http://www.trakya.edu.tr/Enstituler/FenBilimleri/Dergi/net/index.htm
Trakya Univ J Sci, 4(2): 203-213, 2003
ISSN 1302 647X
DIC: 85 FYET420312030104

Derleme / Review

ANA BİLGİSAYARLAR İLE KİŞİSEL BİLGİSAYARLAR ARASINDAKİ ARAYÜZLERİN SİSTEM BÜTÜNLEŞTİRİLMESİNDEKİ ÖNEMİ

Ferzan YALKUT*, Aydın CARUS**, A. Mesut RAZBONYALI***
* ATG Bilgisayar AŞ.
** Trakya Üniversitesi, Müh. Mim. Fak. Bilgisayar Müh. Bölümü 22030 EDİRNE, tel.: 0 284 235 85 31
e-mail: aydinc@trakya.edu.tr
*** Maltepe Üniversitesi, Müh. Mim. Fak. Bilgisayar Müh. Bölümü İSTANBUL

Alınış : 04.02.2003
Kabul Ediliş : 30.05.2003

Özet: Ana bilgisayar ile kişisel bilgisayarlar arasındaki akıllı arayüzler; günümüz bilişim teknolojisi yayınlarında sıklıkla kullanılan e-business, B2B ve B2C gibi kavramların kombinasyonlarından oluşan ve bu kavramları bütünleştiren bir teknolojidir. Bu çalışmada bilgisayar ağ teknolojileri, intranet, extranet, internet ve ortakat mimarisi incelenmiştir.
Dünyadaki kuruluşlar, genellikle ana bilgisayar platformlarında bulunan eski büyük uygulamaları yeni teknoloji ile kullanmanın en iyi yollarını aramaktadırlar.
Bu çalışmada, kuruluşların ana bilgisayarlardaki veri ve uygulamalara, kişisel bilgisayarlar kullanılarak erişim ve denetimin sağlanması için gerekli yazılım mimarilerinin tanımları ortaya konulmuştur. Böylece varolan kurumsal uygulamaların gücü arttırılmakta, daha etkin geliştirme dilleri, işletim ortamı ve yeni yazılım mimarileri kullanılarak eski teknoloji veya eskiden alınmış olan zayıf tasarım kararlarının etkisini azaltmak için çözümler sunulmuştur.

Anahtar kelimeler : API, Arayüz, Entegrasyon, ODBC, Ortakat yazılımı, OLE

The Importance of System Integration Interfaces Between Mainframe Computers And Personel Computers

Abstract: Nowadays we have used many smart interfaces between mainframe computer and personnel computer. This integration technology is often mentioned and published many articles as a kind of combination of B2B,B2C,e-business, m-business concepts. We have mainly evaluated in our study Computer Network, Intranet, Extranet, Internet Technologies and middleware architecture .
All of the Enterprises has been researching the optimal way of how to use the conventional technology based products on their mainframe with the new advanced technology ?
Basically we are defining software architecture which has been used to access and authenticate the stored data & programs in the Mainframe Computers by the help of Personel Computers. As a result we reached a solution for the enterprises. This approach is empowered the used mainframe application and solution by the help of increase of effectiveness, flexibilty, feasibility using the new software developpment platforms, languages and tools.

Key words : API, Interface, Integration, Middleware, ODBC, OLE

Giriş

Günümüzde, bilgisayar kullanımını Ana Bilgisayarlar (Mainframe), Kişisel Bilgisayarlar (PC), internet ve bunlara bağlanabilen akıllı aygıtlar olarak sınıflamak mümkündür. Bu donanımlar arasında veri ve uygulama bütünlüğünün sağlanması için kullanıcı ara yüzlerine ihtiyaç duyulmaktadır.

Bu çalışmada mevcut uygulamalarda ve sistemlerde, herhangi bir değişiklik yapılmadan ara yüzler kullanılarak sistemler arasında veri aktarımı ve kullanımı yöntemleri incelenmiş ve olası çözüm önerileri sunulmuştur. Bu öneriler ile kurumların sistem bütünleşme süreçlerini kısaltmak amaçlanmaktadır. Aynı zamanda; mevcut sistemlerin yeniden tasarlanarak gerçekleştirilmesi yerine, ara yüzlerle bütünleşmesi ekonomik olarak ta ciddi avantajlar sağlamaktadır.

Çoğu kullanıcı, çalıştıkları ortama uyum sağlayan, yerel ve uzaktan erişimli hizmetler arasında şeffaf bir şekilde gidip gelebilen ve büyük ölçüde aygıttan ba-

ğımsız tek ve bütünleşik ortamı tercih edecektir. Dolayısıyla, mevcut uygulamalar değiştirilmeden ana bilgisayarların masaüstü olanaklar ile donatılması için ara yüz kavramı ortaya konulmuştur.

Başlangıçta bilgiler merkezi sistemlerde depolanıp oradan kullanırken, PC'lerin yaygınlaşması sonucu veri kullanımı masa üstü sistemlere dağılmaya başlamıştır. Ancak verilerin çok farklı platformlarda olması yeni bir yapının ortaya çıkmasına neden olmuştur. Bu soruna çözüm olarak, İstemci/sunucu (Client/Server-C/S) mimarisi geliştirilmiştir. Ancak bu mimari ortaya koyulmadan önce geliştirilen uygulamalar, mimariye tam uygun olmadığı için beklenilen sonuçlar tümüyle elde edilememiş ve bazı sorunlar yaşanmıştır. Bu sorunlar;

- Bilgisayar Sistemleri Çeşitliliği,
- Farklı Protokoller,
- Veri Farklılıkları, şeklindedir.

Yazılım mühendisleri programlarını; ağ protokolleri, işletim sistemleri, mikro işlemciler ve farklı veri tabanlarına bağlı olmayan, yani bunlar değişikliğe uğradığında programlarını yeniden gözden geçirerek yeni koşullara karşı test etmeden ve gerekli değişikliği yapmadan, geliştirmek istemektedirler. Bu soruna çözüm olarak ortakat yazılım mimarisi ortaya konmuştur. Ortakat mimarisinde ortakat yazılım servisleri ara yüzlerden oluşan bir bütündür. Bu ara yüzler; işletim sistemi, ağ servisleri ve veri tabanı servisleri ile ilgili olmaktadır. Uygulama programları ara yüzleri ağ üzerindeki bir uygulama için sağladığı avantajlar ise;

- Farklı platformlardaki uygulamalarla ve servislerle etkileşim halindedir,
- Ağ servislerinden bağımsızlık sağlanabilmiştir,
- Güvenilirliği ve kullanışlılığı sağlanmıştır,
- Hiçbir fonksiyon kaybedilmeden büyüme sağlanmıştır,

Bu çalışmada farklı platformlardaki verilerin değişim ve kullanım yöntemleri araştırılarak, mevcut ana bilgisayarlardaki uygulamaların değiştirilmeden masaüstüne getirilmesi için uygun yöntemin belirlenmesinin avantajlar ortaya konmuştur.

