İlgiye Yönelik Yaklaşımla Yazılım Geliştirme
Software Development with Aspect Oriented Approach

Oytun Kurtar, Oya Kalıpsız,

Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği
Yıldız Teknik Üniversitesi
oytunkurtar@gmail.com kalipsiz@yildiz.edu.tr,

Özet

Nesneye Yönelik Programlama (NYP), yazılım mühendisliği için önemli bir dönüm noktasıdır. Çünkü NYP, günlük yaşantımızda karşılaştığımız problem çözme mantığına benzer olarak, problemleri nesne modeli olarak ele alabilmemizi sağlayan bir programlama mekanizması sunar. Fakat NYP teknikleri, yazılım sistemlerindeki artan ihtiyaçlar ve karmaşıklaşan problemler karşısında bazı gereksinimleri karşılamakta yetersiz kalmaya başlamıştır.
Bu çalışmada NYP tekniklerinin eksik kaldıkları yerlerde kullanılabilecek yeni teknikleri içeren bir programlama yaklaşımı üzerinde durulacaktır. Bu yeni yaklaşım İlgiye Yönelik Programlama (Aspect Oriented Programming, İYP) adı ile anılmaktadır. Çalışma süresince, bazı tasarımların neden gerçek kod içerisinde uygulanmasının zor olduğu ve karmaşıklığa neden olduğu incelenmiş ve bunu gidermek için kullanılan İlgiye Yönelik Programlama (İYP) teknikleri üzerinde durulmuştur.

Abstract

Object Oriented Programming (OOP) is a very important milestone in Software Engineering, since it provides a programming mechanism to handle problems as an object model , similar to our daily life problem solving mechanism. But in some cases, OOP techniques remain insufficient due to increasing needs and the complexity of the software systems.
In this work, insufficient parts of the OOP techniques will be discussed and the approach called Aspect Oriented Programming (AOP) will be introduced. During this work, reasons of difficulty of applying some design types in actual code and reasons of the results in complexity is examined, and focused on the AOP techniques to overcome these problems.

1. Giriş

İhtiyaçları sürekli artan ve karmaşıklaşan iş dünyasındaki birçok firma, bu gereksinimlerini yazılım ürünleri ile karşılamayı talep etmekte, süreçlerini bu uygulamalar ile takip ederek verimliliklerini arttırmayı hedeflemektedir. Kolay adapte edilebilir, düşük maliyetli ve değişiklik yönetimi kolay yapılabilen uygulamaların geliştirilmesine ihtiyaç duyulmaktadır.

Günümüzde geliştirilen uygulamaların büyük bir kısmı, özellikle kurumsal uygulamalar, tek bir dosya içerisine yazılmış tek bir modülden oluşmuş yapıda değildirler Bunlar, sistem gereksinimlerini karşılamak için oluşturulmuş modül koleksiyonlarının bir arada çalışmasından oluşmaktadırlar. NYP Tenikleri kullanılarak geliştirilen bu uygulamalarda program modülleri karmaşık hedefler içeren alt modüllere ayrıştırılmaktadır. Aynı hedefleri içeren kodların modüller içerisinde tekrarlanması ise kod karmaşasına yol açmakta ve uygulamaların yönetilebilirliğinin azalmasına yol açmaktadır.
İlgiye Yönelik Programlama'nın (İYP) hedefi bu ve benzeri yazılım geliştirme problemlerine çözüm getirmektir. Bu yeni programlama yaklaşımı, dağıtık işlevselliği merkezileştiren modüller olan ilgilerin oluşturulması üzerine kurulmuştur.

2. İYP Temel Kavramlar

İlgiye Yönelik Progranlama yaklaşımı ile Yazılım Dünyasına yeni girmiş terimler bulunmaktadır. Bunların bilinmesi bu yazılım geliştirme tekniğinin anlaşılmasına temel oluşturmaktadır.

- **İlgi:** İlgiye Yönelik Programlamanın temeli ilgi terimidir. İlgi, özel bir amaç veya kavram anlamına gelmektedir. Teknik terim olarak ifade edilirse ilgiler; yazılım geliştiriciler tarafından ele alınması gereken işlevsel veya işlevsel olmayan kalite hedefli sistem bileşenleridir.

