Bileşen Tümleştirmesine Dayalı Otomatik Uygulama Geliştirimi

Murat Mutlu ÖZTÜRK

Bilgisayar Mühendisliği Bölümü, Mühendislik Fakültesi,

Orta Doğu Teknik Üniversitesi, 06520, Çankaya, Ankara

e-posta: murato@havelsan.com.tr

Özet

Bu çalışma bir Bileşen Yönelimli Yazılım Mühendisliği Modelleme Aracı'nın JavaBeans bileşenleri ile tümleşerek olgunlaştırılması için yapılmıştır. Geliştirilen bir yeni arayüz aracılığı ile tasarım aşamasındaki bir sistemin yapı taşlarını oluşturan bileşen tanımlarına daha önceden geliştirilmiş ve jar biçiminde paketlenmiş JavaBeans bileşenleri atanabilmektedir. Bileşenler arasında kurulan bağlantılar ve bu bağlantıların bileşenler arasında üstlendikleri roller tasarım aşamasında tanımlanmaktadır. Böylece geliştirilmesi istenen sistemin herhangi bir ek yazılım yapmaksızın çalışabilir bir şekilde üretilmesi sağlanmıştır.

Anahtar Kelimeler: Bileşen, Bileşen Yönelimli Yazılım Mühendisliği, Bileşen Yönelimli Yazılım Mühendisliği Modelleme Aracı

Abstract

This study aims the improvement of the Component Oriented Software Engineering Modeling Media to enable integration of JavaBeans components. By means of this new interface, pre-developed reusable JavaBeans components packed in jar format can be assigned to the components that are the building blocks of a system under design. An executable system is generated by defining the connections between components and the roles that those connections undertake in between the components during the design.

Keywords: Component, Component Oriented Software Engineering, Component Oriented Software Engineering Modeling Media

1. Giriş

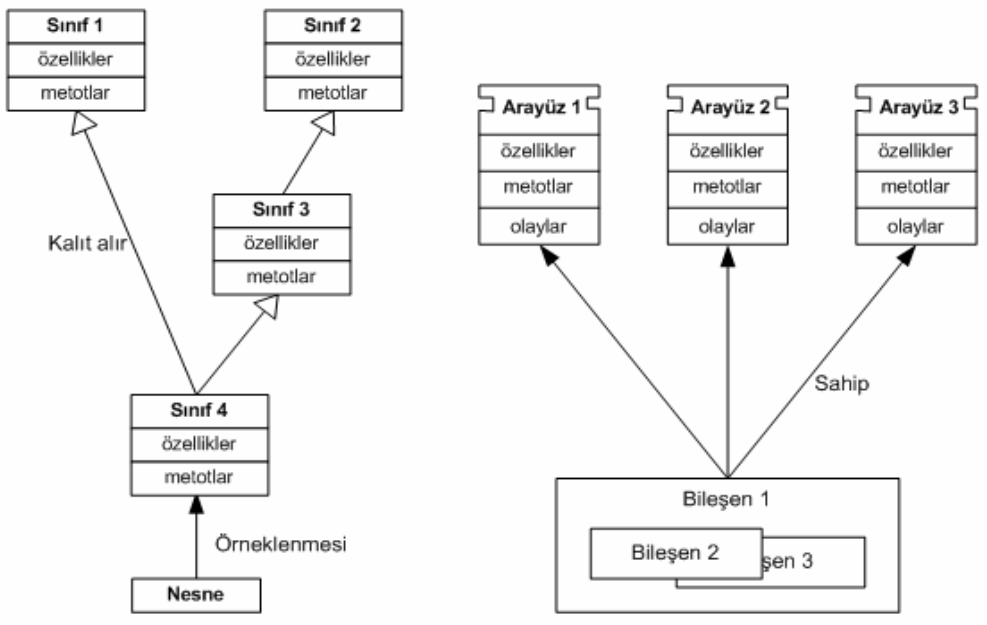
Bileşenler, birlikte çalışabilen ve diğer bileşenler ile arayüzleri belirlenmiş olan yazılım yapıtaşlarıdır [1, 2, 3]. Bileşenler varolan zorluklara rağmen, çalışma esnasında bir araya getirmek üzere oluşturulmuşlardır. Tamamen bileşen yönelimli bir yaklaşımda işlevlerin çoğunluğunun daha önceden geliştirilmiş olduğu varsayılırsa, herhangi bir yeni sistemin geliştirilmesi sadece bileşen bağlantılarının belirlenmesine indirgenir [4].

