Bu tez çalışması, Maltepe Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ..... / ..... / ..... tarih ve ..... / ..... sayılı kararıyla oluşturulan jüri tarafından ……. ***……. Yüksek Lisansı Tezi*** olarak kabul edilmiştir.

JÜRİ

Prof. Dr. Kemal KÖYMEN

Danışman

Prof. Dr. E. Murat ESİN

Üye

Yrd. Doç. Dr. Turgay BİLGİN

Üye

T.C.

MALTEPE ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Microsoft Dynamics Ax platformunda Kurumsal Kaynak Planlama programlarının geliştirme yöntemlerinin değerlendirmesi

Bilgisayar Mühendisliği Anabilim Dalı

YÜKSEK LİSANS TEZİ

HAZIRLAYAN

Alper ŞAMDANCIOĞLU

Danışman

Prof. Dr. Kemal Köymen

İSTANBUL

TEMMUZ 2009

Özet

Microsoft Dynamics Ax platformunda Kurumsal Kaynak Planlama programlarının geliştirme yöntemlerinin değerlendirmesi isimini taşıyan bu yüksek lisans tezi, Maltepe Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği Anabilimdalı bölümünde yaptığım yüksek lisansın tez aşaması için 2009 yılında hazırlanmıştır. Tezimi sunduğum bu dökümanın uzunluğu 100 sayfadır.

Anahtar kelimelerim: Kurumsal Kaynak Planlaması, bilgisayar yazılımları, yazılım mimarisi, yazılım programları, Dynamics, Axapta

Yazımını gerçekleştirdiğim tezin giriş kısmını Kurumsal Kaynak Planlama'nın (KKP) doğumu, KKP' nin günümüzdeki şeklini almasında yardımcı olan etkenler ve KKP nın doğumuna sabep olan diğer kavramların açıklamalarından oluşmaktadır. Gelişme kısmında ise en çok tercih edilen üç adet yazılımın kısa tanıtımı ve ardından bu programların birbirleri ile geliştirme altyapısı göz önünde bulundurularak sağladıkları üstünlüklerin karşılaştırılmalarından oluşmaktadır. Sonuç kısmında seçtiğimiz yazılımın donanımsal ve yazılımsal olarak detaylı tanıtımları yer almaktadır.

Abstract

Evaluation of Enterprise Resource Planing development on Microsoft Dynamics AX platform is the theme of this thesis. This thesis is written in the year of 2009 for the graduate degree education in T.C Maltepe University Life Sciences Institute Computer Scinece Department. This document which i present my thesis is 100 pages long.

Keywords : Enterprise resource planning, computer softwares, software architecture, software programs, Dynamics, Axapta

The introduction part of my thesis includes, the begining phase of ERP, the facts that affects the process to be build in this current situation and explanations of the other concepts that came reason for ERP's begining phase. Progress part of the thesis includes top three software packages and detailed explanations of these three software packages after the completion of the introduction the compare section starts. In comparision the main consider is development platform. The final section is conclusion part the detailed explanation of the chosen software is explained.

Teşekkür

Öncelikle bu konuda araştırma yapmam için beni teşvik eden ve çalışmanın hazırlanmasında benden maddi ve manevi desteğini esirgemeyen saygıdeğer hocam Prof. Dr. Kemal Köymen'e, bana yüksek lisans eğitimim esnasında bilgileri ile beni aydınlatan saygıdeğer hocalarıma sonsuz teşekkürlerimi arzederim.

Alper ŞAMDANCIOĞLU

İÇİNDEKİLER

ÖNSÖZ

Kurumsal kaynak planlama yazılımları, günümüz teknoloji ve koşullarında en hızla gelişmesi gereken ve firmanın iç süreçlerini destekleyen programlardır. AMR Research'ün yaptığı araştırmalara göre kurumsal kaynak planlama pazarında lider olan beş adet yazılımdan biri de Microsoft'un sahip olduğu Microsoft Dynamics Ax'tır.

Microsoft Dynamics Ax gerek firmaların süreçlerine uygulanmasının gerekse geliştirme alt yapısının kolaylığı açısından tercih edilen bir uygulamadır. ERP sistemlerinden çok azı bu kadar çok yönlü bir aracı temin edebilir. Ax'ın geliştirme dili olan X++, radikal olarak zengin ve çok yönlü fonksiyonaliteleri mümkün kılmakta ve en az kod sayısı ile geliştirmelerin yapılabilmesini sağlamaktadır. Daha az kod demek daha az risk, daha çok performans demektir ve obje tabanlılık geliştirme görevlerinin kolay ve hızlı yapılmasını arttırmaktadır. Açık kaynak kodlama özelliği ile değişime ayak uydurmak, firmaya özel süreçlerin efektif bir şekilde yazılım üzerinde takibini sağlamak verimliliği de oldukça arttırmaktadır.

Tez içerisinde kurumsal kaynak planlama programlarının kullanım nedenleri ve Microsoft Dynamics Ax programının tercih nedenlerinden biri olan zengin geliştirme alt yapısı incelenerek uygulama metodolojisine yer verilmiştir.

1 KURUMSAL KAYNAK PLANLAMA

Bu bölüm başlığı altında kurumsal kaynak planlamanın amacı, gelişim evresi ve kurumsal kaynak planlama yazılımlarının ortak özellikleri ele alınacaktır.

1.1 Kurumsal Kaynak Planlamaya Giriş

Kurumsal Kaynak Planlaması (ERP - Enterprise Resource Planing), işletmelerde mal ve hizmet üretimi için gereken işgücü, makine, malzeme gibi kaynakların verimli bir şekilde kullanılmasını sağlamayı temel alarak geliştirilmiş ve işletme içi faklı fonksiyonların (üretim, muhasebe, bilgi işlem, lojistik v.b.g.) birbirleri ile bütünleşik olarak çalışarak karar destek sistemlerini destekleyen ve günümüz rekabet ortamında farklılık sağlayan öğelerden biri olan yönetim sistemleridir. Kurumsal kaynak planlama bir program değil firmanın vizyonu, firmanın uyguladığı bir süreçtir.

Kurumsal Kaynak Planlama yazılımları ise firmaların bu iç süreçlerini yönetebilmeleri için kullandıkları, veri girişi ile veri akışı ve raporlamanın sağlandığı birer karar destek aracıdır.

Kurumsal kaynak planlama süreçlerinin ve yazılımlarının günümüzde sıklıkla kullanılır hale gelmesi, firmaların değişim ihtiyacı ve günü yakalama/rekabet edebilme içgüdülerine dayanmaktadır.

Günümüzde de rekabet koşulları ve teknoloji çağında olduğumuz göz önüne alınarak kurumsal kaynak planlama yazılımları kendi içerilerinde hızlıca kendilerini yenilemektedir. Evrim teorisinde denilir ki; " Doğada en güçlü olan değil, değişen koşullara en iyi uyum sağlayan yaşar." İş hayatının doğası gereği yeni değişkenler ortaya çıktıkça kurumsal kaynak planlama sistemlerinin de çağa ve teknolojiye ayak uydurması kaçınılmazdır.

1.2 Kurumsal Kaynak Planlama Yazılımlarının Gelişimi

Kurumsal kaynak planlama yazılımlarının özellikle 1900'lü yılların başından itibaren bilişim sektöründe artan oranlarda etkisini hissettirmektedir. Bununla birlikte temelini 1960'lı yıllarda geliştirilen stok kontrol yazılımları oluşturmaktadır. [1]

1.2.1 Stok Kontrol Yazılımları

Üretim ile ilgili ilk uygulamalar temeli muhasebeye dayanan stok kontrol ve satınalma yazılımlarıdır. Muhasebe yazılımları bilgisayar ortamında geliştirilen ilk yazılımlar olup, üretim ile ilgili ilk uygulamalar muhasebecilerin stok değerlerini bilme istekleri ile geliştirilmiştir.

1.2.2 Malzeme Gereksinim Planlama Yazılımları

1960'larda kullanılmaya başlayan Malzeme Gereksinim Planlama yazılımları stok kontrol yazılımlarındaki eksikliklere bağlı olarak geliştirilmiştir. Stok kontrolü ve üretim planlamasını destekleyen yazılımları içerir. Bu tür yazılımlar Kurumsal Kaynak Planlaması yazılımlarının kilometre taşlarındandır.

Malzeme Gereksinim Planlama ile ilgili çalışmaları ilk başlatan kişiler Joseph Orlicky, George Plossl ve Oliver Wigh'tir. Bunun yanında APICS(American Production and Inventory Control Society – Amerikan Üretim ve Stok Kontrol Kurumu) 'in de bu konuda önemli katkıları olmuştur. 1950 ve 1960'larda bilgisayarların ucuzlaması ile birlikte Malzeme gereksinim Planlama yazılımlarını kullanan firma sayısı önemli ölçüde artış göstermiştir.

Bu yazılımların amacı doğru yerde doğru zamanda doğru malzemenin bulunmasının sağlanmasıdır.

1.2.3 Kapalı Döngü Malzeme Gereksinim Planlama Yazılımları

Günümüzde değişimin sürekli ve dinamik olduğu göz önüne alınmalıdır. Değişimin sonuçlarından etkilenmemek için diğer kullanılması gereken yöntem ise kapasite planlamasıdır. Malzeme Gereksinim Planlamasında bu araçlar olmasına rağmen yeterli değildir ve yeni araçların geliştirilmesi gerekmektedir. Geliştirilmesi gereken araçlar 4 başlık altında incelenmektedir. Bu başlıklar; satış ve üretim düzeylerinin planlanması, ana planın oluşturulması, talep yönetiminin gerçekleştirilmesi ve kaynak ihtiyaçlarının belirlenmesidir.

Etkin bir kapalı döngü malzeme gereksinim planlama sistemine sahip olan kurumlar, beklenmeyen değişikliklerin etkisini kısa sürede belirleyerek söz konusu değişikliklere uyum sağlamakta ve kendini değişen müşteri talebine göre konumlandırmaktadır.

1.2.4 Üretim Kaynakları Planlama Yazılımları (MRPII)

Kapalı Döngü Malzeme Gereksinim Planlama sistemlerine ek olarak aşağıdaki fonksiyonları içermektedir.

- Satış ve Üretim Planlama – Talep ve tedarik hacimlerini destekler. Bu nedenle üst yönetimin işlemler üzerinde kontrolünü arttırır.
- Finansal Ara yüz – Planlama bölümü birimleri finansal terimlere çevirmekte kullanır.
- Simülasyon – "Ne – eğer" sorularının sorulmasını ve cevaplandırılmasını sağlar.

İşletme planı, satış planı, üretim planı, ana plan, malzeme gereksinim planı, kapasite gereksinim planı, kapasite ve malzemelerin planlanması için destek sistemlerini içermektedir.

1.2.5 Kurumsal Kaynak Planlama Yazılımları

Kurumsal Kaynak Planlama yazılımları; üretim kaynaklı planlama yazılımlarından sonra ortaya çıkan, sadece üretim için kullanılan kaynakların yanında kurum için gerekli tüm kaynakların planlanmasını içeren sistemlerdir. Kurumsal kaynak planlama yazılımlarının gelişimini tetikleyen en önemli nedenlerden biri teknolojinin gelişimidir.

Kurumsal Kaynak Planlama yazılımlarını entegre olmayan sistemlere kıyasla iki önemli avantajı şu şekilde sıralanabilir.

1) Tüm fonksiyon ve departmanları birleştirerek kurum bazında bir görünüm sunması, ve
2) Tüm faaliyetlerin girildiği, kaydedildiği, işlendiği, kontrol edildiği ve raporlandığı bir veritabanı sunmak olarak belirtmektedir.

Kurumsal Kaynak Planlama yazılımları ve geleneksel yazılım geliştirmenin farklılıklarını 3 başlık altında incelemektedir:

1)Süreç ve prosedürlerde değişiklik gerçekleştirilebilir

2)Kurumsal Kaynak Planlama yazılımının kurumda kullanılması için değişiklikler yapılabilir.

3)Kurumsal Kaynak Planlama yazılım satın aldığı firmaya destek ve güncellemeler için bağımlı hale gelir.

Kurumsal kaynak planlama yazılımlarının üretim kaynakları planlamasından teknik farklılıkları grafiksel ara yüz, ilişkisel veri tabanı, yazılım geliştirmede 4. Nesil dil ve bilgisayar destekli yazılım mühendisliği kullanımı, istemci – sunucu mimari ve açık sistem olarak belirtilmektedir.

1.3 Kurumsal Kaynak Planlama Sistemlerinin Değerlendirilmesi

Kurumsal kaynak planlama yazılımlarına geçme nedenleri aşağıda yer alan tabloda açıklanmaktadır. Firmaların ihtiyaçları doğrultusunda temel eksikliklerini gidermek amacıyla firmalar gün geçtikçe bu programlara eğilim göstermeye devam etmektedirler.

Tablo 1.1: Kurumsal Kaynak Planlama Sistemlerinin Değerlendirilmesi

	Kurumsal Kaynak Planlama Sistemlerinden Önce	**Kurumsal Kaynak Planlama Sistemlerinden Sonra**
Bilgi Sistemleri	Birbirinden ayrı sistemler	Entegre Sistemler
Koordinasyon	İşletme fonksiyonları arasında koordinasyon eksikliği	İşletme fonksiyonları arasında koordinasyon oluşturur
Veri Tabanları	Veri entegrasyonu olmaması nedeniyle aynı verinin farklı tanımlanması	Veri entegrasyonu; farklı fonksiyonlar arasında veri tutarlılığı

Bakım	Sistemlerin bakımı parça parça gerçekleştirilir; tutarsızlıklar oluşturur; farklı sistemlerin bakımı maliyeti arttırır.	Bakım süreci tektir; değişiklikler birçok sistemi etkilemektedir.
Ara yüzler	Farklı sistemler arasındaki ara yüzlerin yönetimi zordur	Sistemler arasında ortak ara yüzler
Bilgi	Tutarlı olmayan bilgi	Tutarlı gerçek zamanlı bilgi
Sistem Altyapısı	Güncel olmayabilir.	İstemci – sunucu modeli
Süreçler	Birbiri ile uyumsuz süreçler	Bilgi modeline dayalı birbiri ile uyumlu süreçler
Uygulamalar	Farklı uygulamalar	Tek bir uygulama

Kurumsal Kaynak Planlama yazılımlarının sahip olması gereken özellikler aşağıdaki şekilde özetlenebilir.

- Kurumsal kaynak planlama sistemleri istemci –sunucu ortamlar için tasarlanmış paket yazılımlardır. Kurumsal kaynak planlama yazılımlarının internet ortamında gerçekleştirilmesi ile ilgili bir sınırlama yoktur.
- Kurumsal kaynak planlama sistemleri bir kurumdaki bir çok süreci entegre etmektedir.
- Kurumsal kaynak planlama sistemleri bir kurumda gerçekleşen faaliyetleri büyük oranda desteklemektedir.
- Kurumsal kaynak planlama sistemleri kurum bazında bir veri tabanı kullanır ve veri sisteme sadece bir kere girilir.
- Kurumsal kaynak planlama yazılımları verilere gerçek zamanlı erişimi sağlar.

- Kurumsal kaynak planlama sistemleri kurum bazında gerçekleşen faaliyetlerin planlama ile entegrasyonunu sağlar. (ör. Üretim planlama)[1]

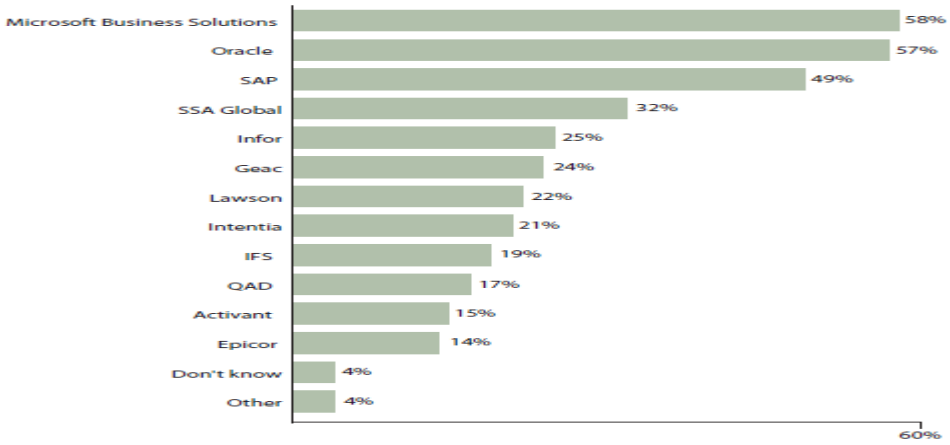
2 KURUMSAL KAYNAK PLANLAMA YAZILIMLARI

2.1 Türkiye ve Dünyada Kurumsal Kaynak Planlama Yazılımları

Araştırmalar, Fortune500'deki şirketlerin tamamının ERP yazılımı kullandığını ortaya koyuyor. Ancak, Capital500'de bu sayı 50 ile sınırlı. Bu nedenle yerli ve yabancı şirketler pazara sarılıyor, henüz çok küçük boyutta olan ERP işini büyütmeye çalışıyor. Dünyada 2003 yılında 9.44 milyar doları yakalayan bu Pazar, Türkiye'de henüz 28 milyon dolar düzeyinde. Bütün şirketlerin gözü büyüklerde. Fakat, yarının hedefi olan KOBİ'ler ise göz ardı edilmiyor. [2]

2.2 Başlıca Kurumsal Kaynak Planlama Yazılımları

Nucleus Research firmasının 2005 yılında yaptığı araştırmada "Gelecek sene içerisinde kurumsal kaynak planlama uygulamasını alacak olsanız hangi tedarikçiyi kullanırdınız" sorusuna verilen cevaplarda liderler; Microsoft Business Solutions, Oracle ve SAP olmuştur. SSA Global ve Infor ise ilk beşte yer almaktadır. [3]



Şekil 1.1: Başlıca Kurumsal Kaynak Planlama Yazılımları [4]

2.2.1 Sap İsimli Firmanın Kısa Tarihçesi

1972'de, beş eski IBM çalışanı tarafından " System Applications and Products in Data Processing" adı ile Almanya Mannheim'da kurulmuştur. Temel vizyonları gerçek zamanlı iş süreçleri için standart bir uygulama yazılımı geliştirmekti. [5]

120'den fazla ülkede 86.000'den fazla müşteri, küçük ve orta ölçekli işletmelerin ihtiyaçlarından küresel işletmelerin tekliflerine kadar her şey için SAP uygulamalarını kullanıyor. [6]

2.2.2 Oracle İsimli Firmanın Kısa Tarihçesi

1977'de kurulan Oracle firması, Kaliforniya merkezlidir. Veri tabanı ile birlikte kurumsal uygulamalara yer vermektedir.

Bugün, Oracle 320.000 müşterisine 145 ülkede hizmet sağlayan dünyadaki en büyük iş yazılım şirketidir. [7]

2.2.3 Microsoft İsimli Firmanın Kısa Tarihçesi

Microsoft Dynamics AX, Microsoft tarafından üretilen, Microsoft Dynamics ailesinden bir KKP (İngilizce:ERP) yazılımıdır.

