Uygulamada Yazılım Mimarisi Kararlarını Etkileyen Etmenler ve Kritik Fayda-Maliyet Öğeleri

Dr. Özlem ALBAYRAK
e-posta: ozlem.albayrak@gmail.com

Özet

Tanımı konusunda 1960 sonlarından günümüze kadar gelen tüm sorun ve uyumsuzluklara karşın, yazılım mimarisi alanında çeşitli konularda gerek üniversiteler ve araştırma kurumlarında çalışan araştırmacılar, ve gerekse endüstride uygulama geliştirenler tarafından çok sayıda çalışmalar yapılmıştır. Literatür çalışmaları, endüstride uygulama geliştirenler tarafından izlenilmekle birlikte, literatürde önerilen modeller uygulamaya değiştirilerek aktarılmaktadır.

Bu bildirinin amacı, yazılım mimarisi alanında yapılan literatür çalışmalarının ışığında, yazılım mimarisi ile ilgili olarak uygulamada yaşanmış örnek deneyimleri paylaşmaktır. Yazılım mimarisi kararlarını etkileyen etmenler ile yazılım mimarisi kritik başarı öğeleri ve maliyet unsurları konusunda, profili bildiride sunulan bir projede yaşanan deneyimler, teori ve uygulama arasındaki kopukluklar, kopuklukların nedenleri ile birlikte anlatılmaktadır.

Abstract

Despite all the problems and discrepancies on software architecture's definitions made since the end of 1960s, many studies have been conducted in the various fields of software architecture, both by the researchers working at the universities and the research establishments, and by the developers working in the industry. Literature studies are followed by the developers in the industry, yet the models suggested by the literature are transferred into practice with modifications.

The purpose of this paper is to share sample lived experiences from the industry, regarding the software architecture, under the light of the software architecture literature review. The factors effecting the software architecture related decisions, and critical success and cost factors, discrepancies between the theory and the practice, and their reasons are explained.

1. Giriş

Uygulamada yazılım projesi geliştirenler literatürde ve sektörde oluşan son gelişmeleri izlemek ve önerilen uygun yöntemler doğrultusunda mimari kararların alınmasıyla ilgili bilgi edinmek durumundadır. Aksi durumda rekabet güçlerinde azalma doğabilir. Yazılım sektöründe çalışanlar, bilimsel araştırmalar tarafından önerilen mimari yöntemler, çözüm önerileri ve araçları konuları ile ilgili olarak literatür çalışmaları hakkında ilgili ve bilgili olmalarına karşın, uygulamadaki yazılım projelerinde alınan mimari kararlar, izlenilen yöntemler ve hatta kullanılan araçların nitelikleri her zaman teoride geçerli olanlarla birebir örtüşmemektedir.

Bu çalışmada bir proje deneyimi kapsamında, yazılım mimarisini etkileyen kararlar, bu kararların planlaması ve uygulamaya geçirilmesinde karşılaşılan farklılıklar anlatılmaktadır. Bu kapsamda yazılım mimarisi kritik başarı ve maliyet öğeleri konusunda da literatür çalışmalarının uygulamadaki yeri ve anlamı karşılaştırılmaktadır.

Bildiride öncelikle literatürde yazılım mimarisi hakkında yapılan çalışmalara değinilmiş, bildiride kullanılan yazılım mimarisi tanımı verilmiştir. Yazılım mimarisi kararlarını etkileyen etmenler ile başarı, fayda-maliyet öğeleri sunulmuştur. Bildiride ayrıca örnek bir yazılım projesi profili sunulmakta ve yazılım mimarisi ile yazılım yöntemlerinin izlenmesi konusunda bu projede gerçekleşenlerle, literatüre göre farklı olan uygulamalar nedenleri ile birlikte vurgulanmaktadır.

Bildirinin son bölümünde, yapılan çalışmanın değerlendirilmesi, ve gelecekte yapılması olası çalışmalara dönük olarak geliştirici öneriler yer almaktadır.

2. Yazılım Mimarisi

Yazılım Mimarisi için evrensel kabul görmüş, standart bir tanım bulunmamaktadır. Bununla birlikte, hem üniversite ve araştırma kurumlarında çalışan bilimadamları, hem de endüstride çalışanlar tarafından yazılım mimarisi ile ilgili olarak çok sayıda ve çeşitli tanımlamalar yapılmıştır [1]. Bu tanımlamalardan seçilmiş iki örnek aşağıda verilmiştir:

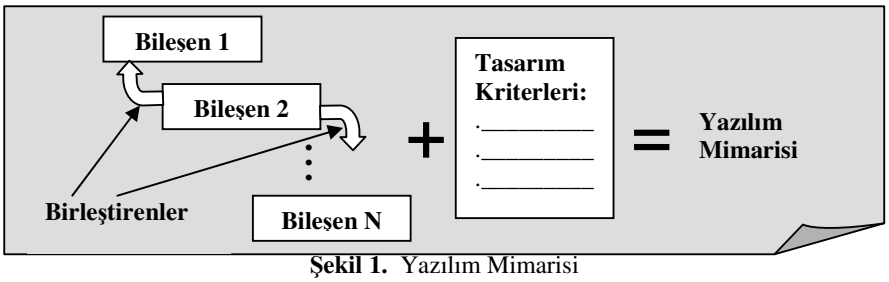
Bir program ya da bir sistemin yazılım mimarisi, yazılım öğeleri, yazılım öğelerinin dışarıdan gözlemlenebilir özellikleri ve yazılım öğeleri arasındaki ilişkilerden oluşan yapı ya da yapılardır [2].

Mimari, bir sistemin temel organizasyonu olarak tanımlanır. Bu organizasyon, sistem bileşenleri, bileşenlerin kendi arasında ve çevreleriyle ilişkileri ile tasarım ve gelişim kurallarını yöneten prensiplerden oluşur [3].