Bütünüyle bileşen yönelimli bir yazılım geliştirme süreci göz önünde bulundurulduğunda, tasarım yukarıdan aşağı bir yaklaşım ile sistemi bileşenlerine ayırmak ve daha sonra bu bileşenler arasındaki etkileşimleri belirleyerek tüm sistemi ortaya koymak yaklaşımı ile yapılır. Tasarım aşamasından sonra ise yapılan en önemli çalışma ortaya konan sistemin bütününü oluşturan

bileşenlerin bir araya getirilmesi aşamasıdır. Aslında sistem en küçük detayını dahi içerecek şekilde kendisini oluşturacak yapı taslarına ve bu yapı tasları arasındaki bağlantı esaslarının detayına kadar indirgenebilmiş ise, ve bu yapı taşları oluşturulan bileşen kütüphanelerinden elde edilebiliyor ise, tasarım aşaması ile tümleştirme aşamasını tek bir adımda düşünebiliriz.
Bileşen Yönelimli Yazılım Mühendisliği (BYYM) metodolojisi, kod geliştirmek yerine entegrasyona dayalı bileşen yönelimli sistem geliştirme paradigmasına uyar [2]. Bu konu üzerine yapılan diğer tez çalışmalarında, Bileşen Yönelimli Yazılım Mühendisliği Modelleme Dili (BYYMMD) için grafik tabanlı sürükle-bırak kabiliyetlerine sahip bir bilgisayar destekli yazılım mühendisliği aracı olan Bileşen Yönelimli Yazılım Mühendisliği Modelleme Aracı (BYYMMA) geliştirilmiştir [5]. Bu çalışmanın temel amacı, geliştirilmiş olan BYYMMA üzerine bu aracı kullanarak tasarlanan sistemin, herhangi bir ek yazılım yapmaksızın, sadece tasarımda kullanılan bileşenler arasındaki etkileşimleri tanımlamak suretiyle oluşturulmasıdır. Bileşen tümlemesi üzerine daha önce yapılan çalışmalar bulunmaktadır [6, 7, 8, 9, 10]. Yapılan bu çalışmalarda, tümleme için betik yada nesne yönelimli diller kullanılmıştır. Diğer çalışmalardan farklı olarak, bu çalışmada grafiksel bir modelleme aracı olan Bileşen Yönelimli Yazılım Mühendisliği Modelleme Aracı (BYYMMA), bir sistemi özel bir alana özgü bileşenşeri tümleyerek oluşturabilme kabiliyetine sahip olacak şekilde geliştirilmiştir. Bu tümleme, bileşenler arasındaki bağlantı noktalarının ve etkileşim detaylarının tanımlanması ile gerçekleştirilir. Bileşen tümlemesini bir tasarım aracı ile birleştirmesiyle bu çalışma bir örnek oluşturmaktadır.

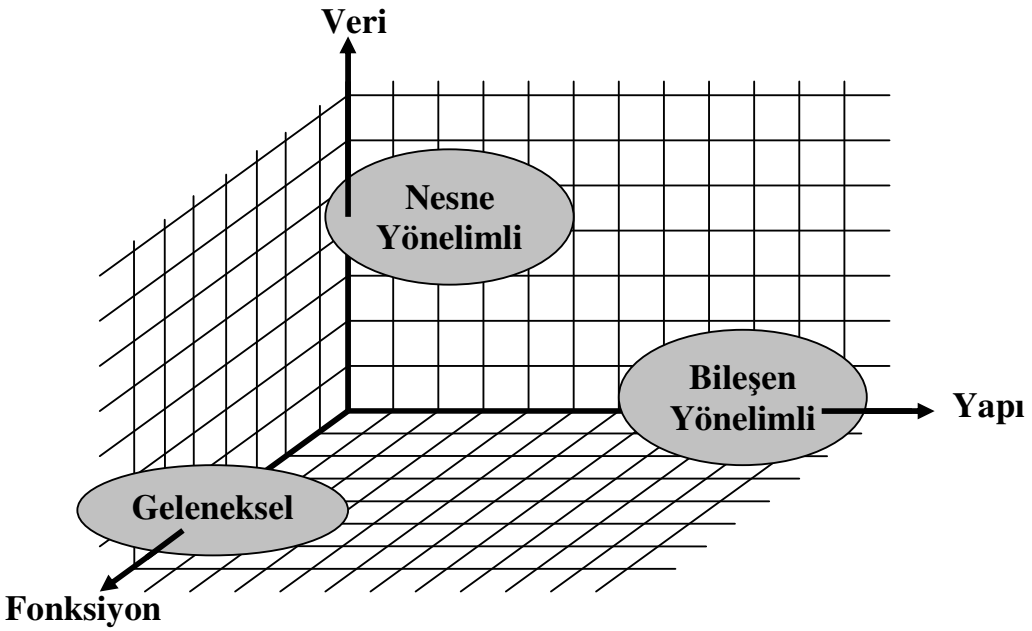
2. Bileşen Yönelimli Yazılım Mühendisliği Modelleme Dili ve Aracı