Axapta yazılımı, Danimarkalı Damgaard kardeşlerin firması olan Damgaard tarafından üretilmiş ve ilk olarak Mart 1998'de Danimarka ve ABD pazarında satışa sunulmuştur. Damgaard Data 2002 yılında bir başka yazılım üreticisi olan Navision Software ile birleşerek önce NavisionDamgaard daha sonra da Navision adını almıştır.Axapta, Navision'un 2002 yılında Microsoft tarafından satın alınması ile Microsoft Business Solutions ürün ailesine dahil olan yazılım günümüzde 45 dili desteklemekte ve alanında yaygın olarak kullanılmaktadır. Microsoft Business Solutions ürün grubu 2006 yılında Microsoft Dynamics olarak isim değiştirmiştir. Axapta'nın ismi bu andan itibaren Dynamics AX olmuştur. [8]

2.3 En Çok Tercih Edilen Üç Kurumsal Kaynak Planlama Yazılımlarının Karşılaştırılması

2.3.1 Maliyet Karşılaştırma

Nucleus Research tarafından ürünlerin ilk maliyetleri ve devam eden süreçlerdeki maliyetleri karşılaştırıldığında aşağıdaki tablo ortaya çıkmaktadır.

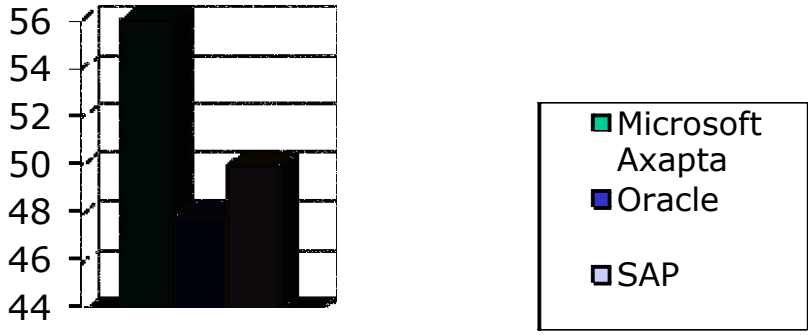
Tablo 2.1 Kurumsal Kaynak Planlama Sistemleri Maliyet Değerlendirme [3]

Ilk Maliyetler	Microsoft Axapta Ortalama	Microsoft Axapta Medyan	Oracle Ortalama	Oracle Medyan	SAP Ortalama	SAP Medyan
Yazılım	571,234	168,000	1,729,688	550,000	1,853,333	1,000,000
Danışmanlık	1,169,745	600,000	2,301,429	787,500	3,637,039	850,000
Donanım	180,188	84,750	284,500	62,500	882,000	345,000
İnsan Kaynağı	346,667	220,000	675,000	750,000	2,480,000	806,663
Eğitim	141,375	72,000	263,088	57,072	1,271,550	137,500
Toplam	2409,207	1,144,750	5,253,705	2,207,072	10,123,92	3,139,163

Devam Eden Maliyetler	Microsoft Axapta Ortalama	Microsoft Axapta Medyan	Oracle Ortalama	Oracle Medyan	SAP Ortalama	SAP Medyan
Yazılım Bakımı	**274,192**	**75,600**	**591,924**	**330,000**	**1,016,616**	**637,500**
İnsan Kaynağı	**576,000**	**300,000**	**1,590,000**	**900,000**	**3,360,000**	**1,440,000**
3 Yıllık Toplam Maliyet	**3,259,399**	**1,269,950**	**7,435,629**	**3,437,072**	**14,500,538**	**5,216,663**

2.3.2 Temel Faydalar

- Axapta müşterilerinden %56'sı verimliliği arttırdı ya da personel maliyetlerini düşürdü.
- Axapta müşterilerinden %44'ü bilişim(IT) harcamalarını düşürdü
- Axapta müşterilerinden %19'u müşteri ya da ortak memnuniyetini arttırdı.
- Axapta müşterilerinden %75'i operasyonlarını ve şeffaflığını geliştirdi.
- Oracle müşterilerinden %48'i kullanıcı verimliliğini arttırdı.
- Oracle müşterilerinden %39'u finansal yönetimini geliştirdi.
- Oracle müşterilerinden %26'sı bilişim(IT) harcamalarını düşürdü.
- SAP müşterilerinden %50'si verimliliğini arttırdı.
- SAP müşterilerinden %35'i organizasyon ve kullanım hakkında bilgisini geliştirdi.
- SAP müşterilerinden %30'u operasyon yönetimini geliştirdi.



Verimlilik artışı sağlayan firmaların % değerleri

Şekil 2.2: Kurumsal Kaynak Planlama Sistemleri Ürün Değerleme

2.3.3 Karşılaşılan Temel Sorunlar

Temel SAP Sorunları

- Genişleme ve yinelenebilirlik yoksunluğu. Firmalar ümit ettikleri kadar kullanıcıya yayma şansına sahip değillerdi.
- Yüksek personel maliyetleri
- Aşırı düzenleme. SAP'nin ihtiyaçlarını karşılayabilmesi için müşteriler yeni ara yüzler ya da özel kodlanan iş süreçleri geliştirdiler. Bu da yayılımı yavaşlattı ve danışmanlık maliyetlerini yükseltti.

Temel Oracle Sorunları

- Dış danışmanların desteğine ihtiyaç duyuyor.
- İhtiyaç fazlası olarak alımlar. Bazı firmalarda uygulamalar kullanılmak üzere rafta bekliyor.
- Birçok firma Oracle uygulamalarının. 0 yayınlarını kullanmasaydı başarılı olamayacaktı.

Temel Axapta Sorunları

- Yerelleştirme düzeylerinin belirlenmesi. Bazı organizasyonlar orijinal beklentilerinden daha fazla yerelleştirme gerektiğini fark edince programın yayılımını durdurmak zorunda kaldılar.
- Bazı firmalar danışman (partner) seçiminde proje süreçlerini ve zorluklarını daha iyi görebilecek partner seçiminde dikkat etmesi gerektiğini fark etti.

2.4 En Çok Tercih Edilen Üç Kurumsal Kaynak Planlama Yazılımlarının Teknoloji Alt Yapılarının Karşılaştırılması

2.4.1 Sap Firmasının Hazırlamış Olduğu Paket Program ile Microsoft Firmasının Hazırlamış Olduğu Paket Programın Karşılaştırılması

Bu bölümde Technologyevaluation firmasının ERP kullanan firmalar için yaptığı araştırmaya değineceğiz. Araştırmalarda öncelikle uygulama yapılan firmanın ihtiyaçları, segmenti ve çevresel faktörler ile firma için faktörler temel alınarak gruplandırma yapılmıştır. Karşılaştırmalarımızda tek bir firma ihtiyacı temel alınarak örnekleme yapılacaktır. Karşılaştırma yalnızca teknik ve geliştirme alt yapısı ile sınırlı tutulmuştur. Firmanın uyguladığı anketler ile aşağıdaki sonuçlar elde edilmiştir. [9]

Temel alınan etkenler;

Firma İhtiyacı: Proses Üretim (Process Manufacturing)

Sektör: Tüm Sektörler

Gelir Aralığı: 50-200 milyon $

Kullanıcı Sayısı: 200-500 kullanıcı

Toplam Bütçe Aralığı: 250-500 bin $

Organizasyonel Yapı: Uluslararası

İş Kategorisi: Üretim Firması

Bölge: Avrupa (Batı, Orta, Doğu)

Dil: İngilizce

Üretim Yöntemi: Her türlü (Yığın, Stoğa Üretim, Siparişe Üretim v.b.)

Modüller: Temel ERP modülleri (Stok, Finans, Lojistik ve Ticaret, Analitik, İş Zekâsı, E- Ticaret, Sabit Kıymet, İş Akışı

Kurulum Süresi: 2-6 Ay

Ürün teknolojisi açısından incelendiğinde Microsoft Dynamics Ax'in %3.63'lük bir fark payı ile ilk sırada olduğu görülmektedir. Ürün teknolojisi başlığı altında temel olarak incelenen baz alınan başlıklar;

- Mimari (Architecture)
    - Veri Entegrasyon Teknolojileri (Data Integration Technologies)
    - Mesajlaşma Protokolleri (Messaging Protocols)
    - Aygıt Ara yüzü (Device Interfaces)
    - Mimari Temel (Architectural Foundation)
    - Web Yeteneği (Web Enablement)
    - Organizasyonel Yapı (Orgnizational Structure)
    - Güvenlik (Application Security)
    - Birden Fazla Yerleşim Yönetimi (Multisite Management)
- Geliştirme Araçları (Application Tools)
    - Yazılım Geliştirme Araçları (Application Developmet Tools)
    - Yazılım Yönetim Araçları (Application Management Tools)

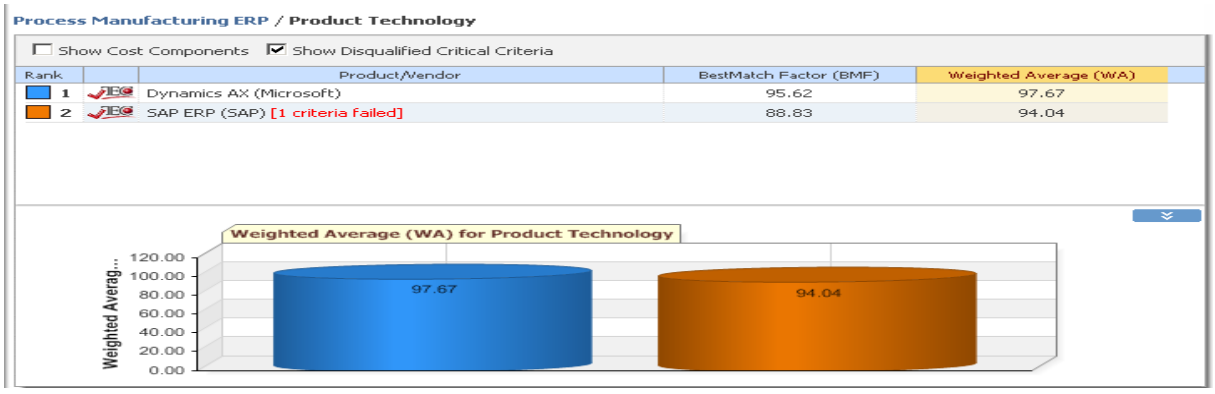
- Süreç Modelleme ve Güncelleştirme (Process Modeling and Updating)
- İş Modeli Geliştirme (Business Model Generation)

- İş Akışı ve Döküman Yönetimi
- Raporlama(Reporting)

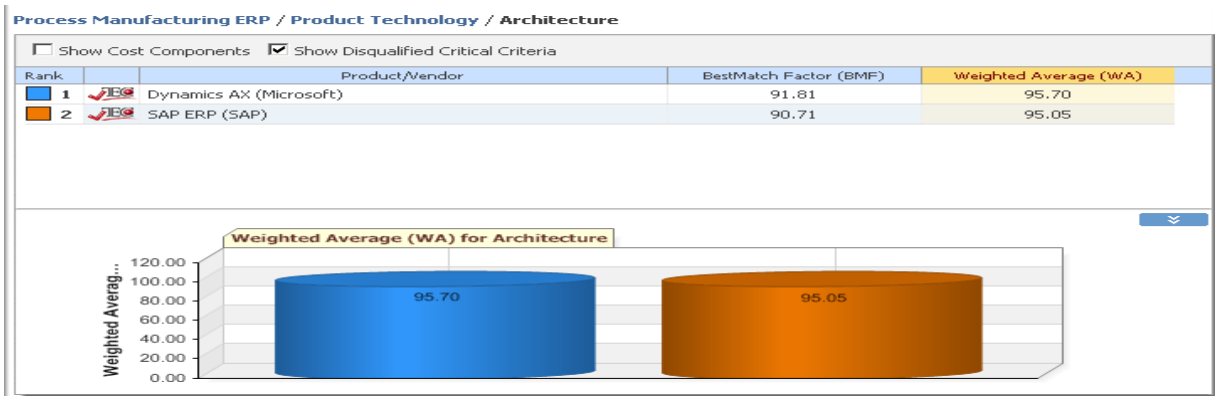
Yapılan araştırmasa mimari, geliştirme araçları ve iş akışı ve döküman yönetimi konularında önde olduğu görülmektedir. Raporlama başlığında ise başa baş olduğu görülmüştür.

Kullanıcı ara yüzü ve platform değişiklik gösterip düzenlenebileceğinden kapsamdan çıkarılmıştır.

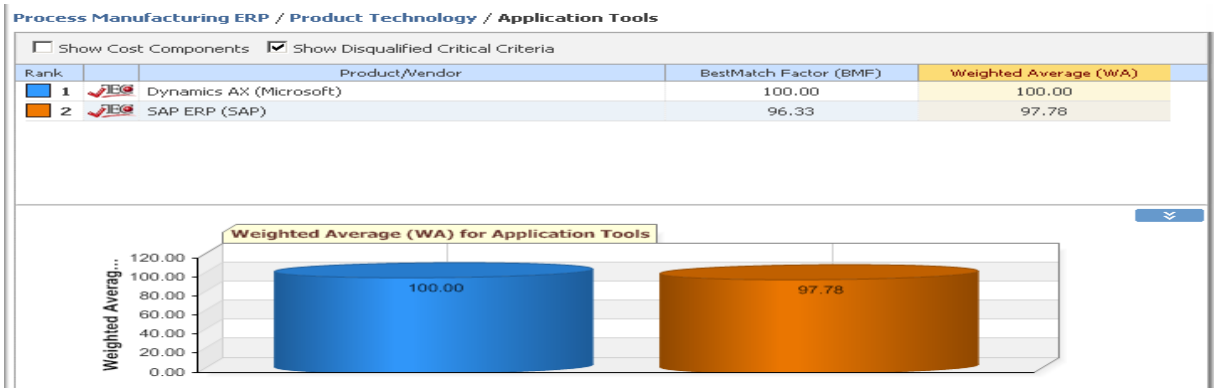
Aşağıda yer alan şekiller ile SAP ile Microsoft paket programlarının birbirlerine sağladıkları üstünlükler gösterilmektedir.



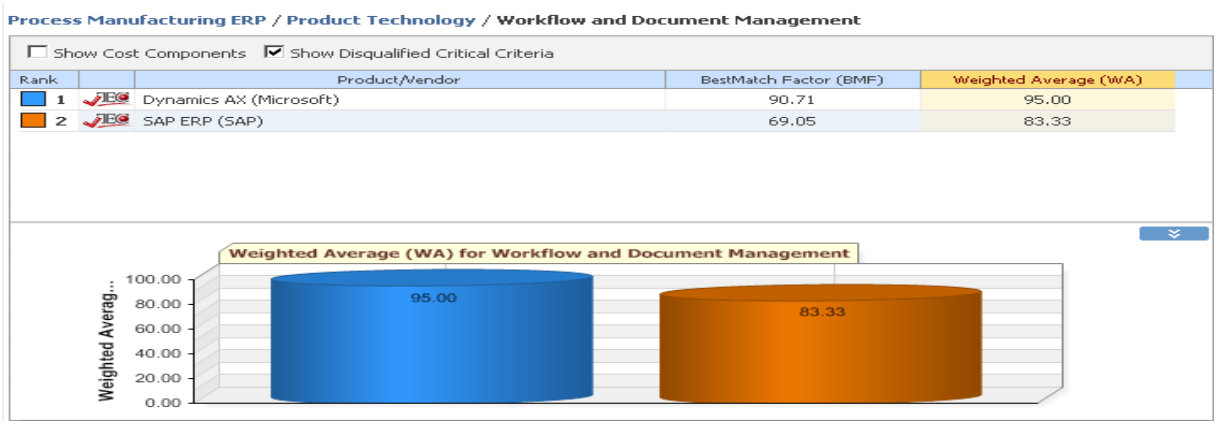
Şekil 3.3: Ürün Teknolojisi karşılaştırılması



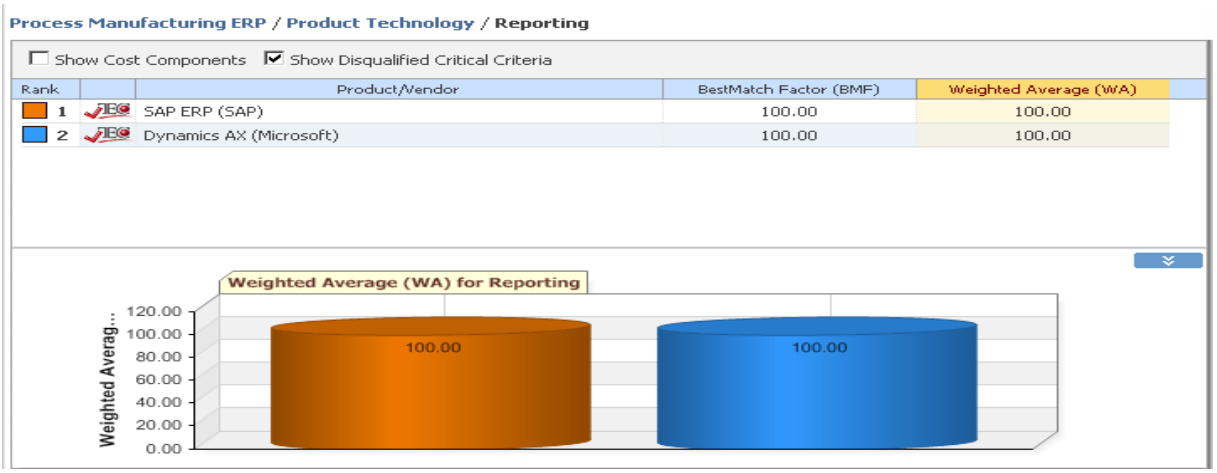
Şekil 4.4: Mimari karşılaştırılması



Şekil 5.5: Geliştirme Araçları karşılaştırılması



Şekil 6.6: İş Akışı ve Doküman Yönetimi karşılaştırılması



Şekil 7.7: Raporlama altyapısının karşılaştırılması

2.4.2 Oracle Firmasının Hazırlamış Olduğu Paket Program ile Microsoft Firmasının Hazırlamış Olduğu Paket Programın Karşılaştırılması

Bu bölümde Microsoft'un Oracle Erp programından Microsoft Dynamics Ax Erp programına geçen birkaç firmanın örnek olay incelemelerine değinilecektir. Anketler bakıldığında Oracle firmasının Erp çözümü Microsoft Dynamics Ax Erp programına en yakın rakip olarak görülmektedir.

Örnek olay incelemelerinde öncelikle Network Equipment Tehnologies firmasının programını değiştirme nedenlerini inceleyeceğiz. Kısaca NET[10] olarak değineceğimiz firma 2006 yılında yüksek operasyon maliyeti, kullanma ve bakım zorluğu nedeni ile Oracle Erp programını değiştirmeye karar vermiştir. 7 yıllık kullanımına rağmen, verimsizliği arttıran ve kullanıcıları hayal kırıklığına uğratan, kullanıcının verimliliğini azaltan, iş sürecinin uygulanması için çeşitli iş çözümleri ve manuel çaba gerektiğini görmüştür.

