AIF: Kurumsal Uygulamalar İçin Bir Yazılım Çerçevesi

Murat Azgın, Serkan Avcı, Sema Söztutar, Burak Arık, Zekai Demirezen

Yapı ve Kredi Bankası, Bankacılık Üssü, 41480, Gebze, Kocaeli
mazgin@ykb.com, savci@ykb.com, ssoztut@ykb.com
barik@ykb.com, zdemirezen@ykb.com

Özet. Kurumsal firmalar, kendi ihtiyaçlarına cevap verecek birçok uygulama geliştirmektedir. Günümüzde bu uygulamalardan beklentiler gittikçe artmaktadır. Ayrıca bakım ve verimlilik açısından tekrar kullanılabilirlik önem kazanmaya devam etmektedir. Bu beklentilere, çevre ihtiyaçlarına ve gelişmelere hızlı cevap verecek esnek, modüler ve uluslararası standartlara uygun bir uygulama çerçevesine ihtiyaç duyulmaktadır. J2EE mimarisi üzerine kurulu bir uygulama mimarisi olan AIF (Application Independent Framework), bu ihtiyaç ve beklentilere cevap verebilmek amacı ile geliştirilmiştir. AIF, çok katmanlı mimariyi destekleyen, temel uygulama ihtiyaçlarının (Kanal / Uygulama / Kullanıcı / Oturum Yönetimi, Yetkilendirme, İzleme, vb..) yanı sıra kurumsal servisler de sunan (İş Akışı, Görev Yönetimi, SüreçYönetimi, vb..) bir uygulama çerçevesidir. Çerçevenin sunduğu bu hizmet ve servislerin bir kısmı çağrılabilir bir kütüphane olarak diğer kısmı ise şablon (soyut sınıflar) olarak sunulmaktadır.

1 Giriş

Yapı Kredi Teknoloji bünyesinde 2002 yılında küçük ve orta ölçekli uygulamalar için yapılan verimlilik ve etkin kaynak kullanımı çalışmaları doğrultusunda AIF[1]'in temelleri ortaya çıkmıştır. Özellikle YKT bünyesinde başlatılan "tekrar kullanılabilirlik oranının yükseltilmesi ve yazılım kalitesinin arttırılması" yönündeki çalışmalarla birlikte AIF son halini almıştır. AIF, RUP[2] [1] süreçleri baz alınarak, J2EE [2] mimarisi üzerine oturtulmuş, tamamen tasarım kalıplarıyla örülmüş, uluslararası kabul görmüş standartlar kullanılarak geliştirilmişitir. Bu makale kapsamında, AIF ve AIF ile birlikte sunulan özellik ve fonksiyonlar anlatılacaktır.

2 Kurumsal Yazılımların Özellikleri

Günümüzde, kurumsal uygulamalara baktığımızda hepsinde benzer fonksiyonel özellikler görmekteyiz. Bu özellikler, kurumsal yazılımlar için "olmazsa olmaz" durumuna gelmiştir. Bu özellikler aşağıdaki ana başlıklarda ele alınabilir:

2.1 Kullanıcı/Rol Yönetimi

Yazılımlar kullanıcılar tarafından kullanılsın diye yapıldığından, her kurumsal yazılımda en temel modül olarak kullanıcı yönetimi bulunur. Bu modülde kullanıcı tanımı yapılır, kullanıcı gruplara atanır, roller verilir ve organizasyon yapısı içerisindeki yeri tanımlanır. Böylece kullanıcı 360 derece belirlenir ve ilişkisi tanımlanır.

2.2 Kanal/Uygulama Yönetimi

Kurumsal firmalarda, aynı iş mantığına veya veriye değişik uygulamalar erişebilmektedir. Bu uygulamalar ise değişik kanalları kullanabilmektedir.

İşte bu erişimlerin daha iyi yönetilmesi ve izlenebilmesi için bir çok kurumsal uygulamada, kanal ve uygulama yönetimi yazılımları veya modülleri bulunmaktadır.

2.3 Güvenlik

Kurumsal uygulamalar için belki de en önemli fonksiyonalitelerden birisi de güvenlik konusudur. Güvenlik, kullanıcının sisteme şifre ile giriş yapmasıyla başlamaktadır. Birinci aşama olarak şifre doğrulama yapılmaktadır. Ardından bu kullanıcı için bir oturum yaratılıp, işlemlerinin bu oturum üzerinden yapılması sağlanır. İşlemler yapılırken kullanıcının yetkileri göz önüne alınır. Bu tür uygulamaların çoğunda verilerin bir kısmı şifrelenerek saklanmaktadır.