Ortakat Mimarisi

Ana bilgisayarlar ile kişisel bilgisayarlar arasındaki veri ve uygulama değişimine temel olan ortakat mimari aşağıdaki biçimlerde olabilmektedir;

Çevrimiçi işlemlerin görüntülenmesi (Online Transaction Processing Monitors -OLTP)

OLTP Monitorları ortakyol yazılımları içinde küçük bir grubu oluşturuyorlar ise de, çok büyük bir öneme sahiptirler. Bu tip ortakyol yazılımları bilgisayarın hızını arttırmak için tasarlanmışlardır ve bunun sonucu şirketler kolaylıkla geniş ve karmaşık uygulamalar gerçekleştirebilirler. OLTP Monitorun seçimi için iki temel neden bulunmaktadır. Birincisi; tümleşik yönetim, ikincisi; güvenlik yeteneğidir. Pek çok OLTP Monitor üretimi yapan şirket, yönlendirme, kurma ve uygulamaların yerleştirilmesi için gerekli olan tüm sistemin ağ yöneticileri tarafından izlenmesini olanaklı kılan bir grafiksel arayüz sağlamaktadır (Judith, 2002).

Uzaktan Erişim Sistemlerinin Harekete Geçirilmesi (Remote Procedure Call -RPC)

Bilgisayar firmaları sistemlerini dağıtık sistemlere bağlama arayışına girdiklerinde RPC gibi bir mekanizmanın zorunluluğu ortaya çıkmıştır. RPC'nin zamana uyumlu yapısı nedeniyle RPC istemci ve sunucunun daima (her zaman bu koşulun yerine getirilmesi garanti edilemez) kullanıma hazır ve fonksiyonel olmaları gerekmektedir. İstemci veya sunucudan herhangi biri kullanıma hazır değil ise (Blok'lanmış durumda ise) RPC problemle karşılaşacak demektir. Örneğin; ağ devre dışı kalmış ise veya uzaktan erişilecek sistem tam anlamı ile fonksiyonel durumda değilse, uygulama devre dışına düşünceye kadar istemci boş olarak bekleyecektir. Zaman uyumlu yapısı nedeniyle etken bir sistem değildir. Eğer iki sistem arasında açık bir bağlantı söz konusu değil ise, bir bilgisayar diğerinin işlem kaynakları serbest hale gelinceye kadar boş durarak beklemek zorundadır (Michah, 2000).

İleti Geçirme Sistemleri (Message Oriented Middleware -MOM)

İleti geçirme sistemleri ortakat yazılım tipleri içinde, kullanıcılar daha önce e-posta kullanmaya alışık olduklarından, kullanıcı yönünden anlaşılması en kolay olan ortakyol yazılımıdır. Bu uygulamada ileti, kaynaklar kullanılmaya hazır olduğunda bir bilgisayardan diğerine iletilir. Bu iletişim şekli zaman uyumsuzdur(asynchronous). Zira bilginin akışı için iki bilgisayarın kullanıma hazır olmasına gerek yoktur. Bu yapı bloklama tipinde de değildir, zira ileti yapılan sistemler arasında her hangi bir noktada bilgi durmaksızın akmaktadır. Bu sistemler ağ bağlantısı için bilgisayarların boş olarak durmasını da engellemektedir (Judith, 2002).

Obje İstemcisi Kılavuzu (Object Request Brokers -ORB)

Obje model, tüm ortamı tanımlamaktadır. Bu tanım içinde objelerin nasıl yaratılacağı tanımlandığı gibi, istemcilerin uygulamalara ulaşmalarını olanaklı kılan ara yüzlerin de nasıl kullanılacağı gösterilmiştir. Obje istemci kılavuzu, bir sunucu uygulamasıdır ve

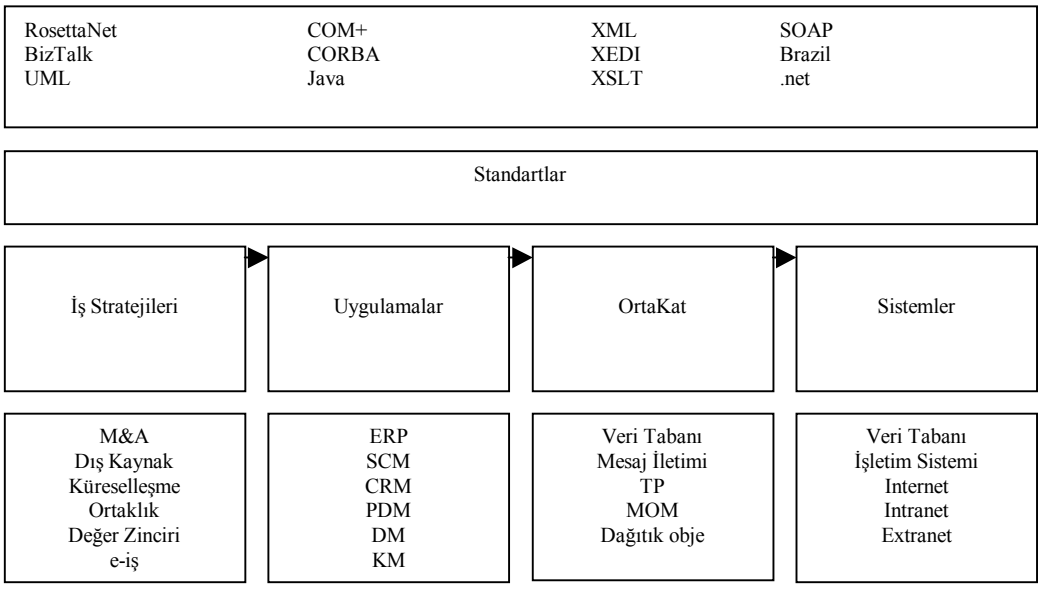
fonksiyonları ağ anahtarlarına iletişim bağlantılarının kurulmasını ve son noktalar arasında bilgilerin gönderilmesini olanaklı kılar. Bir istemci obje ORB den geçerek sunucu objeden bir istemde bulunabilir ve herhangi bir sunucu bunu yine ORB aracılığı ile yanıtlar. ORB istemciler ve sunucular arasında bir kılavuz görevini yerine getirmektedir. İstemciler arayüzlerini objelerin uygulamalarından bağımsız olarak kullanırlar. Bağımsızlık tam bir anahtardır; ORB'ye olan arayüzler ve ORB'yi kullanarak oluşturulan objelere olan arayüzler, mükemmel bir şekilde tanımlanmıştır. Bu tanımlamalar tüm dağıtık ortamı kat edebilen tekrarsız (uniform) bir çerçeve oluşturmaktadırlar. Bu ortamda, istemci uygulaması obje referanslarını kullanarak işlemleri çağırır. ORB, objeleri gerçek olarak yönetmez, o istemci uygulamaları ile objeler arasında iletişimi sağlayan obje referanslarını yönetir. Bu durum hem objelerin nasıl tanımlanacağına ve depolanacağına karar veren uygulama geliştiricisi için, hem de ağ kaynaklarının çok uygun olarak kullanılmasından sorumlu ağ yöneticisi için uygundur (Omg, 2002).

