of a string that supplies part of the information necessary to set up either a destination link or source link. The string you use depends on whether you're working with a destination control or a source form. Each string corresponds to one or more elements of standard DDE syntax—*application*, *topic*, and *item*.

Destination control To set LinkTopic for a destination control, use a string with the syntax *application*|*topic*, as follows:

- *application* is the name of the application from which data is requested, usually the executable file name without any extension—for example, "Excel" (for Microsoft Excel).
- The "pipe" character (|, or character code 124) separates the application from the topic.
- *topic* is the fundamental data grouping used in that application—for example, a worksheet in Microsoft Excel.

In addition, for a destination control only, you need to set the related LinkItem property to specify the *item* element for the link. A cell reference such as "R1C1," for example, would correspond to an item in a Microsoft Excel worksheet.

Source form To set LinkTopic for a source, change the default string, *FormN*, to an appropriate identifier for the form. A destination application uses this string as the *topic* argument when establishing a DDE link with the form. Although this string is all you need to set LinkTopic within Visual Basic for a source form, the destination application itself also needs to specify:

- The *application* element that the destination application uses, which is either the Visual Basic project name without the .MAK extension (if you are running your application in the Visual Basic development environment), or the Visual Basic application name without the .EXE extension (if you are running your application as a stand-alone executable file). The EXEName property of the App object provides this string in your Visual Basic code unless the file name was changed by the user. (App.EXEName always returns the actual file name of the application on disk, whereas DDE always uses the original name that was specified in the Make EXE dialog.)
- The *item* element that the destination application uses, which corresponds to the Name property for the label, picture box, or text box on the source form.

An example of a valid reference from Microsoft Excel to a Visual Basic application acting as a source is: =*VizBasicApplication*|*FormN*!*TextBox1*. You would enter this reference for a destination cell in the Microsoft Excel formula bar.

Strings and syntax While the standard definition for a DDE link includes the *application*, *topic*, and *item* elements, the actual syntax used within applications for a destination link to a source application may vary slightly. For example, within Microsoft Excel, you use the syntax *application*|*topic*!*item*. Within Microsoft Word for Windows, you use *application topic item* (no pipe character or exclamation mark). Within a Visual Basic application, you use *application*|*topic;* the exclamation mark (!) for *topic* is implicit.

Activating and maintaining data links To activate the data link set with LinkTopic, set the LinkMode property to the appropriate nonzero value to specify the type of link you want. As a general rule, you should set LinkMode after you specify LinkTopic. For a destination control, changing LinkTopic breaks an existing link and terminates the DDE conversation. For a source form, changing LinkTopic breaks all destination links that are using that topic. For these reasons, you should always set the LinkMode property to 0 before changing LinkTopic.

EK 2

these reasons, you should always set the LinkMode property to 0 before changing LinkTopic. After changing LinkTopic for a destination control, you must set LinkMode to 1 (Automatic), 2 (Manual), or 3 (Notify) to establish a conversation with the new topic.

Note Setting a permanent data link at design time with the Paste Link command from the Edit menu also sets the LinkMode, LinkTopic, and LinkItem properties. This creates a link that is saved with the form. Each time the form is loaded, Visual Basic attempts to reestablish the conversation.

Data Type

String

LinkMode Property

Applies To

Form, label, picture box, text box.

Description

Determines the type of link used for a DDE conversation and activates the connection as follows:

- Control—allows a destination control on a Visual Basic form to initiate a conversation as specified through the control's LinkTopic and LinkItem properties.
- Form—allows a destination application to initiate a conversation with a Visual Basic source form, as specified through the destination application's *application|topic!item* expression.

Usage

[*form.*][*label.*|*picturebox.*|*textbox.*]LinkMode[= *mode*]

Remarks

For controls used as destinations in DDE conversations, the LinkMode property settings are:

Setting	Description
0	(Default) None. No DDE interaction.
1	Automatic. Destination control is updated each time the linked data changes.
2	Manual. Destination control is updated only when the LinkRequest method is invoked.
3	Notify. A LinkNotify event occurs whenever the linked data changes, but the destination control is updated only when the LinkRequest method is invoked.

For forms used as sources in DDE conversations, the LinkMode property settings are:

Setting	Description
0	(Default) None. No DDE interaction. No destination application can initiate a conversation with the source form as the topic, and no application can poke data to the form. If LinkMode is 0 (None) at design time, you cannot change it to 1 (Source) at run time.
1	Source. Allows any label, picture box, or text box on a form to supply data to any destination application that establishes a DDE conversation with the form. If such a link exists, Visual Basic automatically notifies the destination whenever the contents of a control are changed. In addition, a destination application can poke data to any label, picture box, or text box on the form. If LinkMode is 1 (Source) at design time, you can change it to 0 (None) and back at run time.