2.4 İş Akışı

Günümüzde kurumlar için en önemli konulardan birisi de iş akışı yönetimidir. İş akışı yönetiminin otomasyonu, gün geçtikçe önemini ve popüleritesini arttırmaktadır. Buradaki esas amaç, sistemde varolan iş akışlarının sisteme

[1] **AIF :** Application Independent Framework
[2] **RUP :** Rational Unified Process

tanımlanarak, işin durumunun sistem tarafından izlenebilir ve yönetilebilir olmasıdır. Bu akışlar koşullar değiştikce (örneğin kanuni zorunluklar nedeniyle) kolay bir şekilde değiştirilebiliyor olmalıdır.

2.5 İzleme

Günümüzde uygulamaların hem idari hem de teknik yönden izlenebilir olması oldukça önem kazanmaktadır. İzlenebilir bir sistem ancak gerçek anlamda denetlenip yönetilebilir. Bu nedenle bu tür uygulamalarda kullanıcılar, işlemler ve süreçler sürekli izlenmektedir. Teknik olarak ise uygulamanın kaynak kullanımını, hata oluşumunu ve sistem durumunu sürekli olarak izlemek gerekir.

Teknolojik olarak kurumsal firmalar yazılımlarında belirli özellikler aramaktadır. Bu özelliklerden bir kısmı aşağıda sıralanmıştır:

- Genişleyebilirlik
- Güvenilirlik
- Ölçeklenebilirlik
- Modüler Yapı
- Standartlara Uygunluk
- Çoklu Kanal Desteği

Bütün bu teknolojik özellikler ihtiyaçtan ortaya çıkmıştır. Uygulamalara yük bindikçe "Güvenilirlik" ve "Ölçeklenebilirlik" özelliklerinin devreye girmesi gerekmektedir. Uygulamaya yeni özellikler eklenirken ise "Genişleyebilirlik" ve "Modüler Yapı" özellliği kullanılır. Uygulamanın servis bazlı çalışabilmesi ise "Çoklu Kanal Desteği" kullanılır. Geliştirilen uygulamanın kalitesi ise "Standartlara Uygunluk" ile tescil edilir.

3 AIF Nedir ?

Herhangi bir yazılım projesine başlanmadan önce, yazılım grupları eğer tek bir proje geliştirmeyecekler ise bugün ve gelecekte yer alan projeleri için ortak bir disiplin belirlemek zorundadırlar. Bu disiplin süreçler, standartlar ve yazılım çerçevesini kapsar. Bunun yapılmaması halinde bir süre sonra birbirinden tamamen farklı, yönetilmesi zor, karmaşık projeler oluşur.

AIF kısaca, hazırlanacak herhangi bir yazılım projesinin ihtiyacı olan temel fonksiyonları bünyesinde barındıran, projelerin alternatif dağıtım kanallarına açık doğru bir mimari ile gelişmesini sağlayan, projelere belli disiplin öneren, sunucu ve istemci için nesne yönelimli "Uygulama Çerçevesi" barındıran mimaridir.[3]

AIF, kurumsal uygulamalar için temel oluşturmayı ve bu uygulamaların ihtiyacı olan tüm fonksiyonaliteyi bünyesinde toplamayı hedefler. Yani AIF, kurumsal firmalardaki yazılımların temel ihtiyaçları gözönünde bulundurularak geliştirilmiş bir çözüm mimarisidir. [4]

Şekil 1'de AIF'in genel yapısı gösterilmektedir. AIF hem "Sunucu" hem de "İstemci" tarafı için ayrı olarak uygulama çerçevesi sunmaktadır. Sunucu tarafı uygulama çerçevesi PIM[3] , İstemci tarafı uygulama çerçevesi PIC[4] olarak adlandırılmıştır.

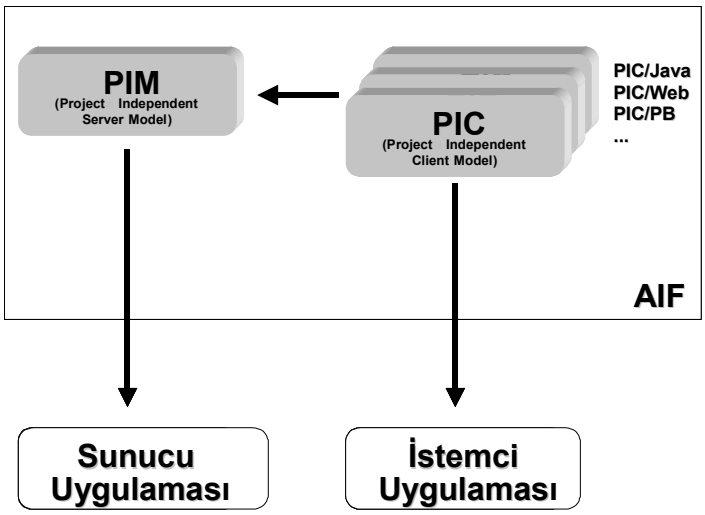
PIM, J2EE üzerine oturtulmuş bir mimaridir. J2EE' nin platform bağımsız olması, tekrar kullanılabilirliği desteklemesi, belirli bir standardizasyon sunması ve kurumsal özellikleri barındırması PIM mimarisinin J2EE teknolojisi üzerine kurulmasının en önemli sebeplerindendir.