Temel olarak nesneler, özellikler ve metotlardan oluşmaktadır. Diğer taraftan, bileşenler nesnelere özgü özelliklerin yanı sıra arayüzlerinde sakladıkları 'olaylar'ıda içermektedirler. Bileşenler, Şekil 1'de görüldüğü gibi bileşim mekanizmasını arayüzlerini kullanarak yapıyor iken nesnelere ait olan sınıf hiyerarşisinde, özellik ve metotlar miras olarak aktarılmaktadır. [2,3]



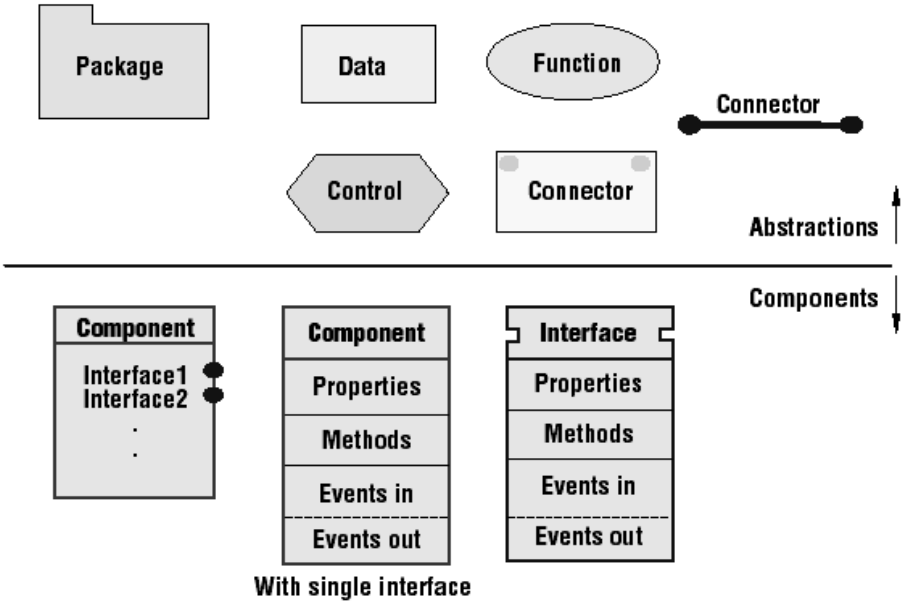
Şekil 1. Nesne ve Bileşen Karşılaştırması [2]

Modelleme paradigmaları, yazılım sistemlerine göre Şekil 2'de görüldüğü gibi üç boyutta değerlendirilebilirler: fonksiyon, veri ve yapı.