Uygulama Entegrasyonunda Ortakat Mimarisinin Önemi

Günümüzde bilişim teknolojileri yayınlarında sıklıkla CRM, XML ve B2B gibi kavramları kullanmaktayız. Bu kavramlar arasındaki ilişkiler açık bir şekilde uygulama entegrasyonunu ifade etmektedirler. Ortakat mimarisi bu kavramları bütünleştirerek yapıştırıcı görevini görmektedir. Dolayısıyla ortakat mimarisi uygulama entegrasyonun temel öğelerinden birisidir.Uygulama entegrasyonunda ortakat mimarisinin gelişmesini sağlayan başlıca kavramlar iş stratejileri, uygulamalar ve ortakat mimarisinin kendisidir (Gianpaolo, 2001).

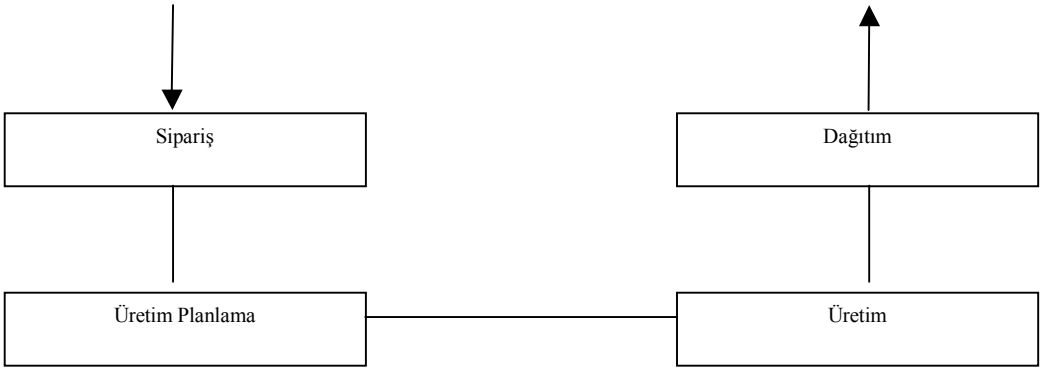
Bu çalışmanın temelinde; küreselleşen dünya ekonomisinde şirketlerin teknoloji, standartlar ve internet kavramlarının bütünleşmesinin sağlanmasıdır. Dünya da yıllık yeni yazılım ve donanım yatırımlarının yaklaşık iki kat kadarda sistem entegrasyonu ve danışmanlık yatırımları söz konusudur. Örnek olarak Procter&Gamble, HP ve Intel firmaları uygulama entegrasyonu sağlamak amacıyla her biri bir milyar dolarlık yatırımlar yapmışlardır (Londergan, 1997).

Ortakat yazılım mimarisi; ne tür uygulamaların ne

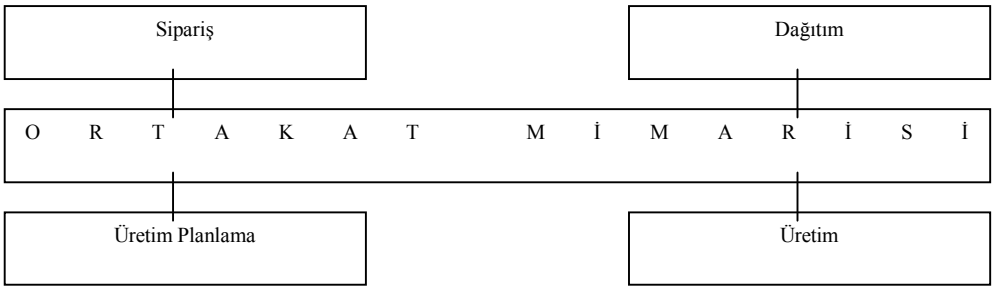


Şekil 1. Uygulama Entegrasyon Zinciri

şekilde entegrasyonun tanımlanmasıdır. Ki bunlar gü-



Şekil 2. Geleneksel Moduler Yapı



Şekil 3. Ortakat Uygulama Bütünleştirme Yapısı

nümüzde B2B, B2C olarak tanımlanmıştır. Dolayısıyla ortakat mimarisinin sistem entegrasyonundaki önemi ortaya çıkmaktadır.

Şekil 1, şekil 2 ve şekil 3 te uygulama entegrasyonun gerçekleştirim zinciri görülmektedir.

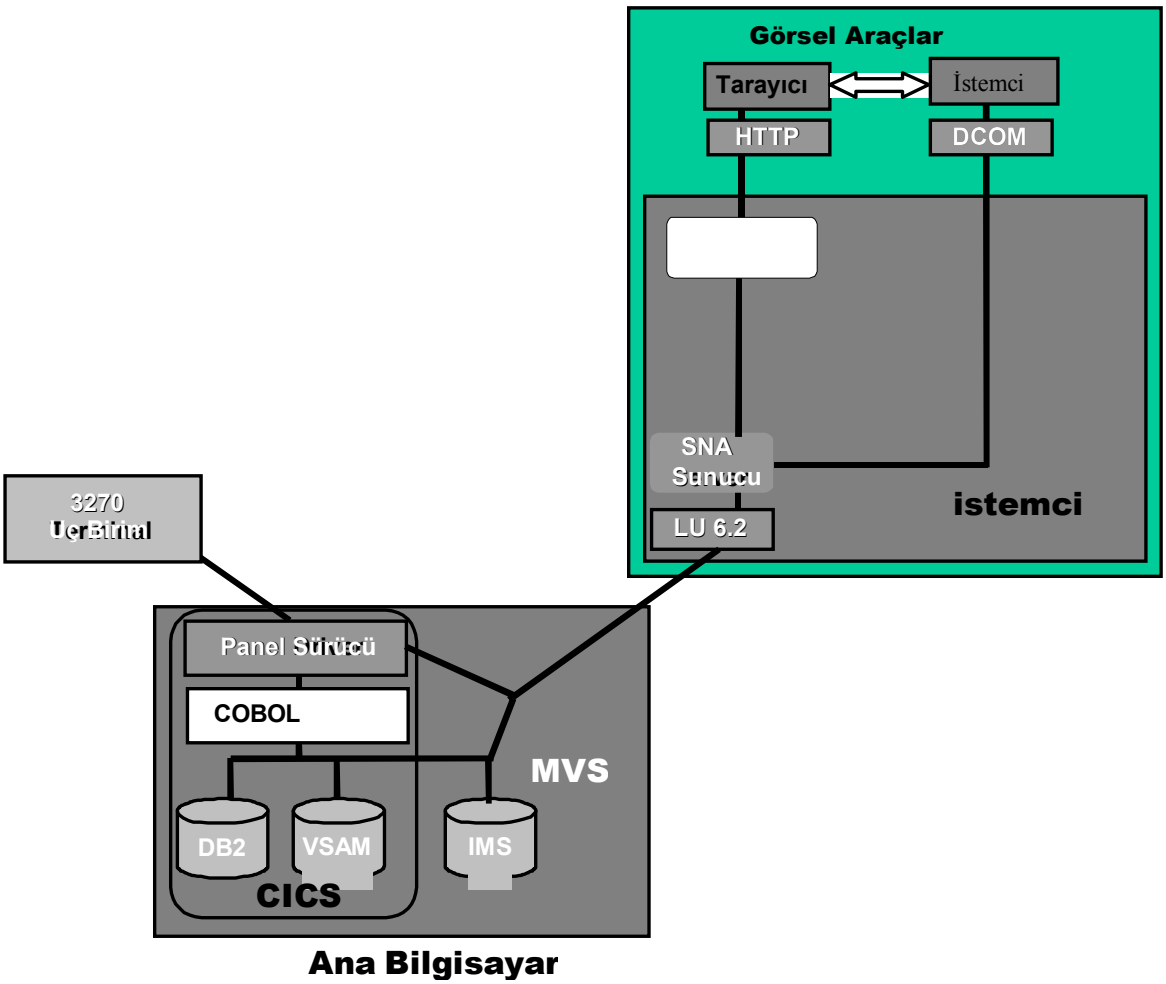
İşletmeler küreselleşen ekonomi sürecinde daha iyi bir yer alabilmesi için temel bilişim teknolojilerinde de yenilikler yapmalıdır. Öncelikle işletmeler uygulama entegrasyonuna geçiş süreçlerini çok iyi bir şekilde tanımlamalıdır. Daha sonra ise bölgeler, sistemler ve veri tabanları üzerinde bilgi akışları tarif edilmelidir.