Son kullanıcının verimliliğini arttırmak için Ax'a geçişe neden olan beş temel etken vardır:

- Kullanılabilirlik ve Aşinalık
- İşlemsel Verimlilik
- İşbirliği
- İş Anlayışı
- Değişkenlik

Bu aşamadan firmanın değişkenlik başlığı altında değindiği temel konular ERP sisteminin yönetim maliyetlerinin düştüğü ve geliştirmeci verimliliğini arttırdığıdır. Artık personel kendi değişikliklerini kendi yaptığından personel kendi kendine yetebilmekte ve kullanıcı ihtiyaçlarını giderebilmektedir. Böylece temel iş süreçlerini yazılımına katabilmektedir. Aynı zamanda geliştirmeci maliyetlerini azaltmış hatta versiyon güncellemesini üç ayda çoğunlukla kendi iç ekibi ile çözmüştür. Bunun içinde Ax'a özgü, güncellemeyi mümkün kılan karşılaştırma (compare) özelliği kullanılmıştır.

Net firması içinde yer alan geliştirmeciler daha önce Ax uygulamasında bulunmamalarına rağmen hızlı bir şekilde geliştirmeyi öğrenmişlerdir. Özellikle alan ekleme, silme ve alan özelliklerinde değişiklik genel olarak talep edilmektedir. Hatta geliştirmecilerden biri "Microsoft Word'de olduğu gibi alanın üzerine gelip, sağ klik ile özellikler görülebilmekte ve değiştirilebilmektedir. Oysa Oracle'da öncelikle alana gitmek, veri tabanına bağlamak, tekrar konuşlandırmak (deploy) gerektirdiği ve bunun Ax'da yalnızca birkaç saniye aldığı" açıklamasında bulunmuştur.

Ayrıca NET firmasının IT direktörü Oracle ile kritik süreçlerini ERP yazılımına uydurmaları gerektiğini fakat şimdi Microsoft Dynamics Ax ile iş ihtiyaçlarını yazılıma adapte edebildiklerini belirtmiştir.

Değişikliğin ve gelişmenin mecbur olduğu günümüz döneminde geliştirme kolaylığı en büyük koşuldur.

Bir başka geçiş örneği ise tedarik zinciri entegratörü olan Paccess[11] firmasıdır. Paccess firması Ax'a sahip olmanın maliyeti azaltmasının ötesinde temel olarak aşağıdaki başlıklara değinmektedir.

- Raporlamanın Gelişmesi
- Kolay Kullanım
- Veri Entegrasyonu
- Büyüme Desteği
- Daha İyi Planlama
- Tedarik Zinciri Yönetimi
- Sektöre Özel İhtiyaçlar

- IT Desteği ve Yönetimi

PACCESS Microsoft Dynamics Ax'i, tek bir çözüm içerisinde, ürün ve servis sektöründe detaylı ihtiyaçları karşılayabilecek tek teknoloji olarak bulmuştur.

PACCESS 'e çok güçlü bir değer katacak, sanatsal bir görkeme sahip geliştirme ortamı ile BT personeline fonksiyonalite kazandıracak, güçlü ERP teknoloji kazandığını belirtmiştir.

2.5 Microsoft Dynamics AX'ın seçilmesi için Başlıca Nedenler

- **Teknik Altyapı :** Microsoft Dynamics AX katmanlı mimarisi ile, Microsoft tarafından sağlanan kaynak kod ile iş ortağı ve müşteri tarafından yapılan geliştirmelerin ayrı katmanlarda saklanmasını sağlar. Bu sayede hem iş ortağı hem müşteri ekibi, var olan yazılımı iş süreçlerine uygun hale getirmek, gerekirse ek fonksiyonalite geliştirmek için sınırsız imkâna kavuşur. Katmanların birbirinden bağımsız bakımının yapılabilmesi sayesinde, sürüm yükseltmeleri en yoğun geliştirme yapılan projelerde bile iki haftayı geçmez.
- **Çalışan Verimliliği** : Bağımsız araştırma şirketi Keystone Strategy tarafından Mart 2007 tarihinde yapılan araştırmaya göre, Microsoft Dynamics kullanıcılarının en yakın rakip SAP' ye karşı %18'lik daha verimli çalıştıkları ortaya konmuştur.
- **Microsoft Office Entegrasyonu :** Microsoft Dynamics AX diğer Microsoft Office ürünleri (Sharepoint, Excel, Word, Outlook, Infopath vb) çift yönlü entegrasyona sahiptir. Çalışanlarınız isterlerse AX ekranlarını açmadan, alışık oldukları ürünler üzerinden veri inceleme, veri güncelleme ve veri girişi yapabilir. Her türlü raporu Excel ya da Word'de açabilirler.

- **Sure Step**: Microsoft Dynamics ürünlerinin kurulumunu hızlandırmak ve kalite güvencesini sağlamak adına oluşturulan Sure Step metodolojisi ile, hem proje yönetimini, hem sistem kurulumunu hem de proje dökümantasyon ve eğitim materyallerini çok hızlı bir şekilde oluşturabilirsiniz.
- **Sizin Yazılıma Değil, Yazılımın Size Uyduğu Çözüm:** Rekabette öne çıkmak için sektörünüz için standart hale gelen iş süreçlerini değil, firmanıza özel iş süreçlerinin desteklenmesi gerekmekte. Standart süreçlere göre tasarlanmış ve bu süreçleri değiştirmenin ya da ek fonksiyonalite geliştirmenin zor olduğu çözümler size rekabette dezavantaj getirecektir. Kısa zamanda ek yazılımlar ile desteklenmeleri kaçınılmaz olacak, bu da ek maliyet ve entegrasyon sorunlarına sebep olacaktır. Microsoft Dynamics AX her türlü iş sürecinizi kendi platformunun üstüne alarak, ERP' ye geçiş amacınız olan tek veritabanı ve tek uygulama hedefinize ulaşmanızı sağlayacaktır.
- **Dış dünya ile hızlı entegrasyon** : Tedarik ve değer zincirinizde bulunan tüm taraflar ile belgelerinizi anlık olarak paylaşarak birlikteliğinizi güçlendirmenize yardımcı olacaktır. Üstün entegrasyon yapısı ile Microsoft Dynamics AX kurulumunuzu dakikalar içinde dış sistemlerle entegre edebilirsiniz.
- **Geliştirme Hızı** : Microsoft Dynamics AX'in geliştirme ortamı ek geliştirmelerinizi hızlı ve güvenli bir şekilde geliştirmenizi sağlar. Nesne bazlı programlama dillerine hakim olan bir uzmanın en fazla iki hafta içinde sistemi anlamaya ve kod geliştirmeye başlaması sayesinde, şirketlerin kendi kaynakları ile geliştirme yapmaya başlamaları çok hızlı olur. Sürüm yükseltmelerinde de şirket içi kaynaklar ile pek çok adım başarı ile gerçekleştirilebildiği için, çok daha az dış danışmanlık hizmeti ile yeni, sürümlere geçilebilir. Değişen iş süreçleri için, şirket kendi kaynakları ile çözümleri hemen devreye alır.
- **Rol Bazlı Ekranlar:** Çalışanlarınız, ihtiyaç duyacağı her türlü bilgi, rapor ve dış uygulama ekranlarına tek bir ekrandan ulaşabilirler. Her türlü belgeyi otomatik olarak üretebilirler. Destek ve operasyon departmanlarında çalışan

sorumlular, ihtiyaç duydukları bilgilere ve kendilerine atanmış görevlere ve Rol Bazlı Kullanıcı ekranları ile daha etkin çalışırlar.

- **İş Zekası ve Raporlama :** Microsoft Dynamics AX'i kullanmaya başladığınız andan itibaren iş zekası küpleri dolmaya başlar. Ek yatırım yapmanıza gerek olmadan SQL BI ile analizlere başlayabilirsiniz. Ayrıca Microsoft® Office Excel®, PerformancePoint Server ve Microsoft Dynamics AX Analiz Çözümleri ile ihtiyaç duyduğunuz tüm analizleri alabilirsiniz. Toplam sahip olma maliyeti için büyük kazanım elde edersiniz.
- **Birleşik İletişime Hazır Çözüm :** Microsoft Office Communication Server ile tam entegre yapısı sayesinde, çalışanlarınız, müşterileriniz ve tedarikçileriniz ile AX ekranlarından ister anlık mesajlaşma, ister sesli konuşma, ister e-posta aracılığı ile anında iletişime geçebilirler. İletişim giderlerinizi çok düşürecek olan bu iletişim şekli, aynı zamanda ilgili taraflar arasında ilişkilerin güçlenmesini de sağlayacaktır.
- **Bilgiye Hızlı ve Kolay Erişim :** Sisteme bağlanmadan Microsoft Dynamics AX Kurumsal Portalleri, PerformancePoint Server ve ya SharePoint Server üzerinden ihtiyaç duyulan bilgiye her an ve her türlü cihazdan ulaşabilirsiniz. Mobile Client aracılığı ile mobil cihazlarınız için çok hızlı ekran tasarlayıp devreye alabilirsiniz.
- **Toplam Sahip Olma Maliyeti :** ERP projelerinin toplam sahip olma maliyetini hesaplamak için, 3 ile 5 yıl içinde yapılacak toplam yatırımı hesap etmek gerekir. Bu hesabın içine ERP Yazılımının Lisans Bedeli, yazılımı şirket ihtiyaçlarına uygun hale getirmek için harcanacak danışmanlık bedeli, sistemi desteklemek ve yedeklemek için gerekli donanım ve alt yapı bedeli, proje sırasında kullanılacak şirket içi personelin toplam maliyeti ve yazılımı kullanacak son kullanıcıların eğitimi için harcanacak şirket içi personel ve eğitmen giderleri mutlaka katılmalıdır. Bağımsız araştırma şirketlerinden Nucleus Research tarafından yapılan bir araştırmada Microsoft Dynamics AX müşterilerinin %56'sı, çoğu ERP' ye geçiş sonrası yaşanan endirekt işgücü artışı yaşamadıklarını, aksine endirekt işçilik giderlerinde düşüş yaşandığını beyan etmişlerdir. Aynı şekilde Microsoft Dynamics AX müşterilerinden

%44'ü, diğer ERP yazılımlarında yaşanan Sistem Yönetim giderlerinde artış yaşamadıklarını ve genel BT Giderlerinin düştüğünü beyan etmişlerdir. Ölçeklenebilirlik ve Performans: Microsoft Dynamics AX, sizinle birlikte büyüyecek bir teknolojiye sahiptir. Microsoft Dynamics AX 5 kullanıcılı küçük ölçekli şirketlerde kullanıldığı gibi 10,000 kullanıcılı Red Flags zincirlerinde de kullanılabilmektedir. Yayınlanan son performans test raporunda 3075 kullanıcı tarafından, saatte 200,000 satır satış siparişi girilerek stres testi uygulanan, 7 Uygulama Sunucusu ile desteklenen sistem, sipariş saklama, onaylama, çeki listesi hazırlama, stok çıkışı gibi işlemler bir saniyenin altında tamamlanmıştır.

- **Kolay Sürüm Yükseltme:** Microsoft Dynamics AX üzerinde yapmış olduğunuz tüm geliştirmeler ayrı katmanlarda saklandığı için, iki yılda bir planlanan AX yeni sürümlerine geçişiniz çok hızlı olmaktadır. Microsoft Dynamics AX'in yapılan geliştirmeleri bir dosya olarak dışa aktarma özelliği sayesinde, ne kadar geliştirme yapmış olursanız olun, bu geliştirmeleri bir dosya olarak dışa aktarabilirsiniz. Bu dosyayı yeni sürüme kopyalayıp, otomatik uyum kontrolü yaptıktan sonra, önerilen kod değişikliklerini gerçekleştirip yeni sürümü devreye alabilirsiniz. Böylelikle sürüm yükseltme maliyetleriniz çok düşük olurken, her iki senede bir yayınlanan ana sürümler ile teknolojiyi çok daha yakından takip edebileceksiniz.
- **Yüksek ARGE Yatırımı :** Microsoft Dynamics AX, dünyanın ARGE'ye en fazla yatırımını yapan Microsoft tarafından stratejik ürün kategorisinde değerlendirilmektedir. Bu nedenle alternatiflerinden çok daha yüksek ARGE bütçeleri ile yazılımı sürekli geliştirmektedir. Microsoft'un "Çalışanlar İçin Teknoloji" vizyonu ile, kurumsal hayatta çalışanların daha verimli olması için sunulan en etkin ürünüdür. Çalışanların bilgiyi hızlı üretip, hızlı paylaşması hedefi ile sürekli geliştirilen Microsoft Dynamics AX, Microsoft'un sunduğu Microsoft Office, Microsoft Office Sharepoint, Exchange Server, Unified Communications, SQL İş Zekası gibi diğer ürünleriyle tam entegre olarak çalışmaktadır. Microsoft ürünleri arasında sunulan bu entegrasyon ile çalışan

verimliliği artarken, BT Giderleri de ortak tek platformun getirdiği avantaj ile sürekli düşmektedir. [8]

Böylece anlaşılacağı gibi Microsoft Dynamics Ax hem süreçlere yaklaşımı ile süreç yönetiminde hem de teknik olarak daha avantajlı bir ürün haline gelmektedir. Tez konusu olarak Microsoft Dynamics Ax'ın teknik avantajları ve yapısı incelenerek değerlendirilecektir.

3 MİMARİ YAPI

Microsoft Dynamics Ax'ı donanım ve geliştirme adı altında iki başlık altında incelenecektir. Bu başlık altında Ax'ın donanım temeline değinilecektir.

Bu bölümde değineceğimiz konuları dört başlık altında toplayabiliriz.

- AOS
- Veritabanı
- Active Directory (Güvenlik)
- Server Configuration Utility, Configuration Utility

3.1 Aos

Axapta yazılımı üç katmanlı olarak çalışabilecek yapıda tasarlanmıştır.

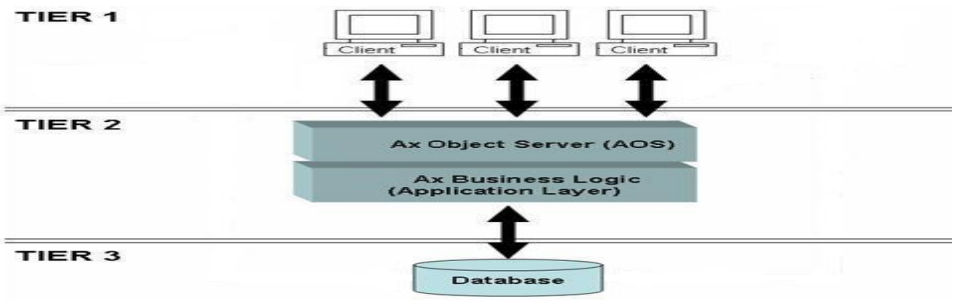
Axapta sisteminin altyapısına baktığımızda dört adet bileşenden oluştuğunu görmekteyiz :

İstemciler (Client)

Uygulama Sunucusu (Application Server)

Uygulama Nesne Sunucusu (Axapta Object Server)

Veritabanı Sunucusu (SQL Server veya Oracle)



Şekil 8.1: Yazılım Katmanları[12]

Şekil 1'deki mimari Axapta'nın 3 katmanlı mimarisidir. Burada veritabanı ile haberleşme ve nesneleri yaratma işlemleri Axapta Nesne Sunucusu aracılığıyla gerçekleşir. Nesne sunucusu uygulamasında önbellekleme (caching) mekanizması da çalıştığı için çok kullanıcılı uygulamalarda büyük bir hız artışı oluşur. İstemciler sadece nesne sunucusundan gelen sonuçları görüntüleyeceği için yüksek maliyetli PC'ler kullanmanız gerekmez. Ayrıca her istemci veritabanı sunucuna erişmeyeceği için network üzerindeki trafik de oldukça düşük olacaktır. Tek bir uygulama nesne sunucusunun yeterli olmadığı durumlarda istediğiniz miktarda nesne sunucusu kurabilir. Bu sunuculara Windows servislerinden ulaşılabilir ve her biri birbirinden ayrı olarak durdurulup başlatılabilir böylelikle her bir sunucunun bakım ve kontrolü diğer çalışan sistemleri etkilemeden yapılabilir. Her uygulama sunucusu

tekrarlanmayan kendine has bir port numarası üzerinden çalışır ve kullanıcılar bu port numarası ve sunucu adını belirterek istedikleri uygulamaya bağlanabilirler. Bu ayarları yaptıkları ekrana Microsoft Dynamics Ax Configuration Utility adı verilir. Uygulama nesne sunucusunun ayarlamalarının yapıldığı ekrana ise Microsoft Dynamics Ax Server Configuration Utility adı verilir. Bu ekran üzerinde uygulama sunucusunun bütün parametrik ayarlarına ulaşmak ve değiştirmek mümkündür. Yapılan ayarların uygulama nesne sunucusunda aktif olabilmesi için Windows servisinin kapatılıp tekrar başlatılması zorunludur, bu işlem esnasında uygulama nesne sunucusuna bağlı her kullanıcının bağlantısı kopacak ve tekrar başlamaları gerekecektir. Bu bölümlere ilerideki bölümde detaylı olarak değinilecektir

3.2 Veri Tabanı (Database)

Her uygulama kendine ait bir veritabanı istemektedir ve uygulama ile ilgili bütün kayıtlar bu veritabanında tutulmaktadır. Veritabanı üzerinde hiçbir modifikasyon yapılmasına gerek yoktur. Veri ile ilgili bütün işlemleri Client üzerinden yapmak mümkündür. Veritabanı üzerinden eklenen tablo ve alanları axapta tarafında görmek erişmek ve veri yazmak mümkün değildir. Axapta üzerinde bilgiler şirket bazında tutulur böylelikle bir uygulama üzerinde birden fazla şirket bilgisi tutabilmek mümkün olmuştur bunu sağlamak için Axapta üzerinde açılan her tablo üzerinde "DataareaId" isimli bir alan tutulmaktadır bu alan 3 karakter uzunluğunda şirket kodu bilgisi tutularak her şirketin bilgisi birbirinden ayrılmaktadır. Böylelikle aynı tablo üzerinde birden fazla şirket bilgisi barınabilmektedir. Her şirket için yeni tablo açılmayacağından veri tabanının daha verimli çalışması sağlanır. Veri tabanı üzerinde tutulan her kayıt tekrarlanmayan bir sayı ile (RecID) kaydedilir. Bu da birincil anahtar bulundurmaksızın tablo üzerinde veri bütünlüğünün sağlanmasına yarar. Axapta kendi sistem tabloları üzerinde hızlandırma amaçlı olarak indekslerini kendisi oluşturmuştur. Ama istenirse geliştirici tarafından da gerek kendi gerekse

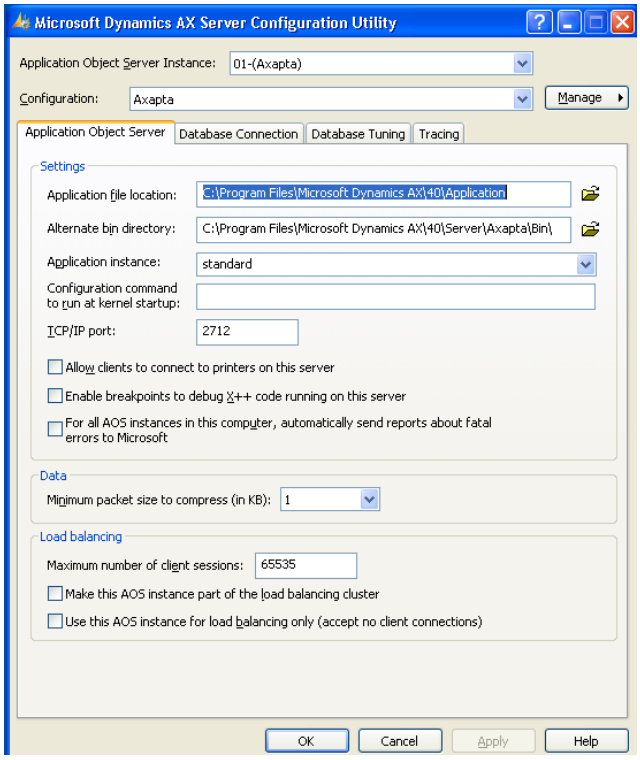
sistem tabloları üzerine ek indeksler oluşturmak mümkündür. Uygulama sunucusu üzerinde çalışan her kullanıcı veri ile ilgili işlem yaptığında aynı anda bu değişiklik veri tabanına yansımaktadır. Bu işlem AOS sayesinde her kullanıcı için yürütülmektedir.