İstemci tarafında ise farklı platformlar için birden fazla uygulama çerçevesi geliştirmek söz konusudur. İstemci tarafı değişik geliştirme ortamları olabileceğinden her ortam için bir uygulama çerçevesi oluşturmak gereklidir. Bu uygulama çerçeveleri genel olarak PIC olarak adlandırılmıştır.

AIF mimarisi kurumsal olarak bir takım hedeflere ulaşabilmek için tasarlanmıştır. Bu mimari ile erişilmeye çalışılan hedefler şu şekilde sıralanabilir :

- Çok katmanlı mimaride hazırlanacak projeler için sunucu ve istemci taraflı temel uygulama çerçevelerini oluşturmak,

[3] **PIM :** Project Independent Server Model
[4] **PIC :** Project Independent Client Model



Şekil 1. AIF Yapısı

- Uygulama geliştirme süreçlerinde verimliliği arttırmak,
- Farklı alternatif dağıtım kanalları ve veri kaynakları ile açık standartlar dahilinde entegre olabilmek,
- Projeler için ortak bir dil, disiplin ve standart sağlamak,
- Kaynakların projeler arasında etkin kullanımını sağlamak,
- Platform ve veri kaynağı tiplerinden bağımsızlık sağlamak,
- Hali hazırdaki projelere kolayca entegre olabilmek,
- Yeni teknolojileri desteklemek ve güncel takibini yapabilmek.

Hedefler doğrultusunda AIF mimarisi kullanacak projeler geliştirilirken, projeler normal yazılım süreçlerinden geçeceklerdir. Mimari mevcut yazılım süreçlerine alternatif önermemekle beraber tasarım, geliştirme ve bakım süreçlerinin verimliliğini arttırmaktadır. Diğer bir fayda da ortak disiplin ve standartlar kullanan projeler arasında etkin kaynak paylaşımının yapılabilecek olmasıdır.

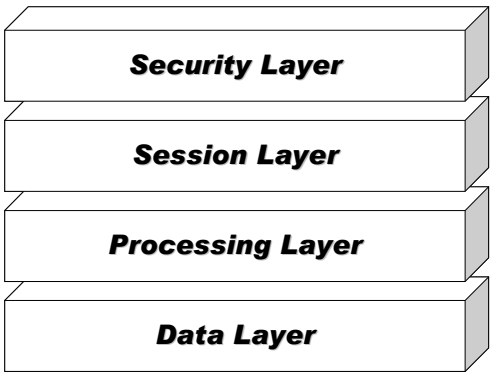
4 Mimari Özellikleri

AIF'in genel mimarisi şekilde gösterilmiştir. En üstte yer alan güvenlik katmanında, kullanıcı doğrulama, yetkilendirme, veri şifreleme ve veri kısıtlama gibi bir işlem gerçekleştirilmeden önce o işleme, kullanıcıya ve uygulamaya dair kontrol işlemleri gerçekleştirilir.

Oturum katmanı güvenlik katmanından geçildikten sonra oturumun yaratılması ve takibi işlemlerini içerir. AIF içerisinde, oturum bilgileri çoklu uygulama sunucusu kullanımını destekleyebilmek için veri tabanında tutulmaktadır.

İşlem katmanında gelen istekle ilgili olarak iş mantığının çalıştırılması sağlanır. Ancak işlem gerçekleştirilmeden önce o işlemle ilgili olarak çalıştırılacak metotların izlenebilir ve denetlenebilir olmasını sağlayan katmanlardan geçilir. Bu sayede bir işlemi kimin yaptığı, hangi parametrelerle yaptığı, bu işlemin gerçekleşmesinin ne kadar zaman aldığı gibi bilgilere erişilebilir.