Geleneksel İstemci/Sunucu yaklaşımında Aşağıdaki diyagram da görüleceği gibi uygulamalar ortakat kullanılmadan direk olarak birbirleriyle iletişim kurmaktadırlar. Bu tür yaklaşım karmaşık sistemlerde problem yaratmaktadır. Uygulamalarda aynı üretici firmaya bağımlı kalınması, uygulama esnekliğinin yok olması ve uygulama diline bağımlı kalınması gibidir.

Ancak ortakat mimarisinde ise; aşağıda görüleceği gibi ara bir katman kullanılarak uygulama entegrasyonu sağlanabilmektedir.

Farklı sistemlerin birbirleriyle konuşabilmesi ve özelliklerinin bilinmesi son derece zor ve kritik bir problemdir. Yukarıdaki şekilde görüleceği üzere; bu görevleri yerine getiren yazılıma ortakat yazılım olarak tanımlanabilir. Uygulama entegrasyonunda ortakat yazılım temel öğedir. Günümüzde nesne tabanlı dağıtık sistemler, giderek klasik istemci-sunucu mimarisine sahip sistemlerin yerini almaktadır. Dağıtık sistem ise ;

- Uygulama bileşenlerinin geniş ağ üzerinde dağılmış olmaları,
- Heterojen yazılım ve donanım yapılarına sahip olmaları,
- Bilinen yazılım parçalarının tekrar tekrar kullanımı,
- Varolan eski sistemlerle entegrasyonu,
- Gelişmekte olan teknoloji ve bugünün uygulamaları, yarının sistem parçaları olacağı için, nesne tabanlı yapılma zorunluluğu,
- Farklı programlama dillerinin kullanımı,
- Dağıtık uygulama bileşenleri arasındaki karşılıklı işlevsel yetenek ve bütünlük,



Şekil 4. Ana bilgisayar verilerine ve programlarına erişim

· Farklı uygulamaların veya parçalarının birbirleriyle ortak kullanımı,

· Heterojen ortamda etkin uygulama geliştirme olarak tanımlayabilmekteyiz.

Nesne yönelim programlama tekniklerinin avantajı olan daha fazla uygulanabilirlik ve daha fazla işlevsellik özelliklerine sahiptirler. Artan bu talep karşısın da bir çok akademik ve ticari kuruluş nesne tabanlı dağıtık sistemler için gerekli altyapıyı oluşturmaya çalışmaktadırlar.

Ana Bilgisayar ile Kişisel Bilgisayar Uygulama Bütünleşme Teknolojileri

Eski sistemlerin istemci/sunucu, intranet ve internet ile başarılı bir şekilde bütünleşmesi için gerekli olan teknolojiler şekil 4 ve şekil 5 te görülmektedir ;

· Uçbirim Öykünme Yazılımları (Terminal Emulation)

Bir istemcinin uç birimmiş gibi ana bilgisayar ile iletişime geçmesini sağlar. Uygulama, pencere, ekran yazdırma ve diğer basit hizmetler gibi grafik kullanıcı arabirimi avantajıyla birlikte uçbirim ekranı gibi görüntülenir.

· Ekran bölümleme bir sonraki bütünleştirme düzeyidir.

Uygulama geliştirme grubu tarafından hızlı uygulama geliştirme ortamı veya daha geleneksel C++ gibi bir dili kullanan grafik arayüzü oluşturmak için kullanılabilir. İşlevsellik, resimler, listeler gibi kolay kullanım ve yeni yetenekler ve ana sistem esaslı verilerin güvenliğini sağlayarak istemci ve ana sistem arasında kolayca bölünebilmektedir. Ekran bölümleme için bir tarayıcının ana sistem ile etkileşim için kullanılabileceği teknolojiler ise ;

· Sistem Ağ Mimarisi kullanıcılara güvenilir ve uygun ana sisteme erişimi sağlamak için eski uygula-

maları ve verileri modern ağ sistemleri ve uygulamaları ile etkin bütünleştirme amaçlı bir temeldir.

· Ana bilgisayarlardaki dosya erişim sistemlerine COM esaslı istemciler üzerinden erişilmelidir.

Uçbirim Öykünmesi

Windows tabanlı istemcilerin ana programlara erişimi gerekmektedir. İstemci ana makinenin uçbirimi gibi davranmaktadır. HTML ile gelişmiş ekran okuma/yazma, intranet veya internet'teki bir istemcinin ana makineye karşı IBM 3270 uçbirimiymiş gibi davranabilmesini sağlar. Windows esaslı bir istemcinin IBM 3270 uçbirimini, IBM S/370 sisteminin istediği kendi uçbirimine öykünmesine izin veren çeşitli programlar vardır.

En basit durumda, Windows esaslı istemciler ana makineye uçbirim öykünmesinin istemci desteği ile bağlıdır. Tümleşmenin en temel biçimi olan uçbirim öykünmesi, istemci yalnızca uçbirim iletişim kurallarına öykünürken arka uçtaki bilgisayarda eski programların %100'ünün çalışıyor olmasını gerektirir.