The following conditions also apply to the LinkMode property:

- Setting LinkMode to a nonzero value for a destination control causes Visual Basic to attempt to initiate the conversation specified in the LinkTopic and LinkItem properties. The source updates the destination control according to the type of link specified (automatic, manual, or notify).
- If a label, picture box, or text box has established an automatic link with a source application, a Change event occurs for that control each time new data is entered, whether or not the value of the data differs from the control's current contents. Note that this is the only way a Change event can occur for a label control.
- If a source application terminates a conversation with a Visual Basic destination control, the value for that controls LinkMode setting changes to 0 (None).
- If you leave LinkMode for a form set to the default 0 (None) at design time, you cannot change LinkMode at run time. If you want a form to act as a source, you must set LinkMode to 1 (Source) at design time. You can then change the value of LinkMode at run time.

Note Setting a permanent data link at design time with the Paste Link command from the Edit menu also sets the LinkMode, LinkTopic, and LinkItem properties. This creates a link that is saved with the form. Each time the form is loaded, Visual Basic attempts to reestablish the

EK 2

saved with the form. Each time the form is loaded, Visual Basic attempts to reestablish the conversation.

Data Type

Integer

EK 2

LinkItem, LinkMode, LinkTopic Properties Example

In the example, each mouse click causes a cell in a Microsoft Excel worksheet to update the contents of a Visual Basic text box. To try this example, start Microsoft Excel, open a new worksheet named Sheet1, and put some data in the first column. Then in Visual Basic, create a form with a text box. Paste the code into the Declarations section and press F5 to run the program.

```
Sub Form_Click ()
   Dim CurRow As String
   Static Row                              ' Worksheet row number.
   Row = Row + 1                           ' Increment Row.
   If Row = 1 Then                         ' First time only.
      ' Make sure the link is not active.
      Text1.LinkMode = 0
      ' Set the application name and topic name.
      Text1.LinkTopic = "Excel|Sheet1"
      Text1.LinkItem = "R1C1"              ' Set the LinkItem.
      Text1.LinkMode = 1                   ' Activate the link to
   Else
      ' Update the Row in the data item.
      CurRow = "R" & Row & "C1"
      Text1.LinkItem = CurRow              ' Set the LinkItem.
   End If
End Sub
```

EK 2

LinkExecute Event

Applies To

Form, MDI Form.

Description

Occurs when a command string is sent by a destination application in a DDE conversation. The destination application expects the source application to perform the operation described by the string.

Syntax

Sub {Form|MDIForm}_LinkExecute (*CmdStr* As String, *Cancel* As Integer)

Remarks

The LinkExecute event uses these arguments:

Argument	Description
CmdStr	The command string sent by the destination application.
Cancel	This argument's value at the end of the LinkExecute event procedure tells the destination whether the command string was accepted or refused. Setting *Cancel* = 0 informs the destination that the command string was accepted. Setting *Cancel* to any nonzero value informs the destination that the command string was rejected. (The default is set to -1, indicating *Cancel*.)

There is no required syntax for *CmdStr*. How your application responds to different strings is completely up to you.

If you have not created a LinkExecute event procedure, Visual Basic rejects command strings from destination applications.

EK 2

LinkExecute Event Example

The example defines a set of commands for destinations to use in DDE conversations to which your application will respond. This example is for illustration only.

```
Sub Form_LinkExecute (CmdStr As String, Cancel As Integer)
   Cancel = False
   Select Case LCase(cmdStr)
   Case "{big}"
      WindowState = 2          ' Maximize window.
   Case "{little}"
      WindowState = 1          ' Minimize window.
   Case "{hide}"
      Visible = False          ' Hide form.
   Case "{view}"
      Visible = True           ' Display form.
   Case Else
      Cancel = True            ' Execute not allowed.
   End Select
End Sub
```

EK 2

LinkError Event

Applies To

Form, MDI Form, label, picture box, text box.

Description

Occurs when there is an error during a DDE conversation. This event is recognized only as the result of a DDE-related error that occurs when no Visual Basic code is being executed. The error number is passed as an argument.

Syntax

Sub {Form|MDIForm}_LinkError (*LinkErr* As Integer)

Sub *ctlname*_LinkError ([*Index* As Integer,]*LinkErr* As Integer)

Remarks

The LinkError event uses these arguments:

Argument	Description
Index	Uniquely identifies a control if it is in a control array.
LinkErr	The error number.

Use a LinkError procedure to notify the user of the particular error that has occurred. You might also include code to fix the problem, or troubleshooting information on reestablishing a connection or on where to go for assistance. For brief messages, use the MsgBox function and MsgBox statement.

The following table lists all error numbers returned for the *LinkErr* argument and a brief explanation of each error.

Error number	Explanation
1	The other application has requested data in the wrong format. This error may occur several times in succession as Visual Basic tries to find a format the other application recognizes.
6	The destination application attempted to continue performing DDE after you set the LinkMode on your source form to 0 (None).
7	All the source links are in use (there is a limit of 128 links per source).
8	For destination controls: an automatic link or LinkRequest method failed to update the data in the control.

For source forms: the destination attempted to poke data to a control and the attempt failed.