- **Enine Kesen İlgi:** Birçok ilgi, sistemin temel fonksiyonelitesini etkileme eğilimindedir. Sistemin temel fonksiyonları ile bütünleşik çalışan ve bu temel fonksiyonların işleyişini etkileyen ilgilere Enine Kesen İlgiler (Crosscutting Concerns) adı verilir. Enine kesen ilgiler, sistemin diğer fonksiyonları ile birlikte çalışmak durumunda olduğundan bunların modüller içine yayılması, projenin ileri safhalarında kontrol edilebilirliğin zorlaşmasına hatta imkansız hale gelmesine neden olabilmektedir

- **Birleşme Noktaları (Join Points):** Bir sistemdeki Birleşme Noktaları ana modül içinde tanımlanan, bu modül içerisinde bir başka kodun (örneğin bir enine kesen ilginin) icra etmeye başlaması gereken yeri belirten iyi tanımlanmış yerler olmalıdır.

- **İcra Noktaları : (Pointcuts):** Birleşme Noktaları ile belirlenmiş özel yerlere gelindiğinde sisteme eklenecek

olan işlevin uygulamanın hangi kısmına ekleneceğinin bilgisinin barındırır. İcra noktaları 3 çeşittir:

- **Önce (Before):** Belirlenmiş metoddan önce çalışır.
- **Esnasında (Around):** Belirlenmiş metod ile birlikte çalışır. Koşul barınırır.
- **Sonra (After):** Belirlenmiş metoddan sonra çalışır

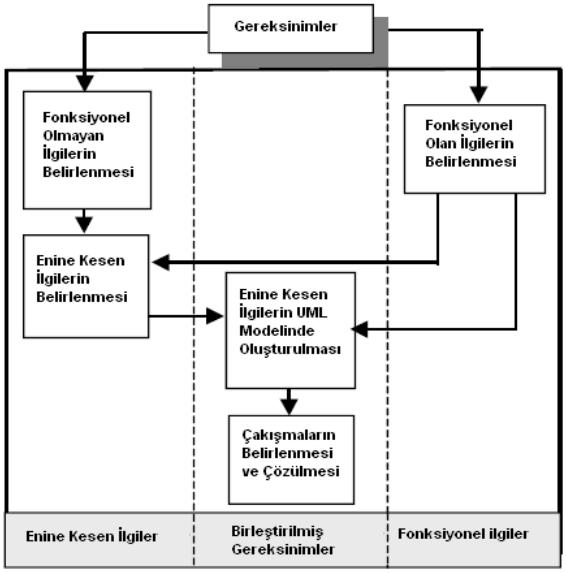
- **İcra Edilecek Kod (Advice):** Bir birleşme noktasına gelindiğinde, enine kesen ilgi kodu içerisinde tanımlanan icra noktasında, işletilecek olan kodu ifade etmek için kullanılır.
- **İYP İlgileri (Aspects):** İYP programlama metodolojisinde, ayrıştırılmış tüm enine kesen işlevlerin bulunduğu birimlere İYP ilgisi (aspect) adı verilmektedir.AspectJ ile geliştirildiğinde bu ilgilerin tamamı java sınıfları yapısındadırlar. Bu nedenle bu ilgilerden, sistemin ana sınıfları ile birlikte çalışan ilgi sınıfları olarak bahsedilecektir.
- **Ek Tip Tanımlamaları (Inter-Type Declerations):** İYP programlama yaklaşımı, mevcut sınıfların genişletilmesi, yeni tip ve metodların da eklenebilmesi için kullanılmaktadır. Ek Tip Tanımlamaları, İYP için önemli bir kavramdır çünkü sınıflara esnek tip tanımlama özelliği eklemektedir.
- **Dokuma (Weaving):** İYP'de, dokuyucu adı verilen mekanizma ayrıştırılmış halde bulunan ilgileri bir araya toplar. Diğer bir deyişle farklı çalışma mantığı olan ilgileri, çalışmaları gerektikleri noktaları belirleyerek birleştirir ve gereksinimleri karşılayan sistemin meydana gelmesini sağlar.