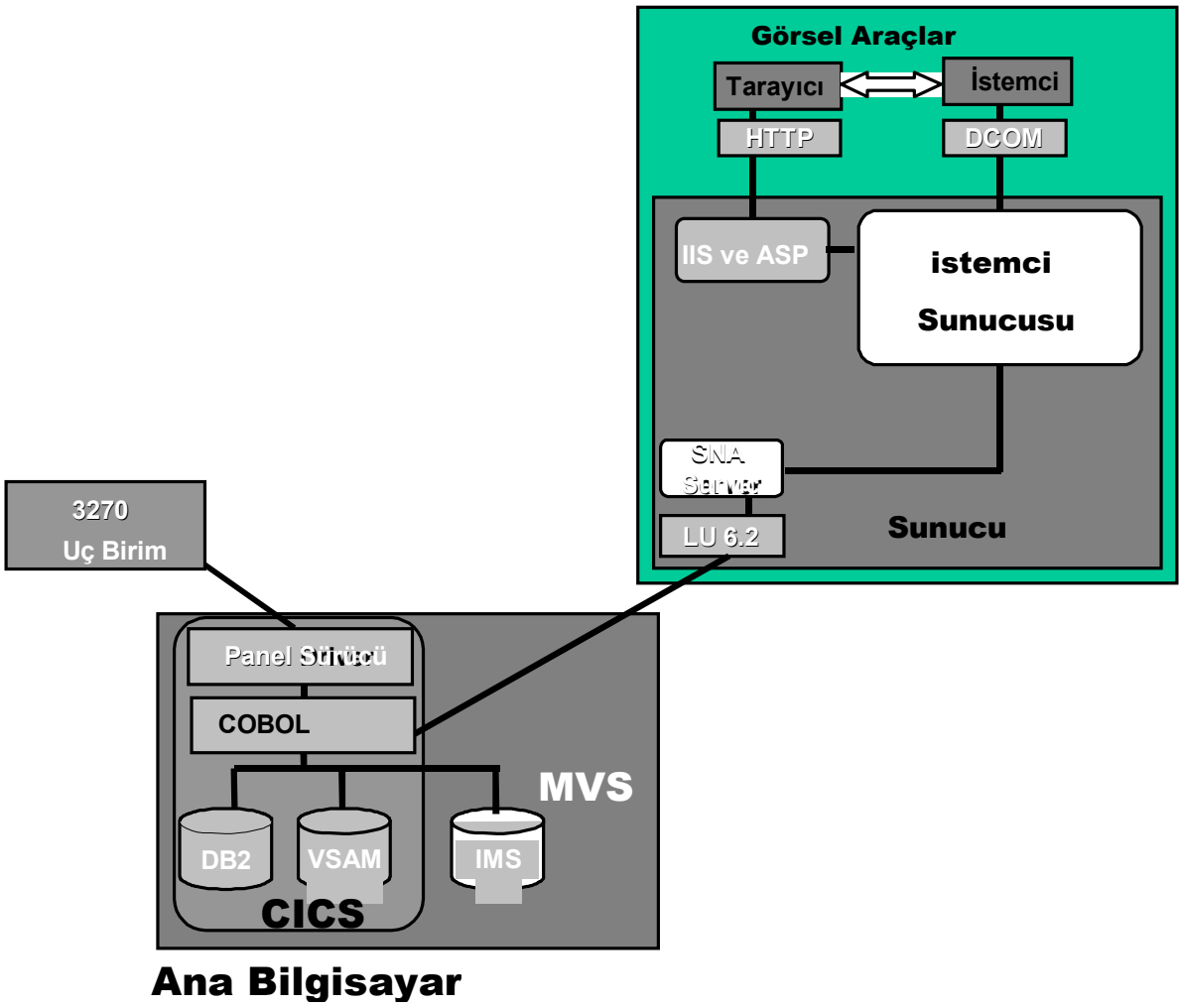
Uçbirim öykünme teknikleri, normal olarak üç yönden birine ayrılarak daha karmaşık yetenekleri desteklemek üzere gelişmiştir. Bunlar;

· İletişim kurallarına özgü kod dilleri:

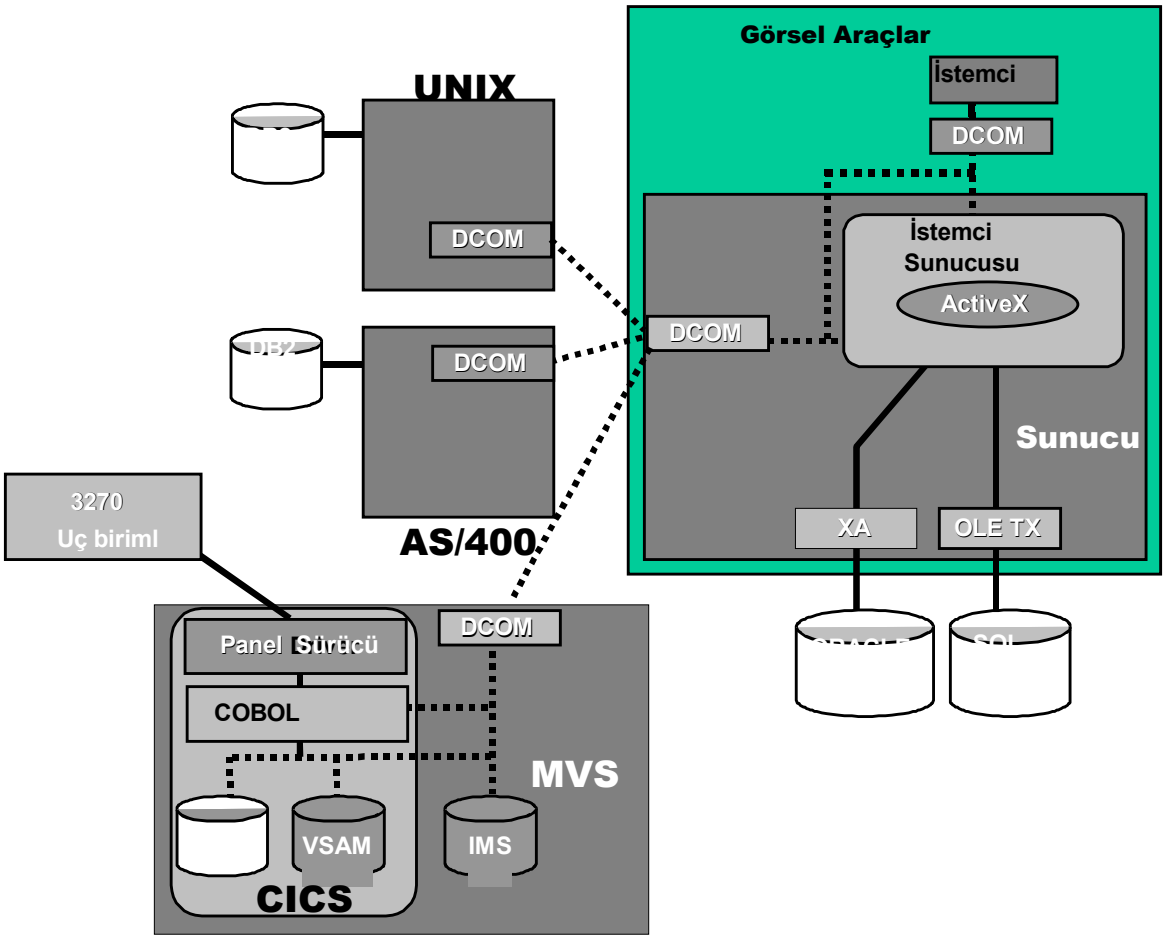
Programcılar, öykünme yazılımları ile birlikte kod dillerini kullanarak, basit uç birim öykünmesinin ötesine geçen denetim ve esneklikler ekleyebilir ve hatta istemci öykünme arayüzünü kendileri yönlendirebilir. Kod dilleri normal olarak, bir dizi komut veya metin gönderir, yanıt bekler ve ekrana gelen çıktıya göre dallanır. Kod dili komutları, performansa karşılık esneklikten vazgeçme anlamına gelen, önceden derleme yerine çalışma sırasında yorumlanır.