3.3 Active Directory

Active Directory Microsoft tarafından kullanıcıların Domain yapısı içerisinde bulunan bilgisayarlar üzerinde oturum açmalarına ve yetkilendirmelerine olanak sağlayan bir yapıdır. Bu yapı sayesinde her kullanıcı yetkisi dahilinde istediği bilgisayara girişi yapabilmekte ve ilgili bilgisayar üzerinde yetkisi olan işlemleri gerçekleştirebilmektedir. Axapta programına girişi yapılırken client uygulaması kullanıcıya hiçbir şekilde şifre ve kullanıcı adı sormadan, Active Directory üzerinden oturum açtığı kullanıcı bilgisi ile AOS a ulaşır ve eğer Axapta tarafında ilgili kullanıcıya programa ulaşma yetkisi verilmiş ise programı başlatıp kendi bölümü ile ilgili menü ve işlemlere ulaşabilir.

3.4 Sunucu Konfigürasyon Uygulaması (Server Configuration Utility) ve Uygulama Konfigürasyonu (Configuration Utility)

Server Configuration Utility uygulama nesne sunucusunun yönetildiği ekrandır.

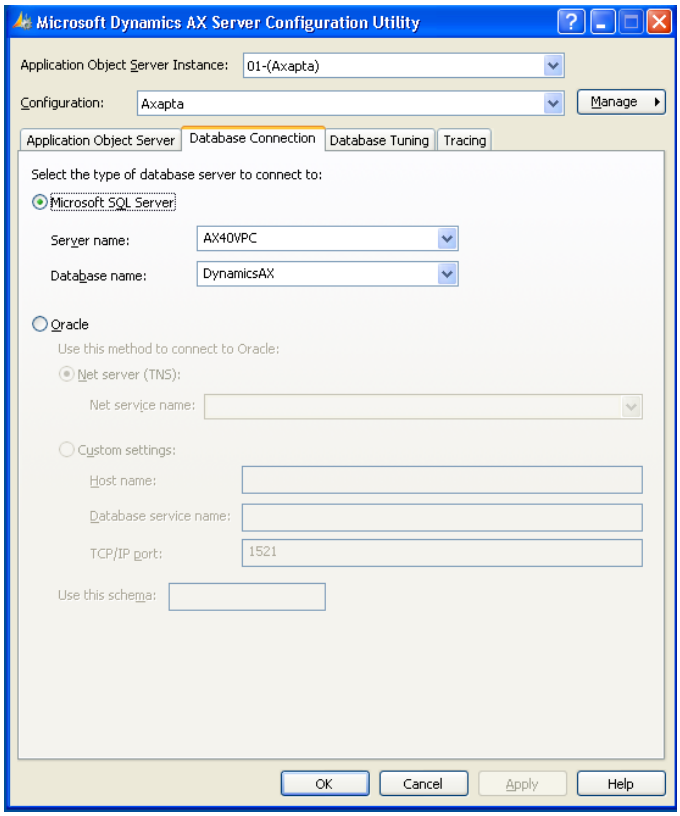


Şekil 9.2: Sunucu Konfigürasyon Uygulaması (İngilizce: Server Configuration Utility) ekranı ile Axapta programının sunucu özelliklerinin ayarlanması. Uygulama kulakçığı bölümü

"Application file location" başlığı altında uygulama dosyalarımızın nerede saklandığı belirtilmektedir. Bu seçtiğimiz klasöre bağlı olarak "Application instance" kutusu bu klasörde bulunan her bir uygulamanın ismi ile dolacaktır ve bunlardan bir tanesini seçerek uygulama sunucusu üzerinde çalışmasını istediğimiz uygulama seçilir. Resimdeki örnekte standart isimli uygulama seçilmiştir. Bu ekran üzerinde sunucumuz üzerinde bulunan her uygulama sunucusuna ulaşabilmemiz ve bunların her birini ayrı ayrı konfigüre edebilmemiz mümkündür. Ekranın en üst tarafında bulunan "application object server instance" başlığı altında bulunan her seçim farklı bir uygulama sunucusunu göstermektedir.

Kullanıcıların bağlanacağı uygulamanın seçimi seçtikleri port ile yapılır. Server Configuration Utility üzerinde bulunan TCP/IP alanında bulunan port numarası ileride bahsedeceğimiz Configuration Utility üzerine yazılacak TCP/IP port ile aynı olmalıdır. Böylelikler birden fazla uygulamanın bulunduğu işletmelerde kullanıcılar istedikleri ortama port numaralarını konfigüre ederek bağlantı kurabilirler.

Microsoft varsayılan olarak port numarası vermeye 2712'den başlamaktadır. Yeni uygulama nesneleri eklendiğinde bu sayı artarak devam edebilir. Kullanıcıların sorunsuz bir şekilde uygulamayı açabilmeleri için ağ üzerinde bu port ile haberleşmeyi engelleyecek herhangi bir kısıtlamanın tanımlanmaması gerekmektedir. Bu portun uygulama sunucu ile veri tabanı sunucusu arasındaki iletişim ile ilgisi yoktur.

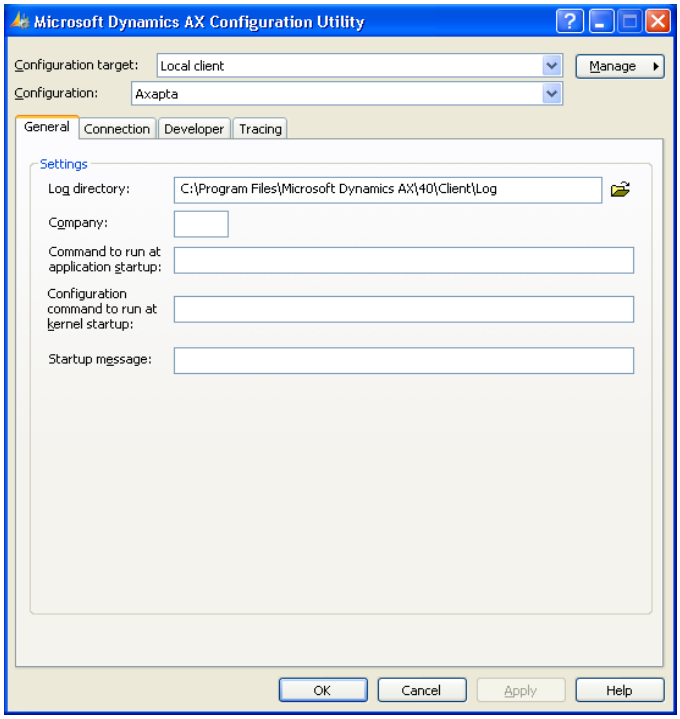


Şekil 10.3: Sunucu Konfigürasyon Uygulaması (İngilizce: Server Configuration Utility) ekranı ile Axapta programının sunucu özelliklerinin ayarlanması. Veritabanı kulakçığı bölümü

Server Configuration Utility üzerindeki ikinci kulakçık olan "Database Connection" üzerinde ilgili uygulamanın bağlantı kuracağı veri tabanı sunucusu ve bu veri tabanına ait veri tabanının seçilebileceği kontroller mevcuttur. Daha öncede belirtildiği gibi her uygulama tek bir veri tabanı ile ilişkilendirilebilir.

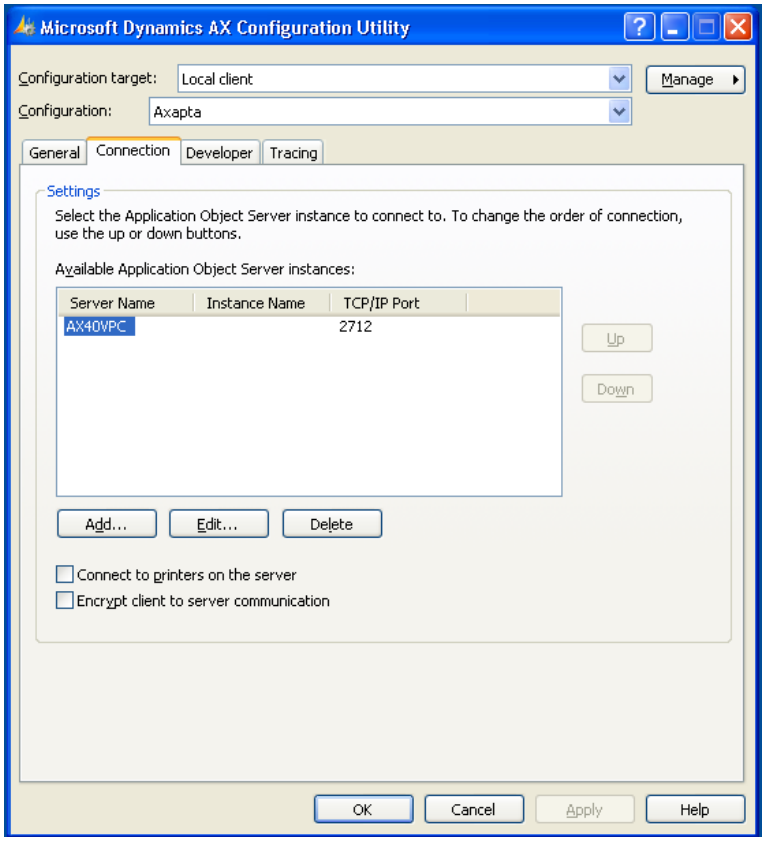
Alt bölümde de görülebileceği gibi hem Microsoft SQL Server ile hem Oracle veri tabanları ile bağlantı kurabilmek mümkündür.

Configuration Utility son kullanıcıların bağlanmak istedikleri uygulama ve bağlantı kuracakları uygulamanın katman bilgisini seçtikleri kurulum ekranıdır.



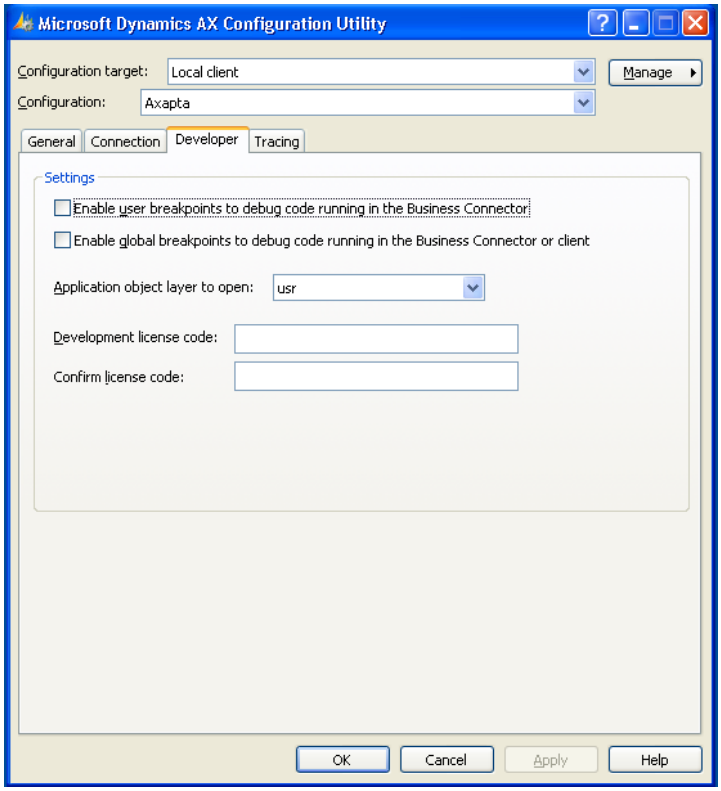
Şekil 11.4: Uygulama Konfigürasyon Uygulaması (İngilizce: Configuration Utility) ekranı ile Axapta programının bağlantı özelliklerinin ayarlanması. Genel kulakçığı bölümü

Genel kulakçığında kullanıcının Axapta açıldığı zaman çalışacağı şirket bilgisi "Company" başlığının yanındaki alan girilecek üç haneli şirket kodu ile belirtilir.



Şekil 12.5: Uygulama Konfigürasyon Uygulaması (İngilizce: Configuration Utility) ekranı ile Axapta programının bağlantı özelliklerinin ayarlanması. Bağlantı kulakçığı bölümü

"Connection" kulakçığında uygulama nesne sunucusunun adı ve bu uygulama için konfigüre edilmiş port numarasının girileceği alanlar mevcuttur.



Şekil 13.6: Uygulama Konfigürasyon Uygulaması (İngilizce: Configuration Utility) ekranı ile Axapta programının bağlantı özelliklerinin ayarlanması. Geliştirme kulakçığı bölümü

"Developer" kulakçığında uygulamaya bağlantı esnasında kullanıcının kullanacağı katman seçimi yapılır. İlgili katmana bağlanabilmek için Microsoft tarafından kullanıcılara verilen lisans üzerindeki katman kodu ilgili alanlara yazılır. Katman mimarisi ile ilgili detaylı açıklama ilgili başlık altında detaylandırılacaktır.

3.5 Geliştirme *Ortamı*[12]

Axapta, sınıfların(classes), tabloların ve daha fazlasının düzenlenebileceği kendi IDE'sine sahiptir. IDE'nin kendisi bile oldukça basittir. Kullanıcıların düzenlemek istedikleri kodları yazabilecekleri kullanıcı alanı, derleme, çalıştırma(run),yardım alma v.b.g. yapabilecekleri butonların bulunduğu bir araç çubuğuna da sahiptir. IDE Microsoft Visual Studio'nun geliştirme ortamına çok benzer komutlara sahiptir.

Axapta geliştirme ortamı, tek ortak çevre üzerinde tasarlama, düzenleme, veri saklama, derleme, ve hata ayıklama gibi çeşitli farklı fonksiyonları içeren bütünleşik bir araç kutusudur. Axapta'nın içerisindeki geliştirme ortamını 3 temel alana ayırabiliriz; IntelliMorph(kullanıcı arayüzü/sunum), MorphX Geliştirme Takımı(MorphX Development Suite)(iş mantığı ve veri sözlüğü) ve veri saklama(veri tabanı).

IntelliMorph

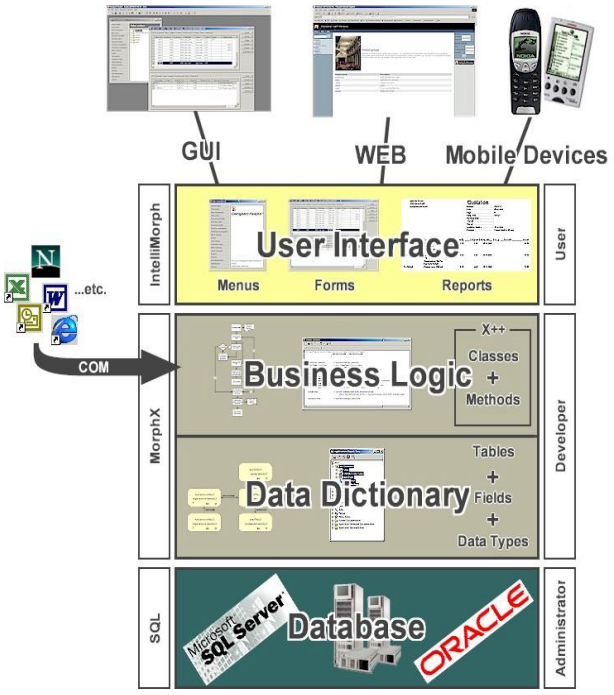
IntelliMorph Axapta'da kullanıcı arayüzünü kontrol eden teknolojidir. Kullanıcı arayüzü, uygulama fonksiyonalitesinin kullanıcıya nasıl sunulduğu ya da gösterildiğidir. Aynı fonksiyonalite birden fazla platform ya da aygıt üzerinde aynı uygulama kodunu kullanarak (Web ya da mobil aygıtlar) gösterilebilir.

IntelliMorph kullanıcı arayüzünün düzenlemesini kontrol eder ve formların, raporların ve menünün düzenlenmesinin zorluklarını ortadan kaldırır.

MorphX Geliştirme Takımı (MorphX Development Suite)

MorpX Geliştirme Takımı(MorphX Development Suite) ERP uygulamalarını geliştirmek için çok amaçlı araç çubuğu olarak dizayn edilmiştir.

MorpX Geliştirme Takımı(MorphX Development Suite) sistem yöneticilerinin ve programcıların Axapta'ya yeni fonksiyonaliteler eklemesini ya da mevcut fonksiyonaliteleri değiştirebilmelerini sağlar. MorpX Geliştirme Takımı(MorphX Development Suite) iş mantığını ve veri modelinin dizaynını idare edilmesini sağlayan ortamdır.



Şekil 14.7: IntelliMorph[13]

İş Mantığı (Business Logic)

Karmaşık ihtiyaçları yeni iş mantıkları gerektirdiğinde, Axapta'nın kendi nesne tabanlı dili X++ kullanılabilir. X++ kapsülleme(encapsulation), miras (inheritance), sınıflar (classes), objeler (objects) ve metodlar (methods) gibi obje tabanlı programlama. Bu dil Javaya benzer sentax(syntax)'lar kullanır. X++ dilinin çeşitli amaçları vardır. Bu bir veri tabanı dilidir, veri tabanı arayüzünü yaratan komut dilidir, rapor yaratma dilidir, kullanıcı arayüzüne form yaratmak için(bölgesel Windows kullanıcıları ve web uygulamaları) kullanılan dildir hatta yardım- sistem dilini de kapsar.

ERP sistemlerinden çok azı bu kadar çok yönlü bir aracı temin edebilir. Axapta X++ radikal olarak zengin ve çok yönlü fonksiyonaliteleri mümkün olan en az kod sayısı ile sağlamaktadır. Daha az kod demek daha az risk daha çok performans demek ve obje tabanlılık geliştirme görevlerinin kolay ve hızlı yapılmasını arttırmaktadır.

Axapta geliştirme ortamı 4 adet başlık altında incelenecektir.