3. İYP Süreçlerinde UML

UML (Unified Modelling Language), yazılım mimarilerinin yaratılması ve dokümante edilmesi için kullanılan standart grafiksel bir dildir. Birçok metod ve teoriye kaynak teşkil edebilecek fikirler ve kavramlar içermektedir. UML, birçok diyagram tipi, modelleme elemanı, gösterim şekli içeren sayısız modelleme tekniği sunmaktadır. Bu teknikler farklı karakteristiklerdeki yazılım projelerinin modellenmesi için farklı yollar sunar. Bu bölümde İYP süreçlerinde UML dili kullanımı incelenecektir. İYP ile yazılım gereksinimlerinin ayrıştırılması Şekil 1'de görülmektedir.
Belirlenmiş enine kesen ilgilerin sistemde tariflenmesi Tablo1'de görülmektedir..

Tablo 1: Bir tablo örneği

Enine Kesen İlgi:	<Adı>
Tanım:	<Çalışma tanımlaması>
Öncelik	<Yüksek, Orta, Düşük>
Gereksinimler Listesi	<İlgiyi tanımlayan gereksinimler>
Model Listesi	<İlgi tarafından etkilenen UML modelleri>



Şekil 1: Gereksinimlerin Ayrıştırılması

4. İYP ve GOF Kalıpları

Yazılım mühendisliğinde tasarım kalıbı, sıkça karşılaşılan problemlere karşı geliştirilmiş genel uygulanabilen çözüm olarak tanımlanabilir. Bir tasarım kalıbı direkt olarak koda dönüştürülebilecek tamamlanmış bir tasarım değildir. Farklı durumlarda problemlerin çözümü için kullanılabilen bir tanımlama veya çözüm şablonudur.
Nesneye Yönelik Program geliştirmede sıkça kullanılan"Gang of Four" (GOF) tasarım kalıpları, sadece belirli bir dil veya kısmi bir programlama yaklaşımı için tasarlanmamışlardır. Tasarım kalıpları yazılım sistemlerinin yeniden kullanılabilirliğine katkıda bulunmak ve sıkça karşılaşılan problemlere çözüm getirmek amacıyla geliştirilmişlerdir. Belirli problemlere etkin çözümler getiren tasarım kalıpları bazen uygulamalar içerisinde enine kesen ilgiler problemine sebep olabilmektedirler. İYP'de tasarım kalıplarının uygulanmasının amacı, bu kalıpların uygulanmasındaki enine kesen ilgiler problemine çözüm getirmektir.

4.1. Tasarım Kalıplarının İlgiye Yönelik Programlama ile Uygulanması

Günümüzde tasarım kalıplarının İYP ile uygulanması alanında birçok araştırmalar yapılmaktadır. Tasarım kalıplarının İYP ile uygulanması ile elde edilecek birçok avantaj bulunmaktadır. Bunlardan bazıları aşağıdaki başlıklar şeklinde listelenebilir:

4.1.1. Yerellik

Yeni işlevsellik kazandıran kodlar ilgiler içerisinde tanımlanır, böylece uygulamaların kodlarında değişiklik yapılmasına gerek kalmaz ve modülerizasyon sağlanmış olur.

4.1.2. *Yeniden Kullanılabilirlik*

İşlevsellik kodunun ilgiler içerisinde uygulanması büyük oranda soyutlama sağlar. Bu özellik sayesinde tanımlanan arayüzler aracılığıyla yeniden kullanılabilirlik oranı artar.

4.1.3. *Birleştirme Şeffaflığı*

Uygulamanın bütününde karmaşıklığa sebep olmadan birçok tasarım kalıbının bir nesneye uygulanabilmesi sağlanır.

4.1.4. *Eklenip – Ayrılabilirlik*

Tasarım kalıpları kullanılarak geliştirilen bir uygulamada uygulanmış tasarım kalıbı uygulamanın genelini etkilemez. İlgiye Yönelik Programlama ile bir parametre değişimi tasarım kalıbının aktif veya pasif yapılabilmesini sağlar.