· Ekran tarama olarak bilinen ekran okuma ve yazma için program desteği:

Uçbirim öykünmesinin sonraki düzeyi ekran taramasıdır. Ekran tarama, IBM ana makine ile olan bir oturumda, ekrandaki belirli yerlere veri gönderilmesine ve buralardan veri okunmasına izin verir. Ekran taramanın avantajı ise; istemcilerin, yalnızca ekrandaki alanlara veri girişi yapılmasını bekleyen ana programlarla etkileşime girmesini sağlar. Bu istemci



Şekil 5. İşlemci Sunucusu



programı, örneğin Visual Basic ile kendi kullanıcı arayüzünü oluşturabilir ve ekran taramayı kullanarak ana verileri, ana programcı tarafından ilk düşünüldüğü biçiminden tamamen ayrı, grafik biçimde kullanıcıya sunabilmektedir.

· Daha yüksek düzeyli nesne ve uygulama programı arayüzleri için destek:

Uçbirim öykünme ürünleri, normal olarak arabelleklere, komut argümanlarına ve diğerlerine erişmek için C kullanabilen bir programcının kullandığı lisans arayüzü ile birlikte verilmektedir. Diğer taraftan gittikçe artan sayıda araç ve geliştirici ürünü, eski verilere ulaşmak üzere yüksek düzey hizmetlerin istemci tarafından yürütülmesi için nesneye dayalı arayüzlerle birlikte verilmektedir. Önceden varolan eski veritabanlarına bağlantı oluşturmak için iletişim kurallarının üstüne kat olarak konulmuş olan açık veri tabanı bağlanabilirliğine (ODBC) API, uygulama geliştiricisi tarafından kolaylıkla erişilebilmektedir.

Platformlar Arası Çok Katlı Uygulamalar için Bütünleşme

Kurumlar platformlar arası (Windows, UNIX, MVS) dağıtılmış uygulamalara gereksinim duymaktadırlar.

Bunun anlamı;

· Parametrelerin birden çok platformda çalışan değişik programlama dilleri arasında iletilmesi,

· Değişik işletim sistemlerinde çalışan farklı veritabanı sistemleri arasında iki aşamalı üstlenmeler gibi daha karmaşık işlemleri yönetmek,

· Kaynak kodunun okunmasının zor olabileceği veya bulunamadığı eski uygulamaları zenginleştirme,

Günümüzde birçok veri merkezi, basit veri erişiminden veya kod denetimi sonuçları vermekten daha fazlasını yapmak üzere bir ana bilgisayar işlem programı erişimine gereksinimi olan istemci uygulamalarını çalıştırmaktadır. Ana bilgisayar ortamındaki dağı-

tılmış işlem süreci için fiili standart LU 6.2 iken ve hem IBM İstemci Bilgi Denetim Sistemi (CICS), hem de IBM Bilgi Yönetim Sistemi (IMS) tarafından kullanılmasına karşın şirketler daha karmaşık bir ortama doğru gelişmişlerdir.

SNA Sunucusu ile her kişisel bilgisayar bir ya da daha fazla Microsoft SNA Server 3.0 bilgisayarına bağlanmak için TCP/IP, IPX/SPX, NetBEUI, Banyan VINES IP veya AppleTalk gibi standart yerel ağ iletişim kuralı (LAN) kullanır. Bu Windows NT tabanlı sistemler de buna karşılık ana bilgisayara ve AS/400 sistemlerine IBM SNA iletişim kurallarını kullanarak bağlanırlar. Kolay kurulum ve merkezi grafik yönetim için araçlar sağlandığı gibi tüm büyük kişisel bilgisayar ve ağ işletim sistemleri, yerel ağ türleri SNA ana bilgisayarları ve bağlantı türleri de desteklenir. SNA Sunucusu istemcilerin ana bilgisayar ile etkin iletişim kurmasını sağlamak için iletişim sürecini ana bilgisayar ve masaüstü kişisel bilgisayardan Windows NT sunucusuna yüklemektedir.

İşlemci sunucusu(Transaction Server), nesne yöntem çağrılarını yakalar ve bu çağrıları uygun ana bilgisayar programına yönlendirir; ayrıca ana bilgisayardan gelen tüm çıktı parametrelerini ve yanıt değerlerini de yönetir. İşlemci sunucusu, yöntem çağrısını yakaladığında yöntemin parametrelerini Windows NT tarafından anlaşılan gösterimlerden ana bilgisayar programlarının anladığı gösterimlere dönüştürür. İşlemci sunucusu, işlemlerin tümü Windows NT sunucusu üzerinde tamamlanır ve ana bilgisayar yüklemesi olmaz. İşlemci sunucusu Windows NT ve ana bilgisayar arasındaki iletişim için LU6.2 gibi standart iletişim kuralları kullanır.

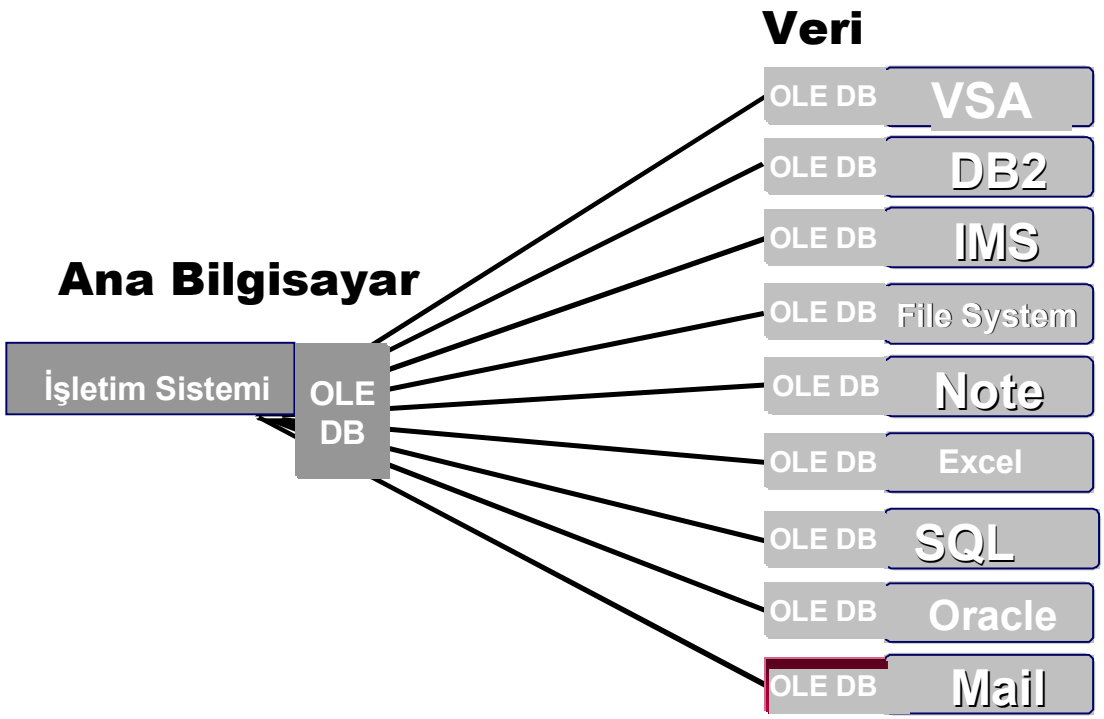
Şekil 5.'te istemci sunucusu yapılandırması gösterilmiştir. İstemci uygulaması oluşturmak için Visual Basic ve Visual C++ gibi görsel araçları kullanılabilmektedir. Bir tarayıcıdan da yürütülebilen bu uygulama, ana bilgisayarda çalışan bir işleme erişmek için işlemci sunucusunu kullanmaktadır. Bu işlemci sunucusu, birden çok iş parçacıkları oluşturma, geliştirilmiş performans için nesne yönetimi ve yük dengeleme dahil olmak üzere belirli özellikleri kullanmaktadır. İstemci uygulaması DCOM' u destekleyen herhangi bir platformda çalışabilir. Bu çözüm ana bilgisayarda DCOM olmasını gerektirmez. DCOM dile bağımlı olmadığı için geliştiriciler, aşağıdakiler dahil olmak üzere, projeye en uygun olan bir dili ve araçları kullanarak bir istemci uygulaması oluşturabilirler. Bu diller Visual Basic, Visual C++, Visual J++, Borland Delphi, Powersoft Powerbuilder ve Micro Focus Object Cobol olabilir. Bu istemci, bundan sonra işlemci sunucusu çalıştıran bir Windows NT sunucusunda bileşenleri kolaylıkla yürütebilir. Arayüz yazılımı geliştirme aracı ne olursa olsun, geliştiricinin izlemesi gereken beş adım vardır;