- Geliştirme Katmanları (Layer Metodolojisi)

- Geliştirme Dili (X++)
- Geliştirme Editörü (Code Editor)
- Hata Ayıklama (Debugging)

3.6 Geliştirme Katmanları Mimarisi (Layer Architecture)

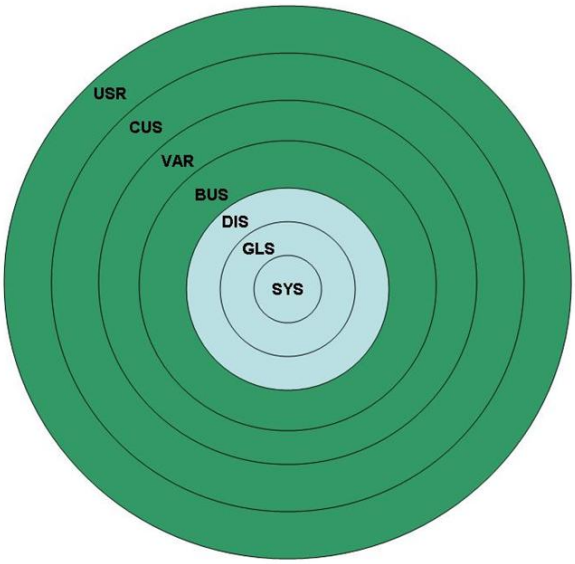
Microsoft Axapta yapılan değişiklikleri kontrol altında tutan benzersiz bir katmanlı yapıya sahiptir. Standard Microsoft Axapta uygulamaları çekirdekte yer almaktadır. Bu çekirdeğin bakımı ve kontrolü Microsoft Business Solutions tarafından sağlanmaktadır Ülkelere, iş alanına ve müşteriye özgü değişiklikler, bu çekirdek katmanı çevreleyen üst katmanlarda yapılabilmektedir. En dıştaki katman her bir kullanıcının kendi kişisel ayarlamalarının tutulduğu katmandır. [12]

Tablo 3.1 Geliştirme Katmanları

SYS	System	Çekirdek uygulama - Microsoft Dynamics Ax'ın Standard uygulamasının bulunduğu en alt katmandır.
GLS	Global Solutions	Hotfixler bu katmandadır
DIS	Distributor	Yerel Microsoft Axapta çözümleri
BUS	Business Solutions	Partner'ların dikey çözümleri (Sektöre yönelik çözümler)

VAR	Value added Reseller	Müşteriye özgü çözümler
CUS	Customer	Kurumun IT bölümü tarafından yapılabilen kendi iç çözümleri
USR	End User	Son kullanıcı değişiklikleri

Sistem Uygulama Katmanları:



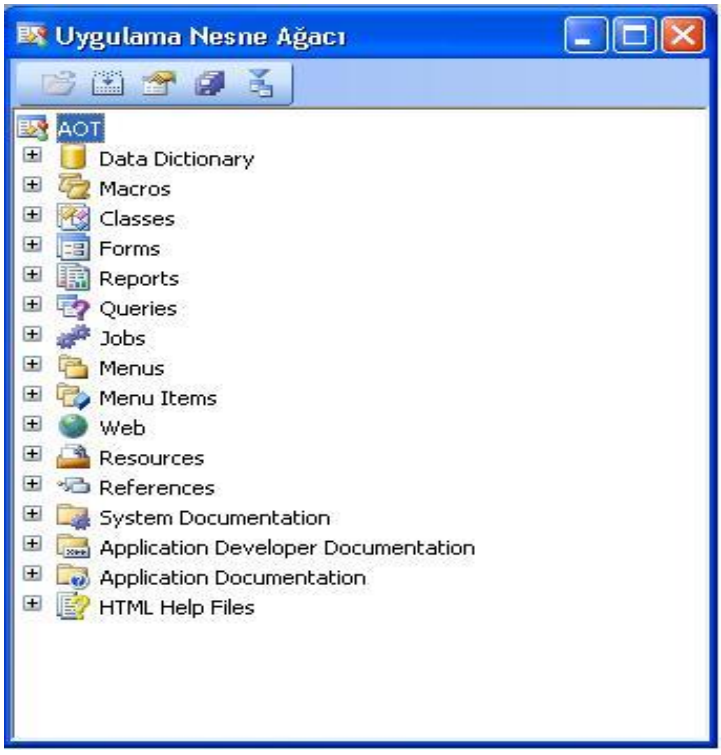
Şekil 15.8: Katman Mimarisi

Standart kurulumdaki nesneler asla değiştirilemez veya silinemez. Üst katmanlardaki değişiklikler ayrı dosyalarda tutulur. Böylece herhangi bir anda orjinal versiyon ile değiştirilmiş versiyon karşılaştırılabilir veya yapılan değişiklikler geri alınabilir.

Örneğin kullanıcı bir rapor çalıştırmak istediğinde kernel ilk olarak en üst katmana bakar burada bir değişiklik yok ise değişiklik bulana kadar alt katmanlara iner ve bu kodu çalıştırır. Değişiklik yok ise orjinal kod çalışır. Orjinal kod SYS veya GLS katmanındadır.

3.7 Geliştirme Dili (X++)

Axapta üzerindeki geliştirmeler uygulama nesne ağacı (AOT) üzerinden yapılmaktadır.



Şekil 16.9: Application Object Tree

AOT üzerinde;

- Veri Sözlüğü (Data Dictionary): Veri ile ilgili bütün objelerin toplandığı ana başlıktır.
    - Tablolar (Tables): Farklı veri tiplerinde veri bulundurabilen objelerdir. Tablonun içerisinde indeks, ilişki (relation) ve verinin her türlü kontrolü için geliştirme yapılabilecek metod kısmına sahiptir.
    - Haritalar (Maps): Axapta üzerinde sistemsel alanlar birden fazla tablo içerisinde bulunmaktadır. Haritalar sayesinde yeni bir veri girişi yapıldığında refere edilen her tablonun ilgili alanına aynı veri sistem tarafından kaydedilir.
    - Bakış (Views): Aralarında ilişki kurulmuş birden fazla tablonun tek bir tabloymuş gibi görüntülenmesine olanak sağlayan nesne biçimidir.
    - Genişletilmiş Data Tipleri (Extended Data Types): Sistemin standart veri tipleri dışında geliştiricinin belirli standartlar ile (alan uzunluğu, hizalama v.b.g) uygulama içerisinde oluşturulabilecek veri tiplerinin türetilebildiği objelerdir.
    - Temel Tipler (Base Enums): Çoktan seçmeli alanların refere edileceği objelerin oluşturulduğu yerdir.
    - Lisans Kodları (License Codes): Lisans bilgilerinin tutulduğu bölümdür.
    - Konfigurasyon Kodları (Configuration Codes): Sistemin yetkilendirme aşamasından kullanılacak grupların tanımlandığı bölümdür.

- Güvenlik Kodları (Security Codes): Sistemin güvenlik bölümünde objelere verilecek yetki kodlarının tanımlandığı yerdir.
- Ortak Tablolar (Table Collections): Axapta veri kaynaklarını şirket (company) denilen bir kavram ile sınıflandırır. Şirket bağımsız çalışması gereken (örneğin; parametreler) tabloların tanımlandığı kısımdır.
- Perspektifler (Perspectives): Raporlama sunucusunun baz alacağı bölümlerin altında bulunacak tabloların tanımlandığı bölümdür.

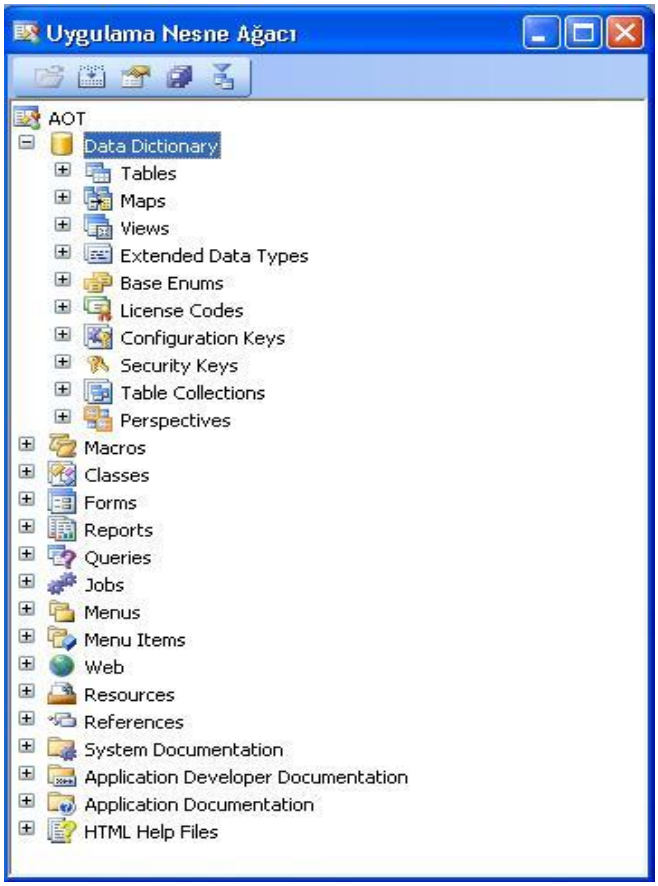
- Makrolar (Macros): Geliştirme içerinde kullanılacak değişkenlerin gerektiğinde varsayılan olarak standart bir değer alması istendiğinde makro oluşturulur.
- Sınıflar (Classes): Nesne tabanlı programlama dillerinin temelini oluşturan sınıf kavramı ve bu sınıf kavramının temel özellikleri (polimorphisim, inheritance) Axapta içerisinde de mevcuttur. Sınıflar başlığı altında sistem sınıflarına ulaşılabilir. Ayrıca geliştirici tarafından da yeni sınıflar oluşturulabilir.
- Formlar (Forms): Uygulamanın kullanıcı ile etkileşime girdiği ara yüzdür.
- Raporlar (Reports):Sistemde bulunan verilerin kullanıcının verdiği kriterlere bağlı olarak belirli bir formatta gösterildiği objelerdir.
- Sorgular(Queries): Sistem içerisinde sık kullanılan sorguların tekrar tekrar program içerisinde tanımlanmasını önlemek amacı ile bir kere oluşturulup sistemde istenilen yerden çağırılabilen sorgu nesneleridir.

- Görevler (Jobs): Geliştiricilerin sınıf yazmadan önce uygulamalarını test etme amacı ile yazdıkları görsel niteliği olmayan program parçalarıdır.
- Menüler(Menus): Kullanıcının sistemde navigasyonunu sağlamak için oluşturulan menüde tanımlama ya da düzenlemenin yapılabileceği bölümdür.
- Menü Öğeleri (Menu Items): AOT içerisinde bulunan sınıf, form veya raporlar nesnelerinin menü karşılıklarının oluşturulması için yaratılacak nesnelerin bulunduğu kısımdır. Bu kısım üçe ayrılır ve her bir alt başlık birbirinden farklı bir simge ile menüde görüntülenir.
    - Görüntü (Display): Formların tanımlandığı başlık gurubudur.
    - Action (Olay): Sınıfların tanımlandığı başlık gurubudur.
    - Output (Çıktı): Raporların tanımlandığı başlık gurubudur.
- Web: Axapta'nın web objelerinin bulunduğu kısımdır.
- Kaynaklar(Resources): Axapta'nın ikon setlerinin bulunduğu kısımdır.
- Referanslar (Referances): Axapta'nın kullandığı kütüphanelerin eklendiği veya görüntülendiği kısımdır.
- Sistem Dökümantasyon (System Documentation): Sistem üzerinde bulunan sistem objelerin açıklamalarının yapıldığı dökümantasyon kısmıdır.
- Uygulama Geliştirmeci Dökümanları (Application Developer Documentation): Sistem üzerinde bulunan tablo ve sınıfların metodlarının açıklamalarının bulunduğu dökümantasyon kısmıdır.

- Uygulama Dökümantasyon (Application Documentation): Sistem üzerinde bulunan standart objelerin açıklamalarının yapıldığı dökümantasyon kısmıdır.

- HTML Yardım Dosyaları (HTML Help Files) : Dökümantasyonların çalıştırılacağı taslak sayfalarının bulunduğu kısımdır.

bulunmaktadır.



Şekil 17.10: Data Dictionary

X++ dilinin kullanıldığı objeler alt bölümler altında anlatılmaktadır.

3.7.1 Tablolar (Tables)

Tüm tablolar değerleme için standart olarak 3 adet değerleme metoduna sahiptir.

Tablo 4.2 Tablo Metodları

Metod Adı	Etkinleşme
ValidateDelete	Silme esnasında silme hareketlerini (delete actions) kontrol eder.
ValidateWrite	Kayıt kapanması aşamasında eğer kayıt satırı üzerinde değişiklik var ise.
ValidateField	Kayıt satırı üzerinde her bir alanın değişikliği ya da alandan çıkılması durumunda. Tek bir alanın kontrolü içinde kullanılabilir. Yasal değerler(Legal Values) alanında kodlama ile kurallar koyularak yapılmaktadır.

Bu metodlar ihtiyaç doğrultusunda tam ya da parçalı olarak üzerine yazılabilir.

Örneğin;

Müşteri riski 1000000'ü aştığında kullanıcıların bir uyarı alması istenmektedir.

Bu işlem Müşteri tablosu üzerinde metod Validatefield() ile yapılabilir.

```
boolean validateField(fieldId p1)
{
   boolean ret;
   ret = super(p1);
   switch (p1)
   {
      case fieldNum(CustTable, VATNum) :
       ret = TaxVATNumTable::checkVATNum(this.VATNum, this, p1);
         break;
      case fieldNum(CustTable, CreditMax) :
         if (this.creditMax < 0)
         {
            ret = checkFailed("Hata");
         }
         else
         {
```

```
            if (this.creditMax >= 100)
            {
                if (box::yesNo("Limit 100'I aştı Kabul ediyor
musunuz?",DialogButton::No) == DialogButton::No)
        ret = checkFailed("Kredi limit değiştirildi");
            }
          }
          break;
   }
   return ret;
}
```

Tablo 5.3 Diğer Tablo Metodları

Formlar üzerindeki veri kaynakları (Data Sources)	validateWrite, validateDelete
Form kontrolleri üzerinde (StringEdit, IntEdit etc.)	validate

3.7.2 Sınıflar (Classes)

Yeni bir metod yaratıldığında standart olarak 3 adet metod oluşur.

Class Declaration

Metodlar oluştuğunda aşağıdaki şekildedir:

```
public class Class name
{
        //Tüm değişkenlerin tanımlandığı yer
        //her bir objenin taşıdığı
        //örneğin:
        str text;
}
```

Class Declaration içinde tüm obje de bildirilecek sınıfın adı ve değişkenler belirlenir.

New

Metodlar oluştuğunda aşağıdaki şekildedir:

```
void New()
{
}
```

Her seferinde bir obje yaratmak için kullanılır. Bu metodu kullanarak objelerin değişkenlerine Dialog sınıf'larında olduğu gibi değer atanabilmektedir. Örneğin text isimli bir metin değişkeni atanmak isteniyorsa;

```
void New(str _text)
{
        text = _text;
}
```

_text yerel metin değişkeni ilgili özgün metoda aittir.

void metodun hiç bir değer döndürmediğini göstermektedir.

Finalize

Bir objenin kullanım işlemi tamamlandığında örneğin başka bir objeye bağlantı kurmadığında veya boşta beklediğinde sistem tarafından imha edilir.

Eğer bu objeyi sisteme bırakmadan yok etmek gerekirse **finalize** methodunu kullanmamız gerekir.

Bir objeyi yok etmenin bir başka yolu da objeyi Null dediğimiz boş değişkene atamaktır.

Örneğin

StopWatch sw; //StopWatch tipinde bir obje tanımlıyoruz

sw = new stopWatch(); //tanımladığımız obje üzerine stopwatch() ismindeki objeden bir obje türetiyoruz.

sw = Null; // eğer bu objeye refere den başka bir obje yok ise objemizi yok ediyoruz

Veya

sw.finalize(); // Objeyi refere eden başka bir obje olsa bile. Eğer objenin finalize

methodu üzerinde bir kod yazılmış ise bu kodunda çalışmasını sağlıyoruz

SINIF METODLARI (Class Methods)

Objenin metodlar ile olabileceği gibi, sınıflara ait metodlarında yaratılması mümkündür. Fakat bu kullanım geniş değildir. Command static kullanarak bu işlem gerçekleştirilebilir.

Sınıf metod kullanarak yukarıdaki metod yazılmak istenirse aşağıdaki şekilde olacaktır:

```
static void test method()
{
        print ”this is a test method”;
        pause;
}
```

Ve aşağıdaki şekilde açılabilir.

```
Class name::test method();
```

Sınıf metodlar aynı zamanda static metodlara atıfta bulunur ve dinamik metodların aksine static metod kullanımı öncesinde obje çağırımına gerek yoktur. Statik metoddaki üye değişkenlerin kullanımı mümkündür.

Statik metodlar Axapta'da sıkça kullanılır çünkü genellikle tablolarda tutulan saklanmış veri ile çalışmak istenmektedir ve bu nedenle üye niteliklerine ihtiyaç duyulmaz.

Main

Genelde kullanılan görevlerde(Job) olduğu gibi , belirli sınıf metodunu direkt olarak menü seçeneğinden çalıştırma şansına sahibiz. Bu metoda **Main()** denir ve aşağıdaki şekilde yazılmaktadır:

```
Static void Main(Args _args)
{
}
```

Bu metod obje örneği yaratmak ve gerekli üye metodları yaratmaktan başka iş yapmamalıdır.

Args'ı kullanmak gerektiğinde veriyi metoda aktarmak için kullanılır.

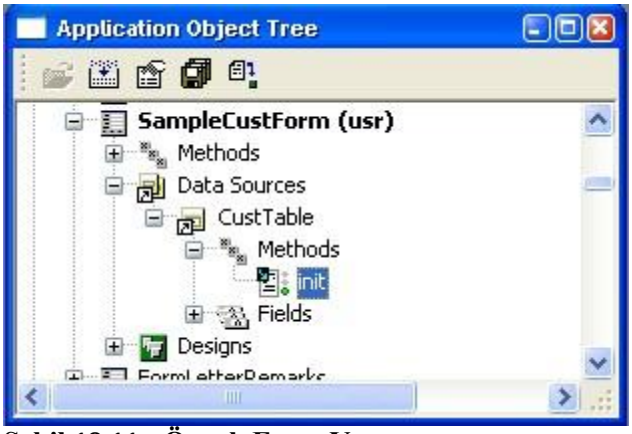
Bu metodu aynı zamanda AOT içerisindeki bir sınıfı vurgulamak ve sağ klik üzerinden ya da menü çubuğundan "Aç" komutunu seçmek için kullanabilirsiniz.

3.7.3 Formlar (Forms)

Formlar üzerinde sorgu yazmak raporlar üzerinde sorgu yazmaktan az ya da çok farklı değildir. Örneğin, eğer verilen bir alandaki belirli değeri olan kayıtları yüklemek için genellikle rapordaki ile aynı işlem yapılmaktadır.