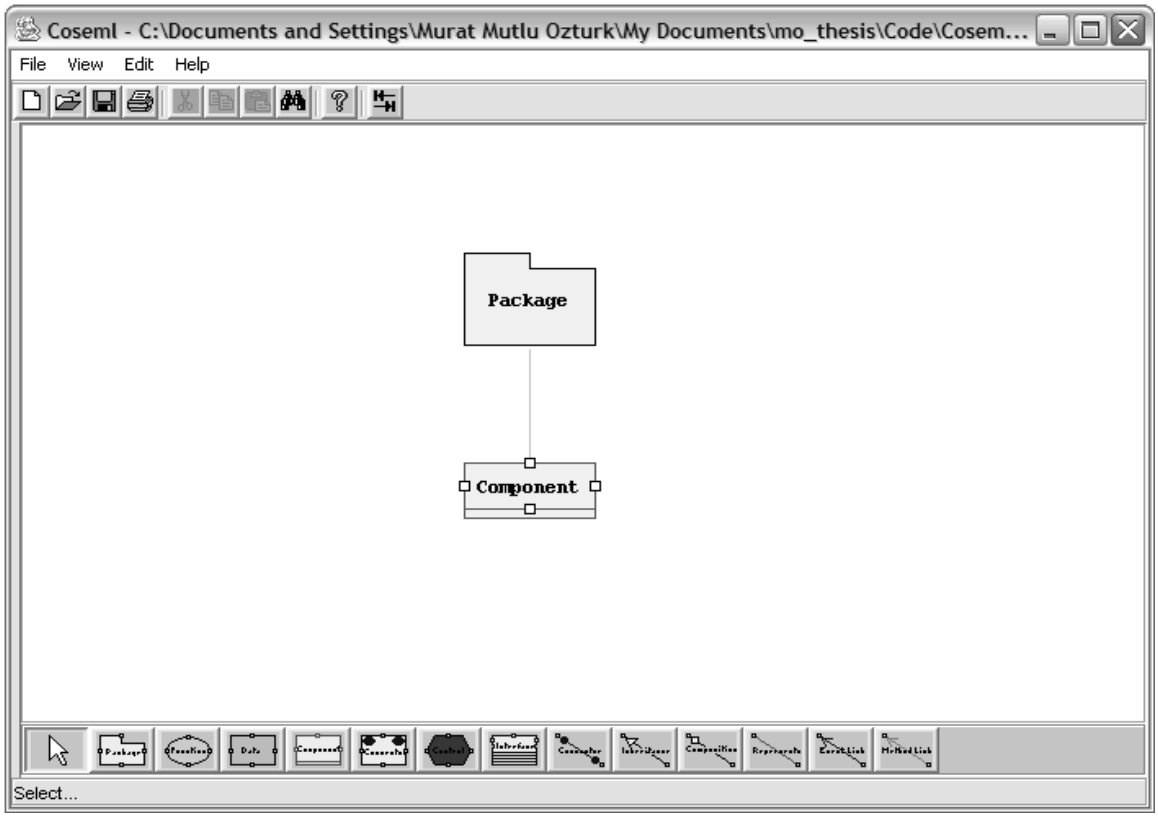
Şekil 2. Modelleme Paradigması Boyutları [2]

Buna göre, geleneksel yöntemler, sistemi fonksiyonel bir bakış açısına dayanarak, nesne yönelimli yöntemler veri soyutlamalarını kullanarak ve son olarak bileşen yönelimli yöntemler yapısal modelleme yaklaşımını adres göstererek tasarlarlar [2]. Bileşen yaklaşımlı bir dil olan BYYMMD, tasarlanan sistemi Şekil 3'te gösterilen ögeler ve bağlayıcılardan oluşan bir grubun yardımı ile yapısal olarak parçalara ayırır [2].



Şekil 3. BYYMMD'de Kullanılan Bileşen ve Bağlayıcı Semboller[3]

BYYMMD'ni bir sistem tasarım aracı ile desteklemek amaçlı olarak bir grafik modelleme aracı olan BYYMMA da geliştirilmiştir. Bu araca ait bir kullanım ile ilgili gösterge örneği Şekil 4'te gösterilmektedir.[2]



Şekil 4. BYYM Modelleme Aracı

3. BYYMMA ile JavaBeans Bileşenlerinin Tümleşmesi

BYYM temel olarak daha önceden oluşturulan yeniden kullanılabilir bileşen kütüphaneleri aracılığı ile sistemlerin tasarlanmasını ve geliştirilmesini öngörmektedir. Bu konu üzerine daha önce yapılmış çalışmalarda, BYYMMD'ne ait belirtimler [2] ve BYYMMD kullanarak sistem tasarımı yapmaya olanak sağlayan BYYM Modelleme Aracı sunulmuştur.