5. İYP Yaklaşımı ile Yazılım Geliştirme Uygulaması

Çalışma süresince anlatılan İYP yaklaşımının bir örnek üzerinde incelenmesinin, bu yaklaşım ile birlikte gelen avantajlar ve dezavantajların görülmesi açısından faydalı olacağı düşünülmüştür. Uygulamada öncelikle gereksinimler analiz edilmiş ve işlevsel gereksinimler ile enine kesen ilgiler ayrıştırılmıştır. Daha sonra uygulamanın nesneye dayalı diyagramları çizilmiş ve enine kesen ilgiler eklenerek aradaki farklar, İYP ile sistemin gerçeklenmesi incelenmiştir. Sistem gereksinimlerine göre uygun araç seçilmiş, ilgilerin içeriklerine ve sisteme etkilerine göre tasarım kalıpları ile gerçeklemeleri sağlanacaktır. Tasarım Kalıplarındaki enine kesen ilgiler problemi de göz önünde bulundurularak bu ilgiler için belirlenen tasarım kalıpları açıklanmış ve bu tasarım kalıplarının İYP ile uygulanmasıyla elde edilen sınıf diyagramları oluşturulmuştur. Belirlenmiş bu proje tasarımı ve planına göre de uygulama gerçekleştirilmiş ve sonuçlar incelenmiştir.
Örnek sistem bir Fatura Takip Uygulamasıdır. Gereksinimlerin analizi ile başlayan uygulama gerçekleştirim süreci, enine kesen ilgilerin tespiti, açıklanması ve ilgiye yönelik gerçekleştirme süreci ile ele alınarak tamamlanmıştır.

5.1. Gereksinimlerin Belirlenmesi

İlgiye Yönelik yaklaşım ile geliştirilecek olan Fatura Takip Sistemi (FTS) uygulamasının gereksinim analizi süreci sonunda gereksinimleri oluşturulmuştur. FTS, genel hatlarıyla satış sonucunda üretilecek olan fatura bilgilerinin onaya müteakip e-posta aracılığıyla ilgili kullanıcıların bilgilendirilmesini kapsayan bir uygulamadır.

5.2. Enine Kesen İlgilerin Belirlenmesi

Gereksinim Analizi sırasında listelenen sistemle ilgili tüm gereksinimlerin bir kısmı sistemden beklenen esas işlevler, diğer bir kısmı ise geliştirilecek olan tüm işlevleri etkileyecek olan ilgilerdir. Bölümün ilk kısmında belirtilen "Gereksinimler" göz önünde bulundurularak tasarlanmış ilgiler, çizelgelerde görülmektedir. Bu ilgiler, gereksinim analizi sırasında çıkarılmış olan gereksinimlerden, sistemden beklenen esas işlevselliği enine kesen ve esas işlevsellikle birlikte arka planda çalışması gereken fonksiyonelliktir. Bu kapsamda belirlenmiş olan 3 adet enine kesen ilgi bulunmaktadır.

5.2.1. *E-Posta Gönderme*

Fatura Giriş, Onaylama veya reddedilme gibi işlevlerle birlikte veya bu işlevlerden sonra çalışması gereken bir işlev olup, temel işlemleri enine kesen bir ilgi olarak belirlenmiştir. Tablo 2, bu ilginin UML diyagramını göstermektedir.

Tablo 2: E-Posta Gönderim İlgisi

Enine Kesen İlgi	Otomatik E-posta Gönderimi
Tanım	Sistem, fatura talebi oluşturma, onaylama, iptal, reddetme ve silme durumlarında tanımlı kullanıcı listesine e-posta göndermelidir.
Öncelik	Yüksek
Gereksinim Listesi	G1,G2,G3,G5,G6,G8
Model Listesi	Senaryolar: Fatura Talebi Oluşturma, Fatura Talebi Silme, Fatura Talebi Onaylama, Fatura Talebi Reddetme ve Fatura Talebi İptal Etme

5.2.2. *Veritabanı İşlemleri*

Sistem işlevleri olan kayıt, silme ve düzenleme gibi işlevlerin, sistemin iş mantığı ile birlikte çalışması gerekliliğinden dolayı kod karmaşasının önlenmesi amacıyla veritabanı işlemlerinin farklı bir katman tarafından yapılmasının sağlanması hedeflenmiştir. Veritabanı işlemleri ilgisinin UML diyagramı Tablo 3'de görülmektedir.

Tablo 3: Veritabanı İşlemleri İlgisi

Enine Kesen İlgi	Veritabanı İşlemleri
Tanım	Sistemin, tüm veritabanı kayıt, okuma ve silme işlerini esas modüllerden soyutlanmış bir ara katman tarafından gerçeklemesi
Öncelik	Yüksek
GereksinimListesi	G1, G2, G5, G6, G7, G8, G9, G10, G11, G12
Model Listesi	Senaryolar: Fatura Talebi Oluşturma, Fatura Tal. Silme, Fat.Tal Onaylama, Fatura Talebi Reddetme ve Fatura Talebi İptal Etme, Ödeme Girişleri, Fatura Taksit Eşleme

5.2.3. Günlük Tutma Sistemi

Sistem Fonksiyonalitesi ile birlikte çalışması gereken sistem günlüğünün tutulması da enine kesen ilgi olarak belirlenmiş ve Tablo 4'de görülen UML diyagramı oluşturulmuştur.