11	Not enough memory for DDE.

The missing numbers in this table correspond to error values that were supported in Visual Basic 1.0 but no longer occur in more recent versions of Visual Basic.

EK 2

LinkError Event Example

The example is attached to a text box control, `MyTextBox`, which handles selected errors. The procedure displays a message (adapted from the error list in the LinkError Event topic), based on the error number passed as the argument `LinkErr`. Note that you can adapt this code to a source form by substituting `Form_LinkError` for `MyTextBox_LinkError`. This example is for illustration only.

```
Sub MyTextBox_LinkError (LinkErr As Integer)
Dim Msg
   Select Case LinkErr
      Case 1
         Msg = "Data in wrong format."
      Case 11
         Msg = "Out of memory for DDE."
   End Select
   MsgBox Msg, 48, "MyTextBox"
End Sub
```

EK 2

LinkClose Event

Applies To

Form, MDI Form, label, picture box, text box.

Description

Occurs when a DDE conversation terminates. Either application in a DDE conversation may terminate a conversation at any time.

Syntax

Sub {Form|MDIForm}_LinkClose ( )

Sub *ctlname*_LinkClose (*Index* As Integer)

Remarks

The argument *Index* uniquely identifies a control if it is in a control array. Typically, you use a LinkClose procedure to notify the user that a DDE conversation has been terminated. You might also include troubleshooting information on reestablishing a connection or where to go for assistance. For brief messages, use the MsgBox function or statement.

EK 2

LinkOpen Event

Applies To

Form, MDI Form, label, picture box, text box.

Description

Occurs when a DDE conversation is being initiated.

Syntax

Sub {Form|MDIForm}_LinkOpen (*Cancel* As Integer)

Sub *ctlname*_LinkOpen ([*Index* As Integer,]*Cancel* As Integer)

Remarks

The LinkOpen event uses these arguments:

Argument	Description
Index	Uniquely identifies a control if it is in a control array.
Cancel	When the LinkOpen event procedure ends, the value of this argument determines whether the DDE conversation is established or not. Leaving *Cancel* = 0 establishes the conversation. Setting *Cancel* to any nonzero value refuses the conversation.

This event occurs for forms when a destination application is initiating a DDE conversation with the form. It occurs for controls when a control is initiating a DDE conversation with a source application.

EK 2

LinkNotify Event

Applies To

Label, picture box, text box.

Description

Occurs when the source has changed the data defined by the DDE link, if the LinkMode property of the destination control is set to 3 (Notify).

Syntax

Sub *ctlname*_LinkNotify (*Index* As Integer)

Remarks

The argument *Index* uniquely identifies a control if it is in a control array. Typically, in the LinkNotify event your code notifies the user, gets the new data immediately, or defers getting the data until later. You can use the LinkRequest method to obtain the new data from the source.

EK 2

LinkNotify Event Example

The example is attached to a picture box control, `Picture1`, which has its LinkMode property set to 3 (Notify) and its LinkTopic and LinkItem properties set to specify picture data in the source. When the source changes this data, the procedure updates the picture box immediately only if the picture box is on the active form (the form that has the focus); otherwise it sets a flag variable. The example is for illustration only.

```
Sub Picture1_LinkNotify ()
   If Screen.ActiveForm Is Me Then
   Picture1.LinkRequest       ' Picture is on active form, so update.
   Else
      NewDataFlag = True      ' Assumed to be a module-level variable.
   End If
End Sub
```

ÖZGEÇMİŞ

21/8/1969'da Tatvan/Bitlis'te doğmuşum. Babamın Şark hizmeti bitince 1974'te Lüleburgaz'a (Kırklareli) geldik. İlk öğrenimimi Lüleburgaz Hürriyet İlkokulunda 1980'de tamamladım. 1983'te Lüleburgaz Orta Okulunu bitirdim. Meslek liseleri sınavına girip Lüleburgaz Endüstri Meslek lisesi Elektrik bölümünü kazandım. Bu liseyi 1986'da birincilikle bitirdim. Aynı sene üniversite sınavıyla İ.T.Ü. Elektrik mühendisliğini kazandım. Temmuz-1990'da okulu (71.65 iyi dereceyle) bitirerek Elektrik mühendisi oldum. Eylül-1990'da Milli Eğitim Bakanlığı yurt dışı sınavlarına girdim, kazandım. İ.T.Ü. bilim sınavına girdim İ.T.Ü. Bigisayar Bilimlerinde mastır hakkı elde ettim. Yurt dışına, mecburi hizmet süresi fazla olduğu için gitmedim. Ekim-1990'da İ.T.Ü.' de İngilizce hazırlık programına başladım. 1991-92'de Bilgisayar bölümündeki fark derslerini verdim. 1993-94'de mastır derslerini tamamlayıp tez aşamasına geldim. Ekim-1993'de özel bir şirkette işe başladım, halen bu şirkette bilgi işlem sorumlusu ve programcı olarak çalışmaktayım.

M. İŞBİLEN