En altta yer alan veri katmanında ise gerçekleştirilecek olan işlemle ilgili olarak, iş mantığının gerektirdiği veri saklamanın, veri getirmenin veya veri silmenin yapılması sağlanmaktadır. Bu katmanın varlığıyla işlem katmanı doğrudan veri kaynağına erişmediği için, işlem katmanı veri kaynağından bağımsız bir yapı kazanmaktadır.

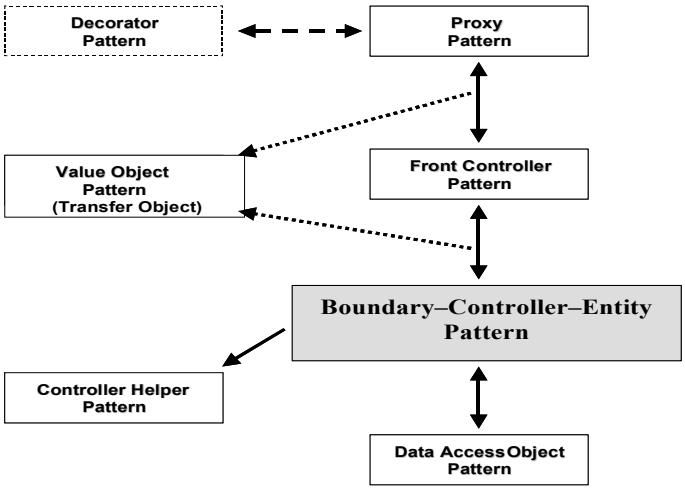


Şekil 2. AIF Katmanları

AIF'in çekirdek mimarisi ise Şekil 3'te gösterilmektedir.

Bu yapıya göre iskeleti "Boundary-Controller-Entity" [5] tasarım kalıbı oluşturmaktadır.

"Boundary" nesneleri diğer nesnelerle etkileşimi sağlamak için bir arabirim oluşturan nesnelerdir. AIF'in yapısında bu nesnelerin önünde izleme ve denetleme için bir merkezi yapı konmuştur. Bu yapı da bir "Front Controller" [6] tasarım kalıbındadır. Bu merkezi yapının ise önüne bir "Proxy" [7] tasarım kalıbı yapısı yerleştirilmiştir. Bu mekanizmanın amacı belirli bir platforma sistemin hizmet vermesidir. Bu nesneler birer EJB veya Servlet olabilmektedir. Bu tamamen kullanıma bağlıdır. İstemcilere gidecek verinin dönüşümünü sağlamak için ise "Proxy" nesnelerin önüne "Decorator" [7] tasarım kalıbında dekoratörler (Örneğin XML dönüşümü) konulabilir.



Şekil 3. Çekirdek Mimari

İskeletteki asıl iş mantığının çalıştığı nesneler "Controller" nesneleridir. Bunlar veriyi alıp iş mantığından geçirir ve "Boundary" nesnelerine iletirler. Genel olarak "Boundary" nesneleri birer "Façade" [7] tasarım kalıbı şeklinde çalışırlar. Bunlar birçok "Controller" nesnelerine erişerek istemcilere isteklerini iletirler. İş mantığı anlamında "Controller" lara yardımcı olan ve "Controller Helper" [6] kalıbında nesneler mevcuttur.

"Entity" nesneleri iş mantığından geçirilecek verileri tutmaktadırlar. Bunlar kısmen tek bir kaydı kısmen de bir kayıtlar dizisini içerirler. Bu nesnelere veriler ise "Data Access Object" [6] tasarım kalıbındaki bir sistem tarafından yüklenmektedir. Bu şekilde veri kaynağından bağımsız bir yapı da kurulmuş olmaktadır.

Tüm bu katmanlar arasında veri iletişimi ise "Value Object" [6] tasarım kalıbı kullanılarak oluşturulmuş nesneler ile sağlanmıştır.

5 Fonksiyonalite

AIF ile sunulan fonksiyonalite aşağıda özetlenmiştir.

5.1 Güvenlik Yönetimi

Kurumsal uygulamalar için güvenlik öncelikli ihtiyaçlardan olduğundan AIF'de kapsamlı bir güvenlik sistemi mevcuttur. Güvenlik kelimesinden kasıt:

- Katmanlar arasında taşınan verinin güvenliği
- Sunulan servislerin sadece yetkili kullanıcılar tarafından kullanımının sağlanması
- Verinin üzerinde işlem yapmak isteyen kullanıcının yetkilendirilmesi

olarak ifade edilmektedir. Güvenlikle ilgili sunulan fonksiyonaliteler şunlardır

- Login/Logout
- Veri Şifreleme
- Veri Kısıtlama
- Yetkilendirme
- Kullanıcı Tanımlı Yetkilendirme
- Çoklu Onay Mekanizması
- Güvenlik Politikaları

5.2 Kanal ve Uygulama Yönetimi

AIF çok katmanlı mimariyi ve çoklu kanal yapısını desteklediğinden bu kanalların ve bu kanallardan istekte bulunan uygulamaların yönetilmesi gerekmektedir.