Formun veri kaynağındaki(datasource) init metodunda, **QueryBuildRange** tipinde bir obje yaratılır ardından **value()** metodu kullanılarak kriter verilecek bir değer belirlenir.

Örneğin; eğer Müşteriler(CustTable) tablosunu baz alan basit bir form yapmak için veri kaynağındaki(data source) Init metodu aşağıdaki şekilde düzenlenmelidir.



Şekil 18.11: Örnek Form Yapısı

Init metodu içerisindeki kod aşağıdaki şekilde olmalıdır.

```
public void init()
{
   QueryBuildRange     queryBuildRange;
   ;
   super();
   queryBuildRange =
this.query().dataSourceNo(1).addRange(fieldnum(CustTable,AccountNum));
   queryBuildRange.value("4000");
}
```

Burada formun üzerinde yer alan very kaynağında(data source) addRange() metodunu Hesap kodu(AccountNum) alanına yeni kriter kullanmak için kullanıyoruz. QueryBuildRange metodundaki Value() ile bu kritere yeni değer atanmaktadır. Super() çağırılarak sorgunun doğru şekilde sıfırlandığından emin

olunmalıdır.

Form çalıştırıldığında sonuç Müşteriler tablosundaki belirli kaydın filtrelenerek görüntülenmesi olacaktır.

Eğer data setinde yer alan aktif kaydın kaldırılması isteniyor ise **First()**, **Last()**, **Next()** ve **Prev()** metodları kullanılabilir. Örneğin son kaydın kaldırılması isteniyor ise:

Datasourcename_ds.last();

Tablo 6.4 Metod Açıklamaları

Method Adı	**Yürütülme**	**Açıklamalar**
Active	Kullanıcı mevcut içinde yeni kayıt yaratmak için kullanır.	Super() çağırılarak yeni kayıt yaratılır.
Create	Kullanıcı veri kaynağında yeni kayıt yaratır.Örneğin, kısa yol Ctrl+N tuşu ile.	Super çağırılarak ekrandaki mevcut kayıttan önce yeni kayıt yaratılır. Mevcut kayıttan sonra bir kayıt üretmek için metodu "doğru" parametresinden sonra konulmalıdır.
defaultMark	Grid kontrolde kullanıcı sol üst köşeyi seçebilir.	Kayıtlar seçilerek grid görüntüde gösterildiğinde, standart olarak 1 değeri ile işaretlenirler. Bu işlem

		kaydın silinmesi ya da kopyalanması için kullanılır.
Delete	Veri tabanında yer alan kaydın silinmesi için kullanılır.	Super(), **validateDelete** ve eğer dönen değer doğru ise veri tabanındaki delete action'I harekete geçirecektir.
deleteMarked	Kullanıcı bir ya da birden fazla seçili kaydı silebilir. (ALT+F9)	Eğer hiçbir kayıt seçili değilse silme işlemi yapılmaz.
dislayOption	Kayıt gösterilmeden önce	**displayOption** kayıt yüklendikten sonra gösterilmeden önce bir kez kullanılır. Bu işlem kaydın renklendirilmesi ya da gri boyası verilmesi için kullanılabilir.
exeuteQuery	Verinin gösterilmesi için Form açılmıştır.	Super() init metodun çağırdığı sorguyu tamamlamış ve kayıtları göstermiştir.
Filter	Form üzerinde Filtreleme komutunu active eder	Eğer standart filtreleme özelliğine bilgi eklenmek isteniyor ise Filter metodu üzerinde kod satırları yazılır.
findValue	Form üzerinde Kayıt Bul komutunu active eder	Super() belirli bir değeri bulur ve and findRecord. Üzerindeki değeri olmasını sağlar.

First	Odak veri tabanındaki ilk kayıttır.	super() çağrısı veri tabanındaki ilk kaydadır.
Init	Form açılmıştır.	Veri tabanının özelliği üzerine, Super() Form yüklendiğinde verinin gösterilmesini sağlar.
initValue	Yeni bir kayıt yaratılmıştır. Kayıt içerisine ilk değerler yerleştirilir.	Super() çağrısı **initValue** metodunu aktive eder ve kayıt içerisine ilk değerler yerleştirilir. Bu metodda tipik olarak yeni kayda değer atanır.Sistem kayıt kullanıcı tarafından bir ya da birden fazla alanı değiştirilene kadar kaydı değerlendirmez.
Last	Odak veri tabanındaki son kayıttır.	super() çağrısı veri tabanındaki son kaydadır.
Leave	Odak veri tabanındaki yeni kayda ya da yeni veri tabanınadır.	**leave** kaydın gereksiz değişiklikleri içindir.Super() bir şey çağırmaz ve sadece bilgi verme amaçlıdır.
leaveRecord	Odak veri tabanındaki yeni kaydadır.	
linkActive	Kullanıcı veri tabanı ilişkili diğer veri tabanı ile birlikte formdaki yeni kayda kaydırılmıştır.	Super()formun bağlı olduğu veri tabanındaki **executeQuery'yi** çağırır. Bu metod sadece veri kaynağı üzerindeki **LinkType** özelliği **Yes** olan iki veri kaynağı için geçerlidir.
Next	Odak veri tabanındaki	super() çağrısı veri tabanındaki

	sonraki kayıttır.	sonraki kaydadır.
Prev	Odak veri tabanındaki önceki kayıttır.	super() çağrısı veri tabanındaki önceki kaydadır.
Print	Dosya altındaki Yazdır fonksiyonunu aktive eder.	Otomatik raporlama özelliklerini aktive eder.
Prompt	Ax üzerindeki filtreleme fonksiyonunu aktive eder.(CTRL+F3).	Super() çağrısı standart form üzerindeki sınırlı sorgu kriterini aktive eder.(**sysQueryForm** form).
Refresh		Super() çağrısı ekrandaki tüm verileri veri tabanındaki tüm alanlar ile günceller. **refresh refreshEx'**i çağırır.
refreshEx	Formlar kayıtlar seçildiği zaman aktive edilir.	**refreshEx** metodu **refresh** metodunun genişletilmiş versiyonudur.Tek parametrelidir ve bir tek satırı güncelleştirir.
removeFilter	Kullanıcının **Filtreleme İptal** butonuna basması.	Super() methodunun çağırılmasıyla form üzerindeki veri kaynağında bulunan **init** metodu ile tanımlanan bütün veri filtreleme modifikasyonları iptal eder.
reRead	Sistem tarafından aktivasyonu gerçekleşmez.	super() metodu mevcut kaydın tekrar okuma işlemini gerçekleştirir.
Research	Sistem tarafından aktivasyonu gerçekleşmez.	super() metodu çağırıldığında **init** metodu üzerinde tanımlanan veritabanı filtreleme işlemini

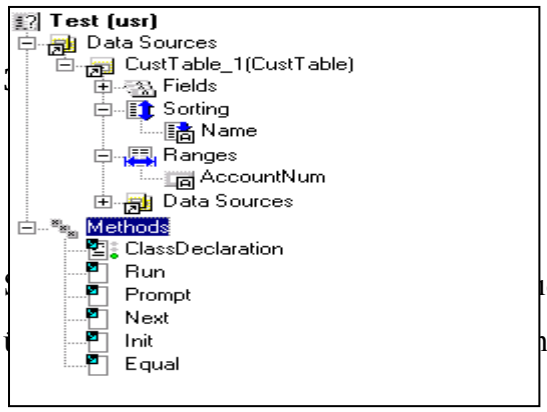
		tekrarlar. **executeQuery** metodundan farklı şekilde bu metod sorgu üzerindeki filtreleme ve sıralama işlemlerin kaybolmasını önler. **research ile executeQuery karşılaştırması.** Metod veya iş (Job) yolu ile kayıt eklenmiş kayıtlar var ise ve formun bilgilerinin tekrar yüklenmesi gerekiyor ise **research** metodunu kullanmalıyız. Eğer sorguyu değiştirip diğer kayıtlarında görüntülenmesini istiyorsak **executeQuery** metodunu kullanmalıyız.
validateDelete	Kayıt silinmesi durumunda.	super() metodu tablo üzerindeki **validateDelete** metodunu çağırır. Bu metodu kullanılarak kendinize özgü veri doğrulama işlemlerinizi gerçekleştirebilirsiniz.
validateWrite	Yeni veya güncelleştirilmiş bir kaydın yazılması durumunda.	super() metodu tablo üzerindeki **validateWrite** metodunu çağırır. Bu metodu kullanarak doğrulama işlemlerinde kendi özel kontroller oluşturulabilir.

Write	Yeni veya güncelleştirilmiş bir kaydın yazılması durumunda.	super() metodu **validateWrite** metodunu tetikleri ve (eğer metod true değeri döndürürse) veritabanı yazma işlemini üslenir.

3.7.4 Haritalar (Maps)

Daha önceki bölümlerde de görüldüğü üzere tablolar başka tablolardan özellik miras (inherit) alamaz. Burada sorulması gereken soru : eğer iki tablo neredeyse aynı yapıdaysa her iki tablo içinde ayrı ayrı metodu yazmamız gerekmektedir.

Harita (Map) kullanıyorsak bu soruya cevabımız hayır olacaktır. Haritalar (Maps) AOT altında "Data Dictionary" altında yer almaktadır. Benzer tabloları birleştirme görevi üstlenen haritaların karıştırılmaması gereken en önemli hususu bu haritaların üzerlerinde veri barındırmayışıdır.



ıdır ve tıpkı formlar ve raporlar gibi AOT

ı kullanıldığını söyleyebiliriz.

Veri arama yollarından biride **while** ile birleştirilmiş **select** cümleciği kullanmaktır**.** Sorgular (Queries) daha avantajlı seçenek olarak görülmektedir. Sıralama filtreleme konularında çok daha özgürlük sağlarlar.

Şekil 19.12: Örnek Sorgu Yapısı

Temelde sorgu bir veya birden fazla veri kaynağı ile sorgu sırasında çalışacak metodlardan oluşmaktadır. Örneğimizde, CustTable sorgumuzun veri kaynağı olacak, veri Name (Adı) alanı üzerinden sıralama yapılarak ve AccountNum (Hesap Numarası) üzerinden filtrelenerek görüntülenebilecektir.

Daha öncede değindiğimiz gibi sorguların avantajı kullanıcının sorguyu çalıştırma anında sorguya müdahale edebilmesi.

Çalıştırma (Execution)

Sorgular kendi başlarına çok ilgi çekici değildirler veriyi verilen sıralama doğrultusunda aramaktan başka bir işlem yapamazlar. İlgi çekme işlemi kod içerisinden çağırılıp çalıştırılınca başlar.

Bu işlemleri yapabilmek için kernel sınıflarından biri olan QueryRun tipinde bir obje tanımlamamız gerekmektedir. Bu tanımladığımız objeyi Query tipinde **Test** adında bir objeye iliştiriyoruz. Bu bahsettiğimiz kodun yazılım şekli aşağıdaki gibidir:

Örnek:

```
CustTable     ct;
Query         q =new Query('Test');
QueryRun      qr=new QueryRun(q);

if (qr.prompt())
{
        while (qr.next())
        {
        ct= qr.get(TableNum(CustTable));
        print ct.Name;
        }
```

```
}
pause;
```

Sorgu (Query) oluşturmak için AOT u kullanmak zorunda değiliz kod içerisinden de kendimize sorgular oluşturabiliriz.

Bu işlemi yapmak için oluşturmamız gereken kod bloğu aşağıdaki gibidir:

Örnek:

```
Query                   q;
QueryRun                qr;
QueryBuildDataSource    qbd;
QueryBuildRange         qbr;
q;      =new Query();
qbd     =q.AddDataSource(TableNum(CustTable));
qbr     =qbd.AddRange(FieldNum(CustTable,AccountNum));
qbd.AddSortField(FieldNum(CustTable,Name));
qr      = new queryRun(q);
```

Yukarıdaki örnek ile **QueryBuildDataSource** ve **QueryBuildRange** kernel sınıflarının kullanımını görmekteyiz. **Value()** metodu **QueryBuildRange** sınıfı içerisinde yer almaktadır. Bu metod hem belirtilen alana değer girmek için hemde belirtilen alandan bilgi almak için kullanılabilir.

Birleştirme (Join)

Birleştirmeyi (**join)** tabloları birbirine iliştirmek için kullanırız. Bu komut **select** e benzer şekilde kullanılır.

Sıradaki örnek bir müşterinin hareketlerinin nasıl aranacağını göstermekte:

Örnek

```
CustTable CT;
CustTrans CTR;

While select AccountNum,Name

From CT
Order by AccountNum
Join* from CTR

Where(CTR.AccountNum==CT.AccountNum)

{ }
```

Where deyimini **join** den sonra doğru bir şekilde yazmak çok önemlidir, çünkü verinin birleştirilmesinden bu deyim sorumludur.

Join verilen talimatlar doğrultusunda farklı çalışma şekilleri gösterir.

Aşağıdaki tablo ilgili özellikleri listelemektedir.

Tablo 7.5 Sorgu Metodları

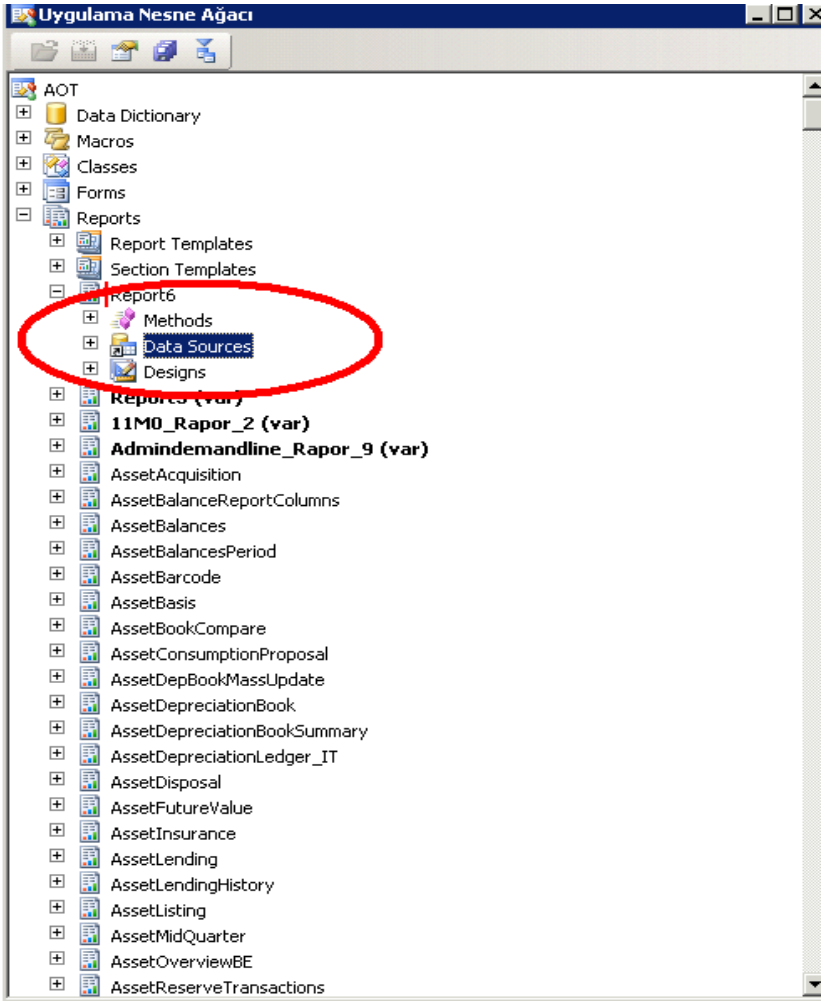
Komut	**Açıklama**
Inner	Müşteri seçimi sadece o müşterinin hareketi mevcutsa yapılır. Eğer **join** den önce hiçbir ön ek kullanılmadıysa, **inner** otomatik olarak kullanılır. Kişinin bütün hareketleri seçilir.
Outer	Müşteri üzerinde hareket olup olmamasına bakmadan o müşteri ile ilgili kayıtları getirir. Kişinin bütün hareketleri seçilir.
Exists	Müşteri sadece üzerinde bir hareket bilgisi kayıtlı ise seçilir ama o kişiye ait sadece bir hareket kaydı getirilir.
Notexists	Müşteri üzerinde hiç hareket işlemi yok ise getirir. Hiç hareket bilgisi getirilmez.

3.7.6 Raporlar (Reports)

AOT içindeki nesnelerden biri de Report nesnesidir. Report nesnesi sayesinde, Axapta içerisinde rapor almayı sağlayan uygulamalar gerçekleştirilebilir. Report nesnesi rapor alımı için özel olarak tasarlandığından dolayı, kullanımı kolay ve kendine has özellikleri olan bir nesnedir. Axapta üzerinde tutulan tüm bilgiler report nesnesi aracılığı ile ekrana, yazıcıya gönderilebilir.

Yeni bir rapor oluşturmak için AOT üzerinde, Reports sekmesi üzerindeyken sağ tuş New Report ile yeni bir rapor oluşturabiliriz. Rapor üzerindeki her sekmeyi ve özelliklerini anlatalım.

Report nesnesini ilk oluşturduğunuz zaman, güncelleyebileceğimiz Methods, DataSources ve Design bölümleri açılır.



Şekil 20.13: Rapor Örneği

Varsayılan olarak, report nesnesinin tüm bölümlerinden kullanılabilecek değişkenlerin tanımlanabileceği ClassDeclaration metodu açılır. Bu metod içerisinde sadece global olarak kullanabilecek değişkenler tanımlanabilir. Burada aynı zamanda üzerine yazılabilir metotlar veya global olarak kullanabilinecek kişisel metotlar yazabilir. Üzerine yazılabilir metot olarak kullanabileceklerden bazıları:

Init: Rapor ilk çalıştırıldığı anda yapılması istenilen işlemlerin kodlarını buraya yazılabilir. Init, her raporda varsayılan olarak çıkan diyaloglar çıkmadan önce çalışır.

Prompt: Her rapor çalıştırıldığında, eğer veri kaynaklarına ait kriter / kriterler girilecekse bu bilgiler tanımlandıktan sonra raporun ekrana, yazıcıya, mail olarak veya belirtilen bir dosya tipi olarak gösterilip / gösterilmeyeceği konusunda bir diyalog açılır. Eğer bu diyalog kullanıcıya göstermek istenilmezse, bu metod kullanılır. Bu metodun super()'i çağırılmazsa, ilgili diyalog görünmeyecektir. Bir raporu ASCII, PDF, RTF, HTML dosya formatı olarak kaydedilebilir.

Run: Bütün diyaloglar, rapor tasarımı çalıştıktan sonra burası aktif olur.

Fetch: Veri kaynaklarında tanımlanmış olan tabloların bilgileri alınıp, rapora gönderilir. Varsayılan olarak zaten Axapta bu işlemi gerçekleştirmektedir. Ancak veri alıp – gönderimi sırasında özel olarak bir kod yazmak, kontrol yapmak istendiğinde bu metot aracılığı ile gerçekleştirilir.

Send: Veri kaynaklarından alınan, eğer varsa fetch metodundan gelen kayıtları rapora gönderir. Send sadece okunan kayıtları raporun görsel bölümüne gönderir. Bu gönderim sırasında bazı kontroller veya özel uygulamalar yapılacaksa kullanılır.