Bu çalışmada BYYM Modelleme Aracı'na tanıtılan bileşen çatısı, Java diline bileşen teknolojisini getirmiş bulunan JavaBeans bileşen çatısıdır. JavaBeans çatısının bileşen teknolojisi olarak seçilmesinin nedenleri:

- BYYM Modelleme Aracı'nın Java kullanılarak yazılmış olması ve bu nedenle geliştirilecek olan yeni ek yapının birleştirilme aşamasında herhangi bir sıkıntı ile karşılaşılmasından kaçınılması,
- JavaBeans Uygulama Programı Arabirimi (UPA) kullanılarak yeniden kullanılabilir platform bağımsız bileşenler geliştirilebilmesi,
- JavaBeans bileşen çatısı, herhangi bir uygulama derleyicisi kullanılarak kolaylıkla uygulamalara ve birleşik bileşenlere dönüştürülebilir olması.
- JavaBeans'lerin en önemli üç özniteliği; dışarıya sunduğu özellikler grubu, diğer bileşenler tarafından çağırılabilen metotlar grubu ve tetiklediği olaylar grubuna sahip olmasıdır. JavaBeans bileşen modeline ait olan bu temel öznitelikler BYYM yaklaşımı tarafından da desteklenmektedir.

Ayrıca tasarlanan sistemin otomatik olarak çalışabilir hale getirilmesi için Java programlama diline JavaBeans çatısını desteklemek amaçlı olarak eklenen *Reflection* (yansıtma) mekanizması da

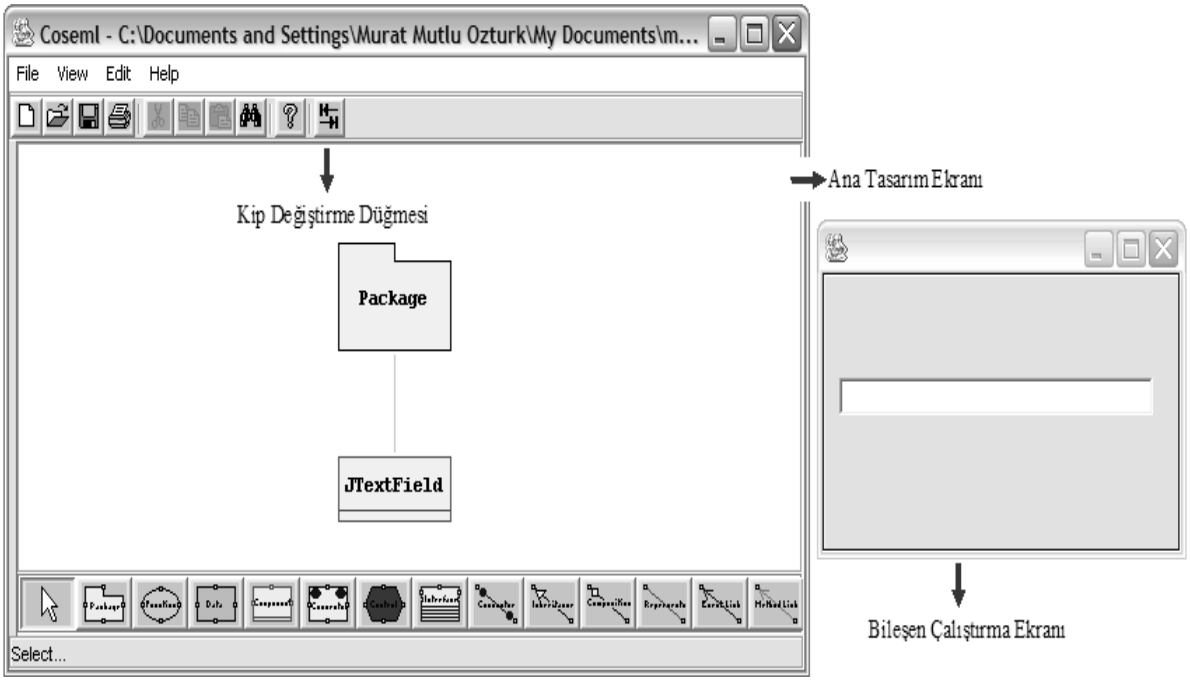
kullanılmıştır. Reflection mekanizması kullanılarak, çalışma esnasında herhangi bir bileşene ait özellikler ve metotlar da dahil olmak üzere bütün bilgilere ulaşılabilmektedir.