- Uygulama yöntemlerini ve parametrelerini tanımlanması,
- Uygulamayı yazmak,
- Uygulamayı sınamak,
- Uygulama bileşenlerinin ilgili kütüphanelere yerleştirmek,
- Uygulama bakımıdır.

Ana Bilgisayar ve DCOM

Şekil 6 'da görüldüğü gibi DCOM, değişik bilgisayarlardaki yerel ağdaki, geniş ağdaki veya Internet'teki nesneler arasındaki iletişimi desteklemek üzere COM'u genişletir. DCOM ile bir uygulama, en fazla mantıklı gelen yerlere dağıtılabilir (Emerald 2000).

DCOM'un, COM'un eksiksiz bir evrimi olması nedeniyle şirketlerin, standartlara dayanan dağıtılmış bilgisayar işlemleri dünyasına geçmeleri için COM esaslı uygulamalar, bileşenler, araçlar ve bilgiler için yapmış oldukları mevcut yatırımlarından yararlanabilirler. DCOM ağ iletişim kuralları düşük düzey ayrıntılarını yöneterek, geliştiricilerin proje ve iş kurallarını oluşturmak üzere yoğunlaşabilmeleri için zaman tanımış olur. Geliştiricinin mevcut ana bilgisayar programları için bir COM yardımcı programı yazmasını gerektirir. Bu çözüm Dağıtılmış Bilgisayar İşlemi Ortamı'nın ana bilgisayarda olmasını gerektirmez. İstemci uygulaması birçok dilden birinde yazılabilir. COM şartnamesi, nesnelerin birbirlerini bulup iletişim kurması için bir yol tanımlamaktadır. Dağıtılmış nesnelerin ve bileşenlerin yerleri istemciler için saydamdır. COM başlatıldıktan sonra gereken sunucuyu tanımlayan bir örnek oluşturulur. İstemciler ve sunucular uzak yordam çağrısı (RPC) kullanarak iletişim kurarlar. Aktarım için RPC gerekir çünkü hem sunucu hem de istemci kendi işlemlerinde nesneler içerirler. İstemcisinde, uzak bileşen iletişimine, istisna ve geri arama işlemine ve birden çok iletişim kuralına izin vermek üzere DCOM'un çok iş parçalı yetenekleri vardır.



Şekil 7. Object linking and embedding (OLE)

İstek DCOM tarafından alındıktan sonra, ana makinede başlatılan hizmetleri yöneten Service Control Manager'a (SCM) iletilir. Bu, tüm sınıfları, programları ve çalışmakta olan hizmetleri kapsar. Gereken nesne çalışmıyorsa, o zaman, uzak ya da yerel olsun, SCM bunu bulur ve çalıştırmaktadır.

Platformlar Arası Kayıt Erişimi

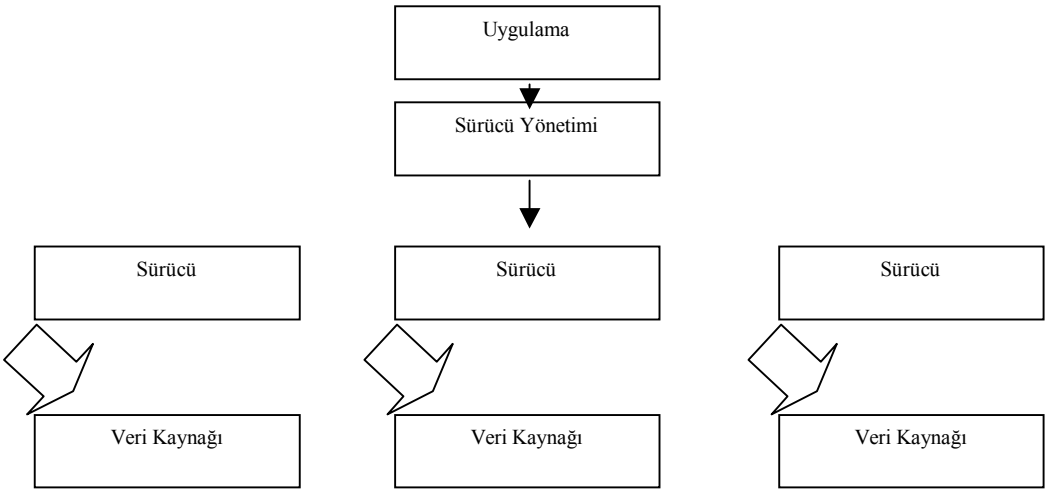
Kurum, verilerinin birçoğu halen kendi erişim yöntemleri veya IBM sistemlerde ise VSAM(Virtual Storage Access Method) biçiminde depolanmaktadırlar. Bunun sonucu olarak, gerçek bir tümleştirme stratejisinin, intranet'te bu bilgilere erişilebilir ve kullanılabilir olması için bir yol sağlaması gerektiği gibi internet için de sağlaması gerekir. İstemcilerin bu tür yetenek istemelerinin nedenleri arasında şunlar sayılabilir (Charles 2001):

- VSAM dosyalarına Web esaslı erişim,
- Hızlı kayıt ve dosya yükleme ile karar destek sistemleri,
- Windows NT'den ana bilgisayara yedekleme,
- VSAM'den SQL Sunucusu'na toplu veri kopyalama,
- Üç kademeli geliştirme,
- Ana bilgisayar esaslı veriler,
- Sunucu tabanlı iş kuralları,
- Küçük istemciler,

Kişisel bilgisayar açısından, ilişkisiz ana bilgisayar verilerine ulaşmak için birden çok platformdaki çok sayıdaki dosya türüne erişim için tam ve tek bir çözüm getirmek üzere OLE DB kullanılmaktadır. Şekil 7 de OLE görülmektedir. Buna uygulama olarak yapılan çalışmada Domino sunucusunda bulunan verilerin Microsoft Word ve Microsoft Excel ürünlerine bağlantısı sağlanmış ve Domino sunucundaki veri tabanı masa üstüne getirilerek kullanıcılara raporlama ve sorgulama olanakları getirilmiştir.