Tablo 3: Günlük Tutma İlgisi

Enine Kesen İlgi	Günlük Tutma
Tanım	Sistem, tüm işlemlerden sonra işlemi yapan kişi, işlem yapılan tarih ve yapılan işlem bilgilerini kaydetmelidir.
Öncelik	Yüksek
Gereksinim Listesi	G1, G2, G3, G4, G5, G6, G8, G9, G10, G11, G12
Model Listesi	Senaryolar: Fatura Talebi Oluşturma, Fatura Talebi Silme, Fatura Talebi Onaylama, Fatura Talebi Reddetme ve Fatura Talebi İptal Etme, Ödeme Girişleri, Fatura Taksit Eşleme, Taksit Güncelleme, Fatura Taksit Eşleme

5.3. Belirlenen İlgilerin GOF Kalıpları ile Tasarlanması

Sistem gereksinimlerinden ayrıştırılarak oluşturulmuş ilgilerin İlgiye Yönelik Programlama teknikleri ile gerçeklenmesinden önce bu ilgilerin nasıl tasarlanacağı konusu üzerinde durulması gerekmektedir. Çalışma süresinde İYP Tekniklerinin GOF Kalıpları üzerindeki etkilerinin ölçümlenmeye çalışılmasından dolayı ilgiler GOF Kalıplarından uygun olanlarının belirlenerek tasarım yapılmıştır.

5.3.1. Otomatik E-posta Gönderme İlgisi:

FTS uygulamasında fatura talep kayıtları ile ilgili tüm durum değişiklikleri hakkında tüm tanımlı kullanıcı listesine e-posta gönderilmesi talep edilmektedir. Standart Java uygulaması ile geliştirildiğinde tüm fatura işlemlerinin gerçekleştiği kodların altında e-posta gönderim ile ilgili kodların da yer alması gerekmektedir. Bu da kod karmaşasına sebep olabileceği gibi, e-posta gönderim kodlarıyla ilgili yapılacak bir değişikliğinde tüm bu kodların geçtiği yerlerde kod düzeltmesi yapmayı gerektirir. Bu nedenle e-posta gönderim işlemi, programın genelini etkileyen bir enine kesen ilgi olarak belirlenmiştir. E-posta gönderme ilgisinin tasarım kalıplarıyla gerçekleştirilmesi kod yayılımı ve dağıtımını engelleyecektir. Bunun için Gözlemci Tasarım Kalıbı uygun bulunmuştur. Gözlemci tasarım kalıbı, gözlenen nesnelerin durum değişikliklerinin algılanıp gerekli işlemlerin yapılmasını sağlamaktadır. Bu yapı temel alınarak fatura taleplerinin durum değişikliklerinin algılanarak otomatik e-posta gönderiminin tetiklenmesi, gözlemci tasarım kalıbı ile gerçeklenmiştir. Gözlemci Tasarım Kalıbının yapısı itibari ile uygulama içerisinde gözlemci ve gözlenen nesnelerin olması gerekmektedir. E-posta gönderim esnasında gözlenen nesne "Fatura" nesneleri gözlemleyen nesne ise MailGonderim sınıfı ise gözlemci konumundadır. Bu durumda Fatura nesnelerinde meydana gelen bir durum değişikliği, MailGonderim nesnesindeki e-posta oluşturma ve oluşturulan e-postayı gönderme metodlarını tetikleyecek ve enine kesen ilgi kodunun çalıştırılması sağlanacaktır.