3.1. BYYMMA'na EklenenYenilikler

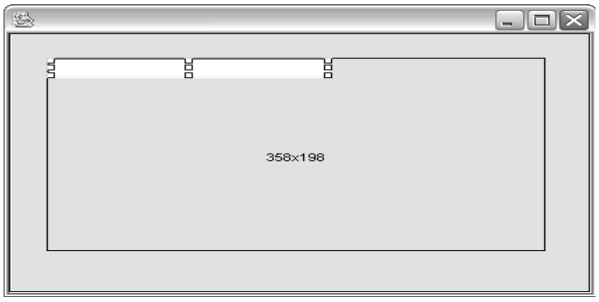
BYYM Modelleme Aracı'nın daha önceki çalışmalarla oluşturulan versiyonuna ilişkin bir örnek görüntü Şekil 4'te gösterilmiştir. Sistemin tasarımına olanak sağlayan ana ekranına ek olarak, bu çalışmada tasarlanan sistemin bileşenlerinin çalışır halde gösterildiği yeni tasarım ekranı eklenmiştir. Ana tasarım ekranı ve yeni eklenen bileşen çalıştırma ekranı Şekil 5'te gösterilmişlerdir.

Ana tasarım ekranı sistemin BYYMMD kullanılarak tasarlanmasına olanak sağlayan kısmıdır. Yapılan tasarım üzerindeki bileşenlere JavaBeans bileşenlerinin ilgili "jar" dosyasını seçmek suretiyle atanması durumunda ise, atanan bileşenin örneklemelerinin yapıldığı ve görüntülendiği ekran Şekil 5'te sağ tarafta görüntülenmiş olan bileşen çalıştırma ekranıdır.

Bileşen çalıştırma ekranının iki çeşit kipi bulunmaktadır. Tasarım kipinde oluşturulan bileşenlerin yerleri ve büyüklükleri ile oynanması sağlanmaktadır (Şekil 6) . Çalıştırma kipinde iken ise oluşturulan sisteme ait bileşenlerin çalışması sağlanmıştır. Bu kipler arasındaki geçişler, ana tasarım ekranı üzerinde bulunan ve Şekil 5'de "Kip Değiştirme Düğmesi" olarak belirtilen düğme aracılığıyla yapılır.