DataSources:

Rapor üzerinde kullanılacak olan tabloların burada veri kaynağı / kaynakları olarak tanımlamaları yapılır. Varsayılan olarak gelen DataSources sekmesinin altında, Query sekmesi, bunu altında ise ikinci bir DataSources sekmesi gelir. Query sekmesi raporun komple tüm sorgusunu temsil eder. Kod içerisinden buradan tanımlanan sorguya Query bilgisi ile ulaşabiliriz.

Query'nin özellikleri;

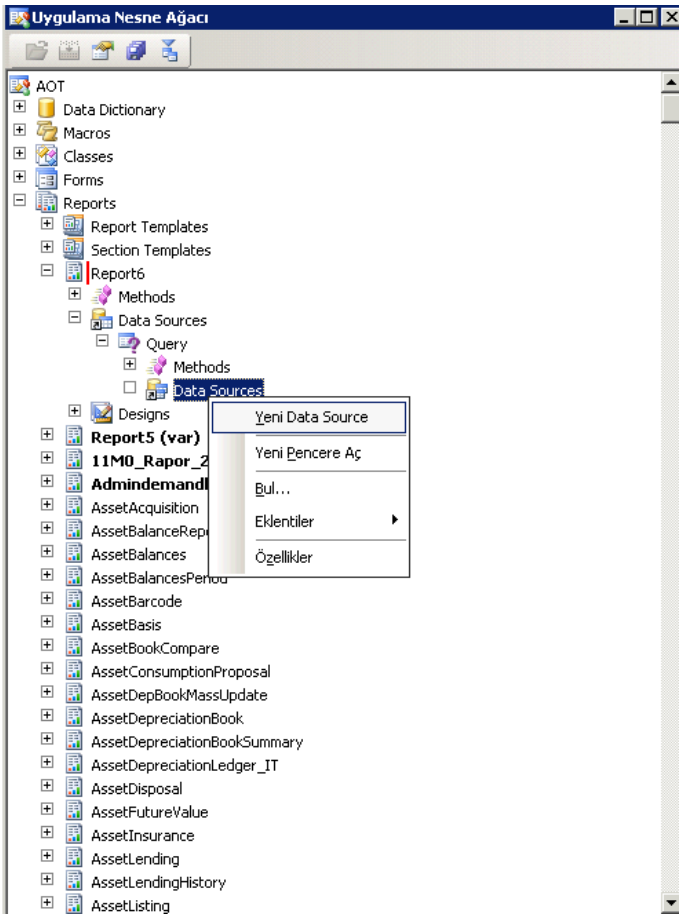
Name: Sorgunun adı tanımlanır.

Title: Sorgunun görünüm adı tanımlanır.

UserUpdate: Veri kaynakları tanımlarındaki kriterlere kullanıcılar giriş yapabilir. Varsayılan olarak Yes gelmektedir.

Interactive: Veri kaynaklarına rapor alımı sırasında kullanıcılar başka sahaları kriter girişi için açabilir veya rapor alımı sırasında başka bir veri kaynağı sorguya ekleyebilir. Bu özellik varsayılan olarak Yes gelmektedir. Bu özellik kaldırılmak istenirse No olarak güncellenmesi gerekmektedir.

DataSources sekmesi üzerindeyken sağ tuş ile **New DataSource** denilerek yeni bir veya daha fazla veri kaynağı oluşturulabilir. Eğer birden fazla veri kaynağı oluşturulursa bu kaynaklar birbirleri ile arasında ilişkili olabilir veya ilişkisiz olabilir.



Şekil 21.14: Rapor Örneği

Yeni bir DataSource'un özellikleri:

OrderMode: Varsayılan olarak bu seçenek Order by modunda gelir. Bu seçenek ile, DataSource'un Sorting sekmesinde tanımlanan sahaya göre sıralama gerçekleştirir. Eğer Group By seçeneği kullanılırsa, yine Sorting sekmesindeki vermiş olunan sahaya göre gruplama gerçekleştirir.

Sorgular bu şekilde görsel ara yüzden güncellenebileceği gibi, kod içerisinden de güncellenebilir. Bu konu ayrı bir başlık altında ayrıntılı olarak anlatılacaktır.

DataSource altındaki sekmeler;

Fields: Bu veri kaynağında gösterilen tablonun tüm sahaları Field sekmesinde görülebilir. Otomatik olarak tablo adı verildiği zaman burası güncellenir.

Sorting: Veri kaynağındaki bilgilerin hangi sahaya göre sıralı olacağı burada belirlenir Eğer veri kaynağı Group By olarak belirlenirse, burada verilen saha isimlerine göre gruplama yapar. Yeni eklenilen sıralama sahasının özellikleri;

Field: Hangi sahaya göre sıralama yapılabileceği bilgisi.

Ordering: A-Z veya Z-A sıralaması tanımlanır.

AutoHeader: Bu sahanın bulunduğu tablonun bilgileri ekrana basıldığı anda, bu sahanın başlık olarak atılıp – atılmayacağı bilgisi tanımlanabilir. Varsayılan olarak No gelmektedir.

AutoSum: Bu sahanın bulunduğu tablonun bilgileri ekrana basıldığı anda, bu saha sayısal ise, son bilgi olarak ekrana basılıp - basılmayacağı bilgisi tanımlanabilir.

Varsayılan olarak No gelmektedir.

Ranges: Veri kaynakları üzerinde kriterler vermek istenildiğinde burada tanımlanabilir. İsterseniz bu kriterlerin değerlerini kullanıcıya verdirebilir, istenildiğinde varsayılan olarak rapora tanımlanabilir. Range sekmesi üzerindeyken sağ tuş **Yeni Range** seçeneği ile yeni bir kriter eklemesi yapılabilir. Range olarak verilen sahaların değiştirilebilir özellikleri;

Field: İlgili tablonun hangi sahasının kriter olarak tanımlanacağı bilgisi tanımlanır.

Value: Bu kriterin rapor çalıştırıldığı anda varsayılan olarak alacağı değer tanımlabilir.

Status: Kriterin nasıl bir amaca yönelik kullanılacağına göre buradaki seçeneklerden biri tanımlanır. Seçenekler;

Open: Kriter açık olur, istenildiği gibi bilgiler girilebilir.

Lock: Kriter kilitli olur. Kullanıcı böyle bir kriter olduğunu görür ancak müdahele edemez.

Hide: Kriter gizlenir, kullanıcı bu saha ile ilgili hiçbir bilgiye sahip olmaz.

DataSources: Veri kaynağına bağlı başka bir veri kaynağı tanımlamak istendiğine aynı şekilde burası kullanılır. Üst bir veri kaynağına sahip olan bir kaynağının ek olarak sahip olduğu özellikler vardır. Bunlar;

FetchMode: 1:n veya 1:1 olarak ilişki seçimi yapılabilir.

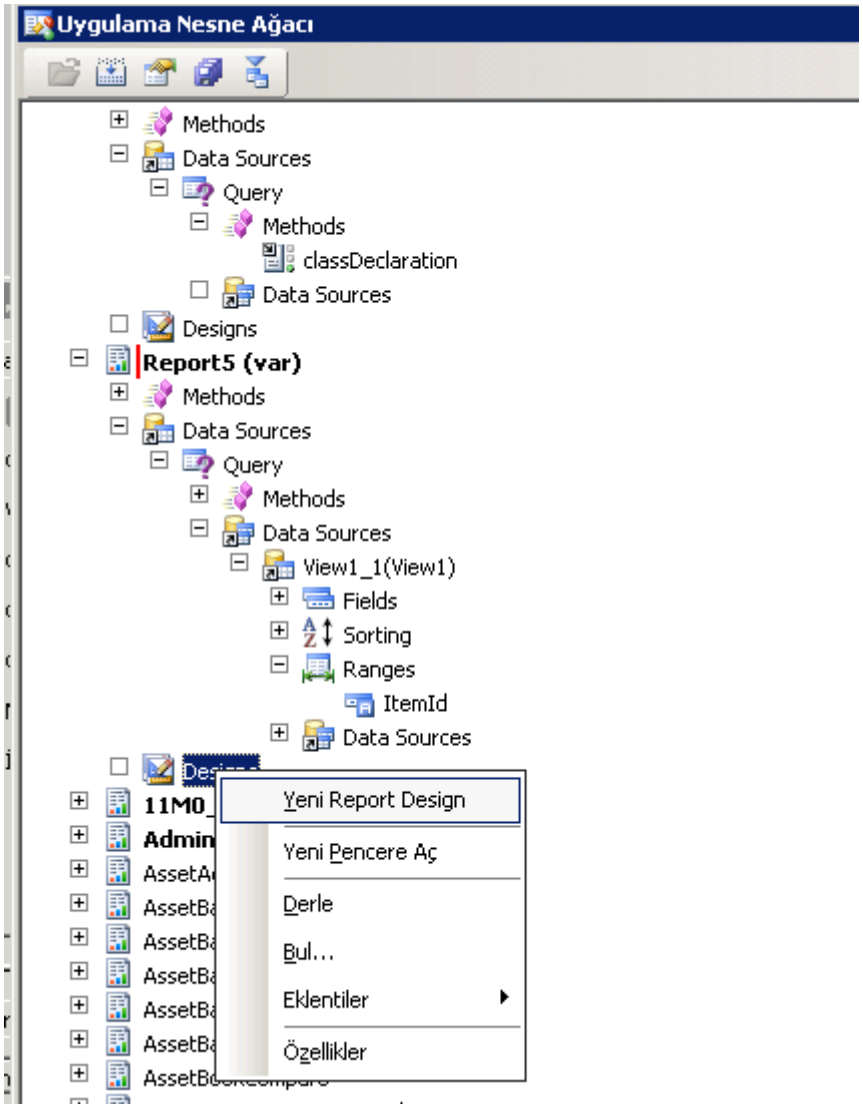
JoinMode: İki veri kaynağı arasındaki ilişki tipini InnerJoin, OuterJoin, ExistsJoin, NoExistsJoin olarak tanımlanabilir.

Relation: Tanımlanan veri kaynağı üzerinde başka bir veri kaynağı tanımlanmış ise ve DataSource'un Relations özelliği Yes olarak belirtilmişse, iki tablo arasında ilişkiler burada otomatik olarak tanımlanır. Eğer isterseniz iki tablo arasındaki ilişkiyi manuel olarak da buradan tanımlanabilir.

Methods: Veri kaynaklarına özel kodlar yazmak istenildiğinde buradaki üzerine yazılabilir metotlar kullanabileceği gibi, özel olarak kişisel metotların da ihtiyaca yönelik kodlar için geliştirilebilir.

Design:

Rapor üzerinde tanımlanacak olan tasarımsal işlemlerin tanımları bu sekme üzerinde tanımlanır. Design sekmesi üzerinde sağ tuş ile New Report Design'ı seçildiğinde, tasarım yapılacak ortam açılır.



Şekil 22.15: Rapor Dizaynı

Bu tasarımda belirleyebileceğimiz özellikler:

AutoDeclaration: Varsayılan olarak No gelir. Yes olarak belirlendiğinde, kod içerisinden rapor tasarımına erişim sağlanır.

Caption: Raporun başlığı bilgisi buradan verilebilir. Boş bırakılırsa, raporun nesne adı olarak bu değeri alır.

Description: Raporun tanımı buraya verilebilir.

Ruler: Rapor tasarımında kullanacağınız birim bilgisi verilebilir. Char, cm ve inch olarak tanımlayabilirsiniz.

ReportTemplate: Raporun tasarım olarak nasıl bir template altında olacağı bilgisi verilebilir.

Font: Rapor üzerindeki text bilgilerin hangi font'da olacağı bilgisi verilebilir.

FontSize: Rapor üzerindeki text bilgilerin büyüklüğü bilgisi verilebilir.

Italic: Rapor üzerindeki text bilgiler eğik olarak görüntülenebilir.

Underline: Rapor üzerindeki text bilgiler altı çizgili olarak görüntülenebilir.

Bold: Rapor üzerindeki text bilgiler kalın olarak görüntülenebilir.

Orientation: Raporun dikey veya yatay olarak çıkmasını sağlar. Varsayılan olarak auto gelir ve ekrandaki bilgilerin genişliklerine göre Axapta'nın otomatik olarak ayarlanmasını sağlar.

Rapor tasarımı ilk oluşturulduğu anda, AutoDesignSpecs isimli rapor tasarımının yapabileceği ortam otomatik olarak, veri kaynaklarına göre tanımlanmış şekilde gelir. Eğer bu ortamda rapor tasarımı yapmak istenmez ise, ReportDesign

üzerindeyken sağ tuş ile **Tasarım Oluştur** denildiğinde GenerateDesign isimli bir rapor tasarım bölümü açılır. Bu ikisinde de rapor tasarımı, raporda görünecek sahaların tanımlanması gibi işlemler gerçekleştirilir. Bu ikisi de temelde aynı amaca hizmet etse de aralarında farklılıklar mevcuttur. Eğer GenerateDesign oluşturulursa, AutoDesignSpecs'deki tanımlar geçersiz olur. Örneğimiz GenerateDesign üzerinde yapılacaktır. Generate Design üzerinde, çıktı olarak gösterilecek bilgiler bazı bölümlere ayrılır.

Bu bölümler Generate Design üzerinde sağ tuş ile yeni bölümünden oluşturulabilir. Bu bölümler;

Prolog: Rapor ilk çalıştığında bu bölümde belirtilen tasarımlar görünür. Burası tek başına bir bölümdür.

Page Header: Prolog ve Epilog dışındaki raporun her sayfasının başlığında görünecek bilgileri burada tanımlanır.

Sections: Rapor sayfalarının gövdesini oluşturacak bölümler burada tanımlanır.

Page Footer: Prolog ve Epilog dışındaki raporun her sayfasının sonunda görünecek bilgiler burada tanımlanır.

Epilog: Rapor çalışması sonlandığında bu bölümde belirtilen tasarımlar görünür. Burası tek başına bir bölümdür.

Header: Sadece raporun ilk sayfasının başlığında görünecek bilgiler için kullanılabilir.

Footer: Sadece raporun son sayfasının sonunda görünecek bilgiler için kullanılabilir.

Programmable Section : Program içeresinden istege bağlı olarak çalıştırılan kısımdır. Element.execute(int _number) fonksiyonuyla çağrılır. _number değişkenine çalıştırılmak istenen Programmable Section"a ait ControlNumber değeri set edilir.

Bu değer Programmable Section sağ tuşNitelikler>Controlnumber ile görüntülenebilir.

Her bir bölüm grubu(section)'da kendine has footer ve header'a sahiptir. Böylelikle, görünüm olarak sayfanın sonu yada başı olmasa bile her bir bölüm grubuna ait başlık ve sonuç bölümü oluşturulabilir.

Generate Design'ı oluşturulduğu anda, sistem otomatik olarak veri kaynağında yapılan tanımlara göre bölümleri oluşturur. Eğer istenilirse bu otomatik tanımları silinip, tanımlar manuel olarak oluşturulabilir.

Her bölüm grubunun kendine ait üzerine yazılabilir metodu olduğu gibi, bölüme özel kullanılacak kişisel metotlar da yazılabilir. Her bölüm grubunda yazılabilecek metot;

ExecuteSection: Rapora basım için her bölüme geldiği anda bu metot aktif hale gelir. Örneğin, bölüme bir kayıt gönderileceği zaman, bazı kontrollerden geçirmek veya uygulamalar yazılmak istendiğinde burası kullanılabilir. Her bölüm super() çağrıldığında aktif olur.

Her bölüm grubu bir tablo ile tanımlanmalıdır. Bunu cursor olarak düşünebiliriz. DataSource'da tanımlanan tablo/tablolar kullanabileceği gibi, DataSource'da olmayan bir tablo ile ihtiyaca yönelik yeni bir bölüm grubu açabilir.

Bölüm grupların temel özellikleri:

Section Group'ların nitelikleri;

Name: Section Group'un adı bilgisi verilir.

Table: Section Group'un hangi tablo bazlı çalışacağı bilgisi verilir.

DataField: Table özelliğinde verilen tablonun herhangi bir sahası bazlı tanıtım yapılmak istenirse bu bilgi verilir.

Body, Header, Footer gibi bölümlerin kendine has özellikleri olsa bile genel anlamda benzer özelliklere sahiptir. Bu özelliklerden bazıları;

Name: Bölümün ismi tanımlanır.

AutoDeclaration: Bölüme global değişken gibi kod ile ulaşabilmek için bu özellik Yes olarak işaretlenir. Varsayılan olarak No gelir.

Top: Bölümün üst konum bilgisi verilir.

Bottom: Bölümün alt konum bilgisi verilir.

Height: Bölümün yükseklik bilgisi verilir.

LineAbove: Bölümün üst kısmının basım şekli belirlenebilir.

LineBelow: Bölümün alt kısmının basım şekli belirlenebilir.

LineLeft: Bölümün sol kısmının basım şekli belirlenebilir.

LineRigth: Bölümün sağ kısmının basım şekli belirlenebilir.

Font: Bölümün font bilgisi verilir.

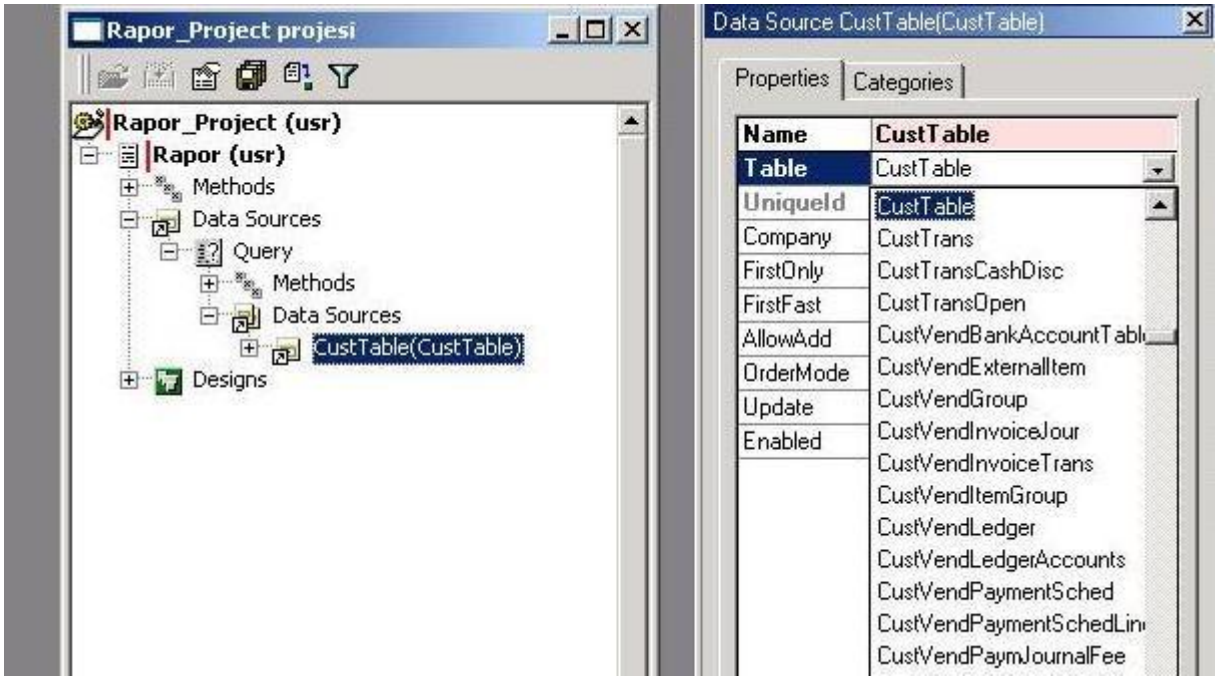
FontSize: Bölümün font büyüklük bilgisi verilir.