5.3.2. Veritabanı İşlemleri İlgisi

Uygulamada faturaların ve ödemelerin sisteme kaydettirilmesi, kaydedilmiş faturaların listelenmesi, durumlarının değiştirilmesi, ödeme işlemlerinin yapılması, günlük kayıtlarının tutulması gibi veritabanından bilgi okuma, veritabanına bilgi yazma gibi işlemlerin gerçekleştirilmesi bir enine kesen ilgi olarak tasarlanmıştır. Bu enine kesen ilgi kodlarının uygulama içerisindeki kodlarla birlikte yazılıp kod yayılımına ve karmaşasına sebebiyet vermemesi için işlevsel sınıflardan soyutlanmaları gerekmektedir. Uygulamada, veritabanı bütünlüğü için eklenmiş olan Hibernate çatısı kod stili ile oluşturulmuş bir Data Access Object (DAO) ara sınıfı kullanılmaktadır. Bu ara sınıf uygulama işlevleri ile direk olarak bağlantılı olan OdemeYoneticisi ve FaturaYoneticisi sınıfları ile iletişim halindedir. DAO arasınıfı içerisinde verilerin kaydedilmesi, okunması ve çeşitli sql komutlarının kullanılan hibernate çatısına uygun yazılmış olan metodları mevcuttur. Bu ara sınıfla iletişim halinde bulunan FaturaYoneticisi ve OdemeYoneticisi sınıfları tüm uygulamanın veritabanı işlemleri ile ilişkili tek bağlantı noktaları olmalarından ötürü bu sınıfların yaratılmış tek bir kopyaları olması tüm işlemlerin bu kopyalar üzerinden gerçekleştirilmesi veritabanı bütünlüğü ve bellek kullanımı açısından gerek şarttır. Bir sınıfın tek bir kopyasının yaratılmasının istendiği durumlarda kullanılan Yegane Tasarım Kalıbı, Veritabanı İşlemleri ilgisi için kullanılabilir. Bu tasarım kalıbının İYP yaklaşımıyla uygulanmasıyla tüm veritabanı işlemi gerektiren sınıflar tarafından (eğer yaratılmamışsa) FaturaYoneticisi veya OdemeYoneticisi sınıflarının bir kopyasının yaratılabilmesi sağlanacaktır.

5.3.3. Günlük Tutma

Uygulamaya eklenecek olan günlük tutma işlevi sistem kullanıcıları tarafından yapılacak olan tüm işlemlerin kullanıcı ve tarih bazında kayıtlarının tutulmasını kapsamaktadır. Yapılacak olan işlemlerin sonunda günlüğünün tutulması, günlük tutma kodunun tüm metodlar içerisine gömülmesini gerektirir. Bu özelliğinden dolayı günlük tutma işlevi de sistemden bir enine kesen ilgi olarak tanımlanmıştır. Tüm işlem komutlarından sonra tanımlanması gereken bir ilgi olan günlük tutma, işletilecek olan uygulama komutlarının kaydedilmesi içerdiğinden dolayı tasarım kalıpları bölümünde anlatılan Komut Tasarım Kalıbı kullanılarak gerçeklenebilir. Komut Tasarım Kalıbı'nın İYP ile uygulanması ile işletilecek olan tüm komutların tek bir icra noktası ile gerçeklenmesi

sağlanabilir. Böylece uygulama, tasarım kalıbının enine kesen ilgi özelliğinden de yalıtılmış olur. Yaratılacak olan günlük tutma sınıfı, sistem tarafından gerçekleştirilen temel işlemlerin tanımlanmış oldukları FaturaYonetici ve OdemeYoneticisi sınıflarına uygulanacak ve bu sınıflar içerisindeki tüm metodlar ile birlikte çalışması sağlanacaktır.

6. Sonuçlar

Bu çalışmada, günümüzde sürekli artan yazılım ihtiyaçlarını karşılamak için NYP yaklaşımını temel alarak onun eksiklerini tamamlamayı hedef alan kullanım alanı genişlemekte olan İYP yazılım geliştirme yaklaşımı çeşitli yönleriyle incelenmiş, birçok problemin çözümünde kullanılan GOF tasarım kalıplarının bu yeni yazılım geliştirme yaklaşımı ile gerçekleştirimi üzerinde durulmuştur. İncelenen bu yeni yaklaşım, örnek Fatura Takip Sistemi üzerindeki enine kesen ilgilerin gerçekleştiriminde kullanılmıştır. Kullanılan yöntemin sisteme kattığı avantajlar ve dezavantajlar incelenmiştir.