Şekil 5. Ana Tasarım Ekranı ve Yeni Bileşen Çalıştırma Ekranı

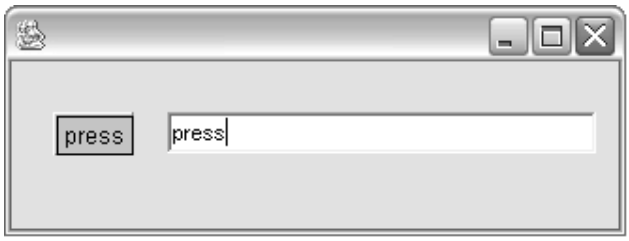


Şekil 6. Bileşen Çalıştırma Ekranı'nın Tasarım Kipindeki Görünümü

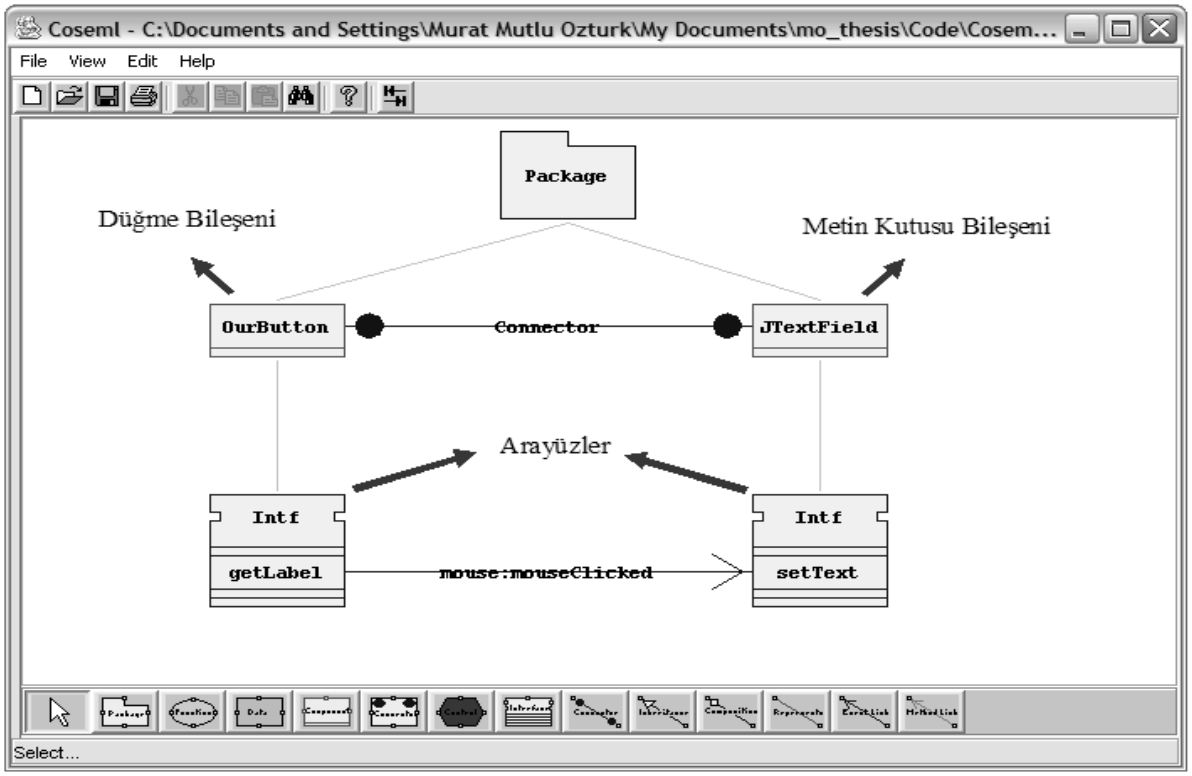
3.2. Çalıştırılabilir Sistem Tasarımı Süreci

BYYMMA'nı kullanarak herhangi bir ek yazılım geliştirmeksizin çalışabilir bir sistem geliştirme süreci için basit bir örnek sunulmaktadır., Bu süreç bir metin kutusu ve bir de düğme bileşeninden oluşan bir sistem tasarımını ele almaktadır. Tasarlanan sistemin çalışma prensibinin düğmeye basıldığında düğme yazısının alınıp metin kutusuna aktarılması olduğunu varsayalım. Buna göre sistemdeki akış, düğme bileşeninin fare ile tıklanması olayının gerçekleşmesi ile tetiklenir, düğme bileşeninden yazıyı alan bir alıcı fonksiyonun çağırılıp, onun çıktı değerinin metin kutusu bileşeni içerisindeki koyucu metoda gönderilmesi ile tamamlanır. Tasarım aşamasında öncelikle, sistemdeki iki bileşen için ana tasarım ekranında iki tane bileşen tanımlanır. Daha sonra bu soyut bileşenlere ana tasarım ekranı kullanılarak JavaBeans bileşen dosyaları atanır. Bu islem tamamlandığında ana tasarım ekranındaki soyut bileşene karşı düşen JavaBeans bileşeni çalıştırma ekranında görüntülenir. Atama işlemlerinin tamamlanmasından sonra sistem üzerindeki bileşenlerin etkileşimlerinin tanımlanması gerekmektedir. Sistemde iki bileşen arasındaki etkileşim, tasarım sırasında o bileşenler arasında kurulan 'bileşen bağlacı' ile tanımlanmaktadır. Bu bileşen bağlacı ile iki bileşen birbirine bağlandıktan sonra gerekli ayrıntılar belirlenmelidir. Bu ayrıntılar, ilişkili olay, bu olayın başlangıç bileşeninde tetikleyeceği metot ve bu metodun bitiş bileşeninde tetikleyeceği metot'dan oluşur. Bu ayrıntılar, ana tasarım ekranında açılan ve uygun olan tüm seçenekleri içeren yardımcı ileti pencereleri yardımıyla belirlenir. Olay ve metot atama işlemleri tamamlandıktan sonra iki bileşen arasında birbiri tarafından kullanılan arayüzler ana tasarım ekranına otomatik olarak eklenir. Sonuç olarak ortaya çıkan tasarım, Şekil 8'de ana tasarım ekranındaki görüntü olarak verilmektedir.