Italic: Bölümün sağa yatık bilgisi verilir.

Bold: Bölümün koyu bilgisi verilir.

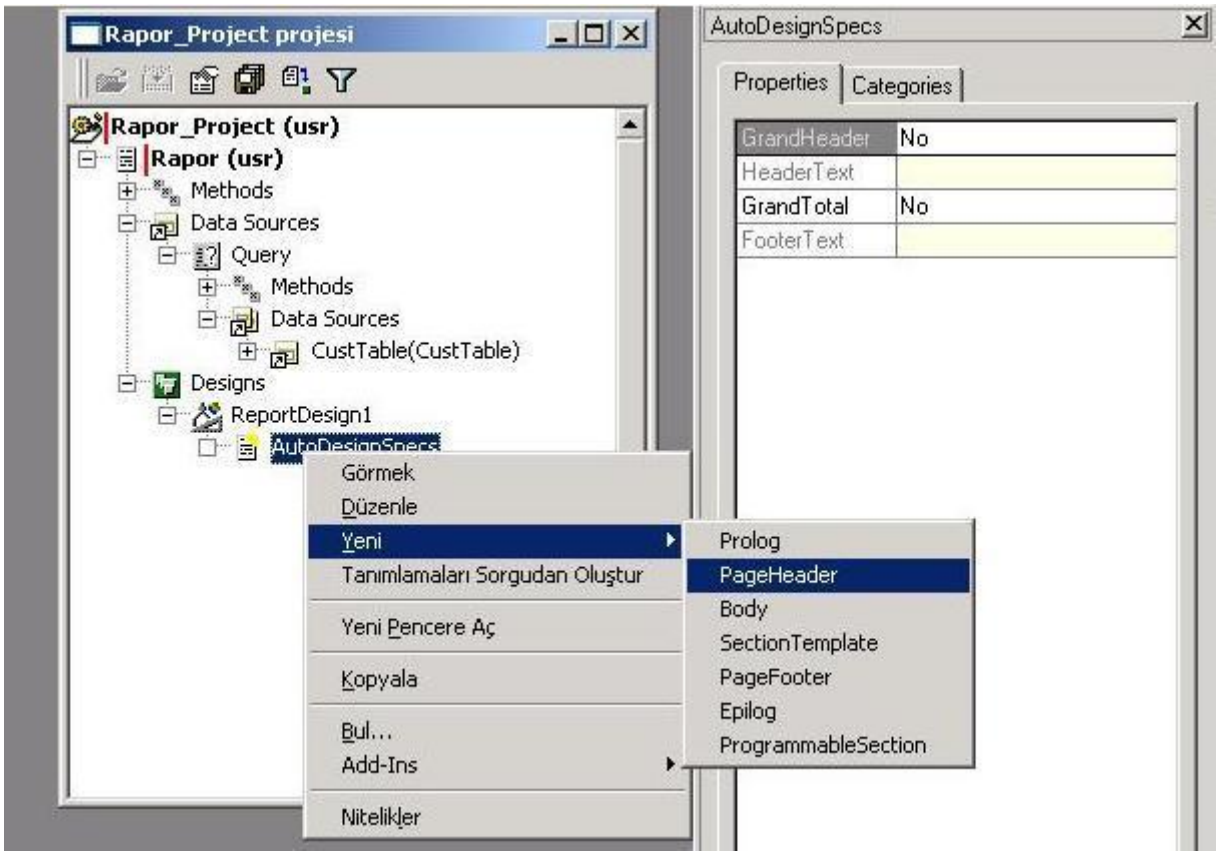
Underline: Bölümün altı çizili bilgisi verilir.

Müşteri bilgilerini sergileyen basit bir rapor yapmak istendiğinde, rapor için ilk olarak aşağıdaki adımları gerçekleştirilebilir. Projects>Shared>Yeni Projects ile yeni bir proje oluşturulur. Oluşturulan projede sağ tuş Yeni>Report ile yeni bir rapor oluşturulur. Raporda sağ tuş Query>Datassources>Query>Datassources>Yeni Data Source ile yeni bir data source oluşturulur. Data source Table özeliğine CustTable atanır.



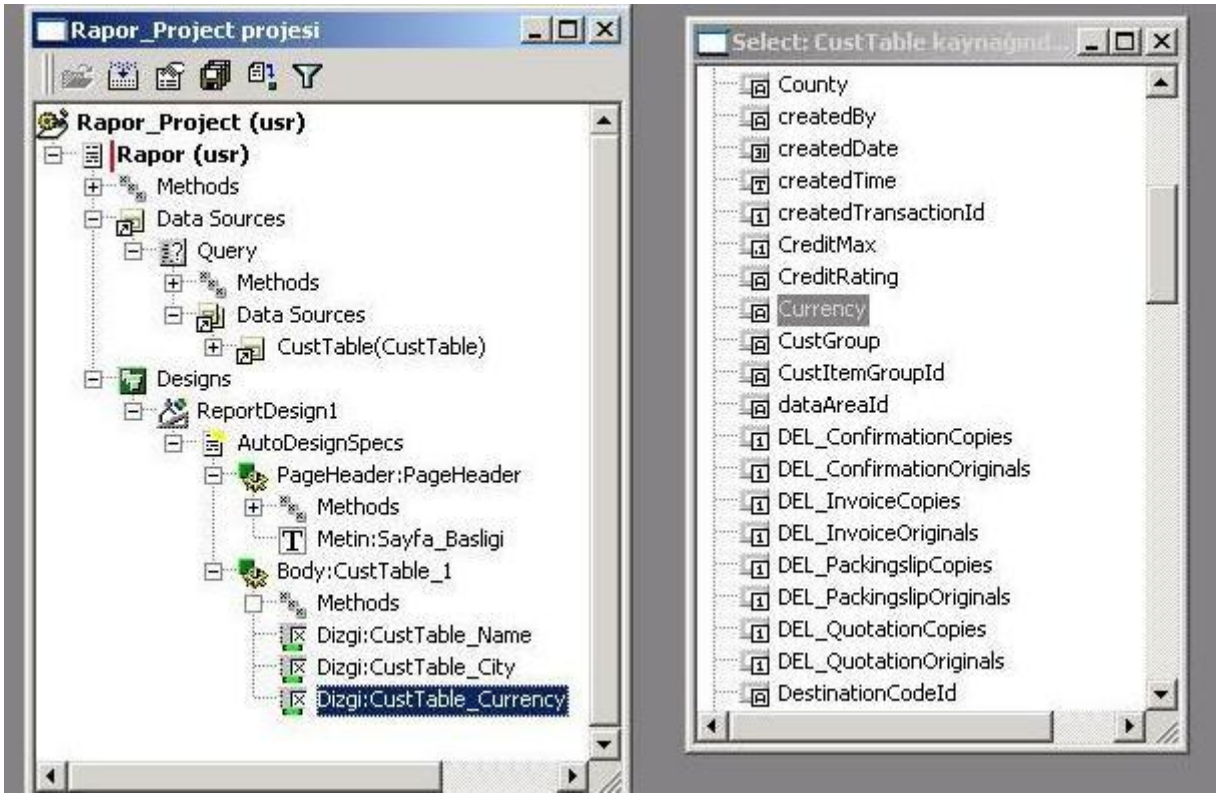
Şekil 23.16: Geliştirme Özellikleri

Report>Design>Yeni Report Design seçerek rapor dizaynı oluşturulur. Oluşturulan rapor dizaynında AutoDesignSpecs"e sağ tuş ile tıklandığında rapor üzerine koyabilecek bölümlerin listesini görülür.



Şekil 24.17: Rapor Dizaynı Oluşturma

Bu bölümlerden page header ve body"den birer adet eklenebilir. Page header üzerindeyken sağ tuş Yeni Kontrol>Metin ile rapora metin tipinde bir nesne koyulabilir. Metin nesnesinin text özeliğine "Müşteri Raporları" yazalım. Body üzerindeyken sağ tuş YeniKontrol>Cust Table kaynağından alan seçelim. Karşımıza çıkan listeden istediğimiz alanları body"nin içine sürükleyelim. Projemizde örnek olarak Name, City ve Currency alanları seçilmiştir.



Şekil 25.18: Rapora Alan Ekleme

Raporu çalıştırmak için rapora sağ tuş ile tıklayıp "açık" seçildiğinde. Rapor görüntüsü aşağıdakine benzer bir yapıda olacaktır.

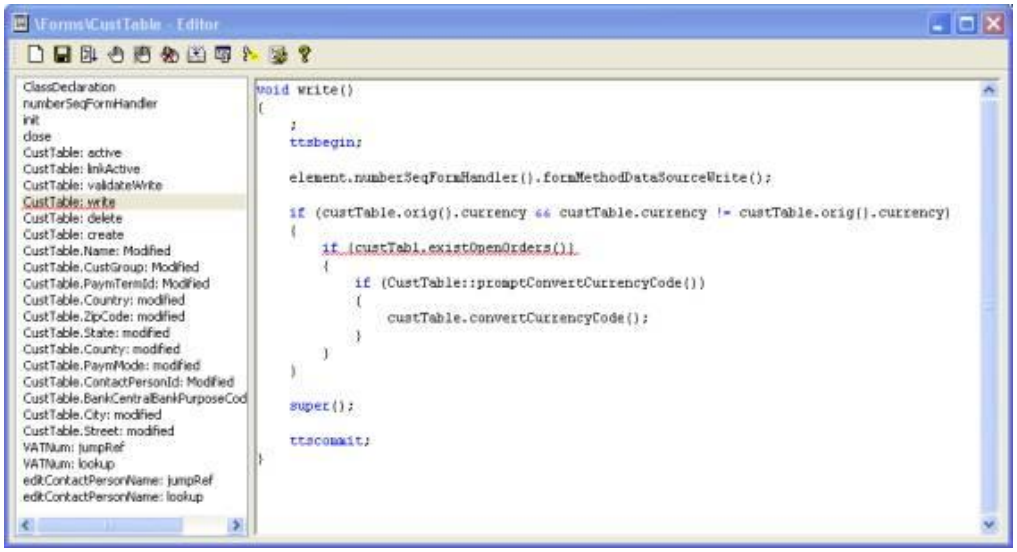
Adı	Şehir	Para birimi
Light and Design	Frederikssund	EUR
The Bulb	Madrid	EUR
The Bright Idea	Wien	DKK
Office Lights Inc.	Bruxelles	EUR
The Specialist	Dublin	EUR
Office Supplies Inc.	Oslo	EUR
Furniture World	Den Haag	EUR
Office Design Inc.	Berlin	EUR
The Warehouse	Copenhagen	EUR
Habitat	Vedbaek	EUR
The Lamp Shop	Roskilde	EUR

Şekil 26.19: Rapor Çıktısı

3.8 Geliştirme Editörü (Code Editor)

X++ Editorü küçük kodlamalar ile değişikliklerin geliştirilmesini sağlar. Metodlar sadece drag-drop ile kopyalanabilir.

Microsoft Axapta kolay kodlamayı sağlamak için, seçenekleriniz hakkında sizi sürekli bilgilendirir. Bir metoda çağrı yaparken bir tool-tip yardım size parametre bilgisini gösterir ve bir class instance'ı yaratırken bir seçim listesi sayesinde sınıfın'ın metodlarını görebilir, bunlardan birini seçebilirsiniz.



Şekil 27.20: Form Üzerinde Geliştirme

Yazma anında syntax check vardır. Hataların altı otomatikman çizilir. Formlar, sınıflar, raporlar, sorgular ve tablolar gibi çeşitli nesnelere kendi X++ kodunuzu ekleyebilirsiniz.

3.9 Hata Ayıklama (Debugging)

Axapta Debugger(Hata ayıklayıcı) adında güçlü yazılım geliştirme aracı ile donatılmıştır. Debugger tanısal amaçlara hizmet eden diğer programın düzenlenmesinin kontrolü için kullanılan özel bir program olarak tanımlanabilir. Örneğin; X++ programlarında hataları (bugs) bulmak için.

Axapta'nın Debugger'ı IDE (Integrated Development Environment) editor penceresi yolu ile etkileşimli hata ayıklamayı da sağlamaktadır. Debugger ile;

- Bir kod parçacığının üzerinde "içe" ya da "üzerinden" fonksiyonları ile atlanabilmektedir.
- Kursör yada kesme (breakpoint) ile programın belirli bir bölümüne gitmek ve işlemi durdurmak
- Değişken penceresinden programın yürütülmesi esnasında değişkenlerin adı, kapsamı ve değerleri görülebilmektedir.
- Yürütülen kod yığını (call stack) izlenebilmektedir.
- Sistem statüsü izlenebilmektedir.
- Kod satırlarının numaraları görülebilmektedir.

Kesme (Breakpoint)

Kesmeler X++ programında yürütmenin durdurulması için gerekli spesifik noktalara konulur.

Debugger penceresi 4 bölmeye ayrılmıştır;

Değişkenler (Variables)

Değişkenlerin mevcut yöntem yığını içerisinde yer alan değerlerini gösteren ekrandır. Duruşlar arasında değişkenler değiştiğinde değişkenler farklı bir renk alacaktır ki değişkenin değiştiği daha kolayca anlaşılmış olsun.

Değişkenler ekranında aynı zamanda değişkenin değerinin düzenlenmesi de mümkündür.

Yöntem Yığını (Call Stack)

Yöntem yığının fonksiyonunu/çağırdığı metodları göstererek kullanıcıların işlem anında hangi fonksiyon /metodun çağırıldığını görmesini sağlar.

Yöntem yığını üzerindeki Foksiyon/metod yöntem yığınının seviyesini değiştirmek için seçilebilir. Yani seçili fonksiyon/metod için kaynak kod kaynak kod ekranında gösterilmektedir.

İzleme(Watch)

Kullanıcı tanımlı değişken kriterlerini gösterir. Sadece ismini girerek ya da değişkenler ekranından sürükle bırak yöntemi ile görülmek istenen değişkenler bu ekranda izlenebilir.

Duruşlar arasında değişkenler değiştiğinde değişkenler farklı bir renk alacaktır ki değişkenin değiştiği daha kolayca anlaşılmış olsun.

Değişkenler ekranında aynı zamanda değişkenin değerinin düzenlenmesi de mümkündür.

Çıktı(Output)

Hata ayıklama amacı ile X++ kod içerisinde yazılan metin(text) bilgiler bu ekranda izlenmektedir.

4 UYGULAMA METODOLOJİSİ

Axapta ile uygulama geliştirirken aşağıdaki dokuz adımı izlemeniz gerekmektedir :

1. Kavramsallaştırma : Problemi anlayın

2.Projenizi oluşturun

3. Genişletilmiş veri tiplerinizi ve temel dizilerinizi tanımlayın.

4. Tablolarınızı yaratın.

5. Sınıflarınızı yaratın ve iş mantığınızı yazın.

6. Kullanıcıyla etkileşim için form'larınızı tasarlayın.

7. Raporlarınızı yaratın.

8. Formlarınıza ve raporlarınıza erişim için menü nesneleri yaratın.

9. Sistem güvenliği ayarlarını yapın.

SONUÇ

Kurumsal Kaynak Planlama projelerinin başarılı olabilmeleri için en önemli husus kurumsal ihtiyaçların, beklentilerin doğru bir şekilde tespit edilmesi ve süreçlerle
ilgili tüm verilerin sağlıklı bir şekilde sisteme işlenmesidir.

Zamanla değişen rekabet unsurları, pazarda meydana elen değişmelere tepki hızını yükseltmeye ve bunları yaparken de maliyetlerini düşürmeye zorlamıştır.

Değişimlere hızlı ve ekonomik tepki verebilmek için, AX'ın kullanıcı dostu (user friendly) olan özelliği de bu aşamada devreye girmektedir. Açık kod sistemi ile uyarlama kolaylığı, daha etkin kodlama yöntemleri ile daha az kod sayısı kullanma, katman mimari ise geliştirilebilirlik ve hızlı uyarlama olanağı , obje tabanlılık özelliği ile hızlı uyumlaştırma süreçleri, standart tanımlamaların kodlama ihtiyacı duyulmadan kullanıcılar tarafından yapılabilmesi ve göze hitap eden detaylı geliştirmeci ara yüzleri ile ERP yazılım sektöründe daima lider firmalar arasında yer alacaktır.

ŞEKİL LİSTESİ

TABLO LİSTESİ

ÖZGEÇMİŞ

09.04.1982 tarihinde Ankara'da doğdu. İlk ve Ortaokulu Ankara' da Arı Koleji'nde tamamladı. 1996 yılında girdiği Marmara Kolejinden 2000 yılında mezun olup aynı yıl Maltepe Üniversitesi Bilgisayar Mühendisliği Bölümüne başladı. 2004 yılında okuduğu bölümden mezun oldu. 2004 yılında Bilgeadam Bilgi Akademileri kurumunda Öğretmen olarak işe başladı ve 2005 yılında işinden ayrılıp Maltepe Üniversitesi Yazılım ekibine katıldı, aynı sene yüksek lisans öğrenimine başlayıp 2007 yılında öğrenimini dondurup askerlik görevini yaptı. Askerlik görevini bitirdiği 2008 yılında Oyak Teknoloji Bilişim ve Kart Hizmetleri firmasında yazılım kadrosunda göreve başladı ve halen aynı yerde çalışmakta olan Alper ŞAMDANCIOĞLU, iyi derecede İngilizce bilmektedir.

Alper ŞAMDANCIOĞLU

KAYNAKÇA

[1] Yrd.Doç.Dr. Y.SAATÇIOĞLU, Kurumsal Kaynak Planlama, IZMIR: İLKEM YAYINCILIK,2007 s. 1-6

[2] http://www.capital.com.tr/haber.aspx?HBR_KOD=292

[3] Microsoft Business Solutions - Axapta Competitive ROI Positioning www.NucleusReserach.com Nucleus Research, Inc

[4]AMR Research, 2005

[5] http://www.sap.com/about/company/history/index.epx

[6] http://www.sap.com/turkey/about/index.epx

[7] http://www.oracle.com/timeline/index.html

[8] www.microsoft.com

[9] http://erp.technologyevaluation.com/

[10] http://www.microsoft.com/dynamics/casestudies/ax.aspx?casestudyid=4000003979

[11] http://www.hitachiconsulting.com/files/pdfRepository/PR_PACCESS_MicrosoftDynamics_FINAL.pdf

[12] www.bilgininadresi.com

[13] http://thaiaxapta.com.118.webhostforasp.net/Portals/0/stories/AX15.jpg