Sistem ilgilerinin ayrıştırılması ile belirlenmiş olan Enine Kesen ilgiler, İYP ilgileri olarak tasarlanmışlardır. Belirlenmiş bu ilgiler GOF Kalıpları ve İYP tekniklerinin bir arada uygulanmasıyla gerçeklenmiştir. GOF Kalıplarına İYP tekniğinin uygulanmasıyla kalıpların gerçeklenmesi sırasında meydana gelen gereksiz kod tekrarlaması ve arayüz gerçeklemesinden kaynaklanan kod yeniden yazımı engellenmiştir. Tasarım kalıplarının gerçeklenmesi aşamasında her tasarım kalıbının kuralı soyut bir ilgide tanımlanmış, daha sonra tanımlanan gerçek bir ilgide bu soyut ilgide tanımlanmış olan kural mevcut ilgi verileriyle gerçeklenmiştir. İlgi kuralının soyut ilgi ile tanımlanması, sisteme daha sonradan eklenecek olan aynı kuralın kullanımını gerektiren farklı ilgiler olması durumunda tasarım kalıbının yeniden kullanılabilirliğini sağlamıştır.

Sistemin İYP tekniği kullanılarak uygulanması ile mevcut uygulama kodları değiştirilmeden yeni işlevlerin eklenmesi sağlanmıştır. Bu nedenle mevcut uygulama kodları farklı uygulamalarda da kullanılabilmektedir. İlgilerin sisteme eklenip ayrılması için mevcut ilgi kodunun eklenip çıkarılması yeterlidir. Uygulamanın kodlarında bir değişiklik yapılmasına gerek kalmamaktadır. Uygulamanın geliştirilmesi esnasında her yeni eklenen özellik ilgi sınıfları şeklinde eklendiğinen dolayı sistemin bakım yapılabilirliği de kolaylaşmıştır.

Çalışmanın gerçekleştirimi esnasında karşılaşılan en büyük güçlük tasarım kalıplarının İYP uygulamalarının pratikte çok az kullanılmış olmasıdır. Kurumsal uygulama geliştirme ortamlarında da İYP yaygın olarak kullanılmamakta ve daha çok akademik düzeyde incelemeleri ve araştırmaları yapılmaktadır. Bu nedenle kurumsal düzeydeki uygulamaların geliştirilmesinde kullanılabilmesi için pratik bilgi, zaman-maliyet hesaplamalarının ve derleme-çalışma zamanı performans planlamalarının proje özellikleri bazında değerlendirilerek yapılması gerekmektedir.

Belirli alanlardaki uygulamalara temel sağlayacak, genişletilebilir ve bakımı yapılabilir, kolay yönetilebilir ve zaman maliyet planlarına uygun bir yapının oluşturulması zor bir süreçtir. Bu çalışmada anlatılan GOF kalıpları ve İYP yaklaşımının kullanımı, yazılım geliştirme süreci sonunda istenilen özelliklere sahip uygulamalar geliştirilmesine yardımcı olacak, NYP yaklaşımı ile karşılaşılan birçok probleme çözüm olacaktır. Bu konulardaki akademik araştırmaların genişletilmesine destek olmak ve İYP kullanımının yaygınlaştırılması bu çalışmanın hedeflerinden biridir.

7. Kaynaklar

[1] Araújo, J., Moreira, A., Brito, I. ve Rashid A., (2002), "Aspect-Oriented Requirements with UML", *Workshop on Aspect-Oriented Modelling with UML*, October 2002, Dresden Germany.

[2] Bodkin, R. ve Laddad, R., (2004), "Zen and The Art Of Aspect Oriented Programming, Aspects can greatly simplify design and maintanence of complex systems" , *Linux Magazine*

[3] Charfi, A. ve Mezini, M., (2005), "Application of Aspect-Oriented Programming to Workflows The case of Web Service Composition with AO4BPEL", *Software Techonology Group,* March 2005, Sousse, Tunisia.

[4] Charfi, A. ve Mezini, M., (2006), "Aspect-Oriented Workflow Languages", *Talk @ coopIS* 2006, 3 Nov. 2006, Montpellier-France.

[5] Kande, M., Kienzle J. ve Strohmeier H., (2002a), "From AOP to UML- a bottom-up approach", Proceedings of the AOM with UML *workshop at AOSD*, 2002.

[6] Pawlak, R., Seinturier, L., ve Retaillé, J.P, (2005), Foundations of AOP for J2EE Development, Apress, USA