Kanal ve uygulama yönetimi:

- Kullanıcı tanımlarının ve profillerinin merkezi olmasını
- Kullanıcı şifreleri ve kanallar arası şifre paylaşımını
- Uygulamaların aktif hale getirilmesini yada bloke edilmesini
- Uygulama çalışma aralığının kontrol edilmesini
- Versiyon kontrolunu
- Kanal fonksiyonlarının kontrolunu
- Merkezi raporlamanın yapılmasını sağlar.

5.3 İzleme ve Denetleme Yönetimi

AIF'de yapılan her iş izlenebilir ve denetlenebilir yapıya sahiptir. Burada sunulan fonksiyonalite şöyle sıralanabilir:

- Veri değişimlerinin takip edilmesi
  - Veri işlemlerinin saklanması
  - İstenen fonksiyonlar için aktivite log'u
  - Verinin yaratılması ve güncellenmesi
- Hata kayıtlarının tutulması (Error Logging)
- Kod takibi (Trace Logging)

5.4 İş akışı, Görev ve Süreç Yönetimi

AIF'in temel modüllerinden olan bu modülün sunduğu fonksiyonalitenin bir kısmı aşağıda sunulmuştur:

- İş akışlarının dinamik olarak oluşturulup değiştirilmesi
- Durum değişimlerinde tetikleme oluşturulması
- Hiyerarşik görev tanımlama yapılması
- Süreç tanımlama ve iş akışlarına atama

5.5 Kullanıcı Yönetimi

AIF üzerinde kullanıcı, kullanıcı grubu, rol, organizasyon tanımları ile bunlar arasındaki ilişkilerin belirlenmesi için

gerekli fonksiyonalite bulunmaktadır. Kullanıcılar ile ilgili, dil ve bölge tanımı gibi, kullanıcı profillerinin takip edilmesi bu bölüm sayesinde sağlanır.

5.6 Diğer Servisler

AIF son kullanıcılar için popüler bazı servisleri bünyesinde barındırır. Bu sayede uygulama programcıları geliştirecekleri projelerin daha kullanıcı dostu olmasını sağlamalarının yanı sıra süreç akışı sırasında bazı ek fonksiyonlar da elde ederler. Kullanıcı servisleri kısaca şu maddeleri içerir

- Ajanda (Organizer)
- İletişim Merkezi (Contact Center)
- Takvim (Calender)

6 Sonuç

AIF, mimarisi gereği platformdan, veri kaynağından ve sunuculardan bağımsız bir yazılım çerçevesidir. Kurumsal bir uygulamanın temel ihtiyaçlarının hepsini karşılayan AIF, aynı zamanda soyut yapıları ve arabirimleri ile programcılara nesneye dayalı programlama imkanlarını da en üst seviyede kullanılmasını sağlamaktadır.

AIF çalışmasının sonraki aşamalarında, farklı kaynaklara erişim imkanı verilecek, yeni kanallar eklenecek ve kurumsal fonksiyonalite ile servis sayısı arttırılacaktır. Bu şekilde kurum içerisinde öğrenilmesi, geliştirilmesi, uygulanması ve bakımı kolay bir çerçeve oturtularak verimlilik, etkin insan kaynağı kullanımı ve yazılım kalitesi arttırılmış olunacaktır.

Kaynakça

1. Grady Booch., Object Solutions : Managing The Object-Oriented Project, The Addison-Wesley Publishing , 1996
2. Nicolas Kassem, Designing Enterprise Applications with the Java 2 Platform, Sun Microsystems Inc., Ekim 2000
3. Bruce Eckel, Thinking in Pattern, Problem-Solving Techniques using Java, , MindView Inc.
4. Mohamed Fayad, Object-Oriented Application Frameworks, http://www.cs.wustl.edu/~schmidt/CACM-frameworks.html
5. Ivar Jacobson, Magnus Christerson, Patrik Jonsson, and Gunnar Overgaard, Object-Oriented Software Engineering- AUse Case Driven Approach, Addison- Wesley Publishing, Haziran 1992
6. Deepak Alur, Dan Malks, John Crupi, Core J2EE Patterns: Best Practices and Design Strategies, Sun Microsystems Inc., Haziran 2003
1. Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, Design Patterns, The Addison-Wesley Publishing, 1994