ODBC Open Database Connectivitiy

ODBC; bilgisayar ağları içerisinde bilgisayarların istemci veya sunucu olmasına bakılmaksızın verilerin paylaşımı, uygulamaların paylaşımı için geliştirilmiş alt düzeyli uygulama programı ara yüzleridir. ODBC ara yüzleri, veri tabanı yönetim sistemlerindeki verilere de erişimi sağlanmaktadır. Uygulama programcıları veri tabanı yönetim sistemlerini göz önüne almaksızın sistemlerini geliştirmektedirler. Daha sonradan yazılan çeşitli modüller uygulama programı ile veri tabanı yönetim sistemleri arasında veri değişimi sağlanmaktadır. Bu modüllere sürücü (drivers) adı verilmektedir. Bu yöntem ile uygulama programları veri



Şekil 8. ODBC Yönetimi

tabanı yönetim sistemlerine bağımlı kalmaksızın geliştirilmiş olmaktadır.

ODBC ara yüzleri aşağıdaki gibi tanımlanmaktadır;

· ODBC işlevlerini yerine getiren kütüphanenin oluşturularak, SQL komutları ile veri tabanı yönetim sistemlerindeki verilere uygulama programları aracılığıyla erişim sağlanmalıdır,

· Veri tabanı yönetim sistemlerine bağlantı ve kullanımın uygulama programından sağlanmalıdır.

· Veri yapıları standart olarak tanımlanmalıdır.

ODBC sürücüleri ile esnek uygulama gelişimi sağlanmıştır. Yani; uygulama programlarındaki obje kod değiştirilmeden farklı veri tabanı yönetim sistemlerine erişim sağlanmaktadır. Şekil 8 de gösterilen ODBC mimarisi dört bileşenden oluşmaktadır. Bunlar ;

· Uygulama (Application) (Spreadsheet, Word Processor, Uygulama geliştirme araçları)

· Sürücü Yönetimi (Driver Manager)

· Sürücü (Driver)

· Veri Kaynağı (Data Source)

Buna uygulama olarak yapılan çalışmada; ODBC arayüzleri kullanılarak Lotus Notes ile SQL Sunucusu arasında çeşitli uygulamalar geliştirilmiştir. Bu uygulamalar ile Lotus Notes tan SQL arasında akıllı arayüzler tanımlanarak SQL veri tabanı yönetim dizgesindeki verilere direk erişim sağlanmıştır. Farklı işletim sistemlerinde de uygulamalar geliştirilmiş ve ana bilgisayardaki verilerin masaüstüne getirilmesi sağlanmıştır.

Sistem Bütünleştirme Teknolojilerinin Sağladığı Avantajlar

Eski uygulamaların genişletilerek ve masaüstü avantajlarıyla birlikte varolan programlama bilgisini de kullanarak kurumlara ciddi maliyet avantajları sağlanacaktır. Bunlar;

· Yeni teknolojileri işletme : Rekabet avantajı oluşturmak için en az iş kaybı ile daha hızlı çözümler getiren daha iyi alt yapının avantajlarından yararlanırken birbirinden farklı sistemler arasındaki ayrımları azaltır.

· Eski yatırımları koruma : Bütünleştirme yoluyla, eski sistemler daha esnek ve güçlü sistemler içine dahil edilmiş halde çalışmaya devam edebilirler. Günümüzde, verilere ve eski sistemlere, ağa bağlanmış özel uygulamalar ve tarayıcılarla çalışan bilgisayarlarla erişmek kaçınılmazdır. Bu yaklaşım, kurumlara daha düzenli ve daha az maliyetli çözüm olanakları sağlamıştır.

· Riski azaltma : Bütünleştirme yıllarca sürebilecek taşıma stratejisinin desteklenmesine olanak tanır. Taşıma işlemini tek bir büyük projede tamamlamak yerine, bileşenleri kullanarak adım adım gelişen bir yaklaşım kullanmak riski azaltır ve başarıyı garantilemektedir. Erişim mimarisi bir kere oluşturulduğunda, dizisel süreç oluşturmak için gereksinimlerin önceliği belirlenir, böylece her adım tamamlandığında kuruluşa yeni değerler eklenir. Geçerli sistem hala etkin olduğundan, kuruluş üzerindeki etkisini azaltan bir geri çekilme mümkündür.

· Destek ve eğitim harcamalarını azaltma : Ana bilgisayar programcılarından oluşmuş çalışanlarla, bilgi sistemi avantajlı olarak yönetilebilir. Bu programcı grupları yüksek kaliteli yazılımlar oluşturabilirler ve yetenekleri (kod bakımı, iş bilgisi, yapılanmış metodolojisi ve kaliteye önem vermeleri vb.) oldukça değerlidir. Programcıların tümü istemci/sunucu veya Web geliştirmeye geçmek istemezken, bütünleşme kuruluşlara geliştiricilerin yeteneklerini yüksek kaliteli desteğin devamını sağlayarak maliyetleri düşürmesine olanak tanımaktadır.

· Şirket üretkenliğini artırma : Bu stratejiyi kullanarak son kullanıcılar ve geliştiriciler, birden çok platform aralarında hiç boşluk olmadan birlikte çalışabilecekleri için daha verimli olacaklardır. Geliştiriciler, istemci ve sunucudan bağımsız olarak en iyi çözümü uygulamak için teknolojiyi kullanırken, son kullanıcılar yerden bağımsız bilgilere ulaşmak için tutarlı grafik ara yüzüyle çalışabilmektedirler.

Kaynaklar

1. BERNSTEIN PA, Middleware: A model for Distributed Service Communications Of ACM 1996
2. CHARLES P, Programming Windows, The definitive Guide to the Win32 API, Microsoft Pres, 2001
3. DEBBIE L, STEVE KD, Lotus Notes and Domino 5 Development Unleashed,, Addison Wesley 2001
4. ECKERSON, WW, Three Tier Client/Server Architecture: Achieving Scalability, Performance and Efficiency in Clinet Server Application, Open Information System 1995
5. EMERALD, DCOM and CORBA Side by Side, Step by Step, and Layer by Layer, Journal, 2000
6. GIANPAOLO R, Role of Middleware in Application Integration, Helsinki University of Technology,2001
7. G.WINFIELD T, LAWRANCE C. S, Designing Systems For Internet Commerce, Addison Wesley, 1999
8. GÜMÜŞKAYA H, DAVULCU O, AKBAŞ E, DURSUN T, A Corba and Java Mobile Agents Based Network Management Architecture, ISCIS XIV, S.975-982, 1999 Kuşadası Turkey
9. JUDITH M.M, The Complete Book of Middleware, Auerbach, 2002
10. LONDERGAN S, The Lotus Domino Server Second Edition, M&T Books,1997
11. MICHAH L, GEORGE V, NINO V, DADO V, Middleware Networks: Concept, Design and Deployment of Internet Infrastructure, 2000
12. OMG (Object Management Group), The Common Object Request Broker : Architecture and Specification, 2002