Tasarlanan sistemin çalışabilir sürümü ise Şekil 7'deki bileşen çalıştırma ekranında gösterilmektedir. Buna göre, sistemdeki düğme bileşenine basıldığı zaman, metin kutusu bileşenine düğme bileşeninin yazısı aktarılmaktadır.



Şekil 7. Bileşen Çalıştırma Ekranı : Örnek sistemin çalışır hali



Şekil 8. Ana Tasarım Ekranı : İki Bileşenli Basit Bir Sistem Tasarımı

4. Sonuç:

BYYM Modelleme Aracı'na Jar kütüphaneleri eklenmiş ve sistem geliştirme amaçlı olarak bu kütüphanelerdeki mevcut bileşenler kod yazmadan bağlanabilir ve çalıştırılabilir hale getirilmiştir. Sınırlı ölçüde gerçekleştirilen denemelerde istenilen kabiliyetin gerçekleştirildiği ve aracın koşturulabilir kod üretebilecek duruma getirilebileceği gözlenmiştir. Ancak, gerçek projelerde bu yaklaşımın denenerek ne derece etkin olabileceği henüz cevaplanmamış bir sorudur. Bu yaklaşım ile eldeki kısıtlı deneyimle de olsa verilecek bir sistem tanımından, ayrıştırma ve mevcut bileşenlerin tümleştirilmesi yolu ile kod oluşturulabileceği bizce bir ölçüde gösterilebilmiştir.

5. Gelecekte Yapılabilecek Çalışmalar:

Bu çalışmaya ek olarak yapılabilecek birçok çalışma bulunmaktadır. Öncelikle BYYMMA'na işbirliği ya da ardışıllık kavramları da eklenerek daha karmaşık sistem tasarımlarının başarılması sağlanabilir. Mevcut hali ile ortam dinamik modellemeyi desteklememektedir.

Bunun yanında, BYYMMA'na bir katman daha eklenerek, oluşturulmuş bulunan tasarım ve çalışma ekranları yardımı ile uygulama oluşturma amaçlı bir altyapı geliştirilebilir. Bu sayede BYYM'nin kullanılabilirliği arttırılmış olacaktır.

6. Referanslar

[1] Szyperski C., "Component Software: Beyond Object-Oriented Programming", Addison-Wesley, NewYork, 1998.

[2] Doğru H. A., "Component-Oriented Software Engineering Modeling Language: COSEML", Ankara, Aralık, 1998.

[3] Doğru H. A. ve Tanık M. M., "A Process Model for Component Oriented Software Engineering", IEEE Software, Vol.20, No. 2, s.34-41, Mart/Nisan 2003.

[4] Brown A. W. ve Wallnau K. C., "The Current State of CBSE", IEEE Software, Eylül-Ekim, 1998.

[5] Tanık M. M. ve Chen E. S., "Fundamentals of Computing for Software Engineers", Van Nostrand Reinhold, NewYork, 1991.

[6] P. H. Fröhlick, A. Gal, ve M. Franz, "Supporting Software Composition at the Programming Language Level", Science of Computer Programming 56, pp. 41-57, 2005

[7] O. Nierstrasz, S. Gibbs, ve D. Tsichritzis, "Component-Oriented Software Development", Communications of the ACM, Vol. 35, No. 9. Special Issue on Analysis and Modeling in Software Development, pp. 160-165, Eylül, 1992.

[8] O. Nierstrasz, D. Tsichritzis, V.D. Mey, ve M. Stadelmann, "Objects + Scripts = Applications", Proceedings, Esprit 1991 Conference, Kluwer, Dordrecht, pp. 534–552. 1991.

[9] V.D. Mey, "Visual Composition of Software Applications", Object-Oriented Software Composition, Prentice Hall, O. Nierstrasz and D. Tsichritzis (Ed.), pp. 275-303, 1995.

[10] L. F. Carpets, M. A. M. Carpetz, and D. Li, "Component-Based Software Development", IECON'01: The 27th Annual Conference of the IEEE Industrial Electronics Society, 2001