Yazılım mimarisine yönelik olarak yapılan yüzlerce tanım içerisinde, Garlan ve Shaw tarafından yapılan tanım, en fazla referans edilen tanım olmasına karşın, bu tanım da genel kabul görmemiştir [1, 4, 5]. Yazılım mimarisi konusu 1960'ların sonlarından itibaren konuşulan ve günümüzde çok yoğun olarak çalışılan bir konu olmasına karşın, neden hala genel kabul görmüş bir tanımının yapılamadığı hakkında, "Yazılım Mimarisini Tanımlamak Neden Bu Kadar Zor" başlıklı bir bildiri de yazılmıştır [4]. Bu bildirinin yazarları tarafından 2001 yılında yapılan başka bir çalışmada da yazılım mühendisliği alanında, yazılım mimarisi ile ilgili olarak teoride ve uygulamada oluşan problemlerin kaynağı olarak yazılım mimarisi ile geleneksel anlamda kullanılan mimari arasında analojiler kurulmuş olması gösterilmiştir [5].

Yazılım mimarisinin genel kabul görmüş bir tanımının yapılamamış olması, yazılım mimarisi ile ilgili kapsamlı çalışmaların yapılmasına engel olamamıştır. Bir yandan yazılım mimarisi tanımları ile ilgili çalışmalar sürerken, diğer yandan da, yazılım mimarisi türleri, yazılım mühendisi rolleri ile ilgili olarak çok sayıda araştırmalar yapılmıştır [5, 6, 7, 8, 9, 10, 11, 12].

Yazılım mimarileri, bir yazılım sisteminden diğerine farklılıklar göstermekle birlikte temelde tüm yazılım mimarilerinde ortak özellikler bulunur [9, 11, 12]. Genel olarak bir yazılım mimarisi; bileşenler, birleştiriciler ve tasarım kriterlerinden oluşan bir bütündür, Şekil 1.



Şekil 1. Yazılım Mimarisi

Son zamanlarda yazılım mimarisi konusu ile ilgili yapılan çalışmalar tanımlama çalışmalarından daha çok, yazılım mimarilerinin oluşturulması ve değerlendirilmesi, analiz ve kontrol edilmesi ile doğrulanması konularına yöneliktir [13, 14, 15, 16, 17, 18, 19, 20].

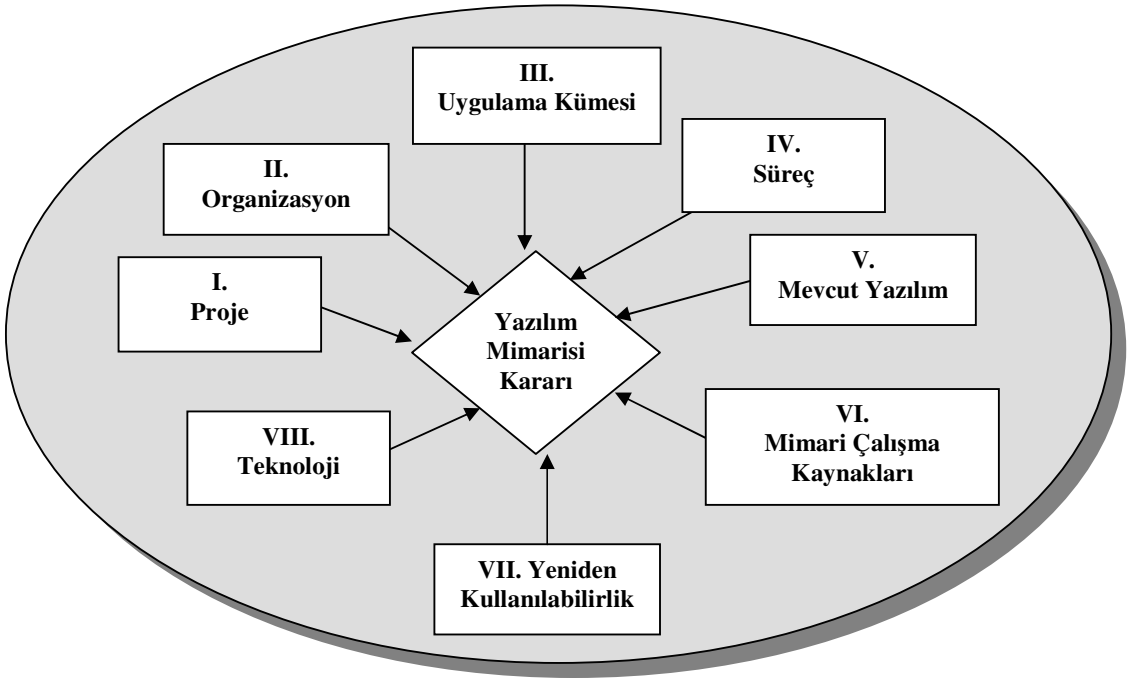
2.1. Yazılım Mimarisi Kararları ve Etkileyen Etmenler

Yazılım mimarisi ile ilgili olarak literatürde yapılan inceleme ve çalışmalar uygulamada yakından izlenmesine karşın, uygulamada alınan yaşamalar ve kararlar literatürde önerilenlerden farklılıklar göstermektedir.

Yazılım geliştirme yöntemlerinin en zayıf halkası birbirini izleyen süreçler arasında, önceki süreçlerde üretilmiş ürünler ile izleyen sürece ait ürünler arasındaki geçişin net olarak tanımlanmamasıdır. Literatürde erken tasarım kararlarını içeren yazılım mimarisi kararları bütün sistemin kalitesini etkileyebilecek derece önemli olduğundan, analiz aşaması ürünlerinden yazılım mimarisine geçişi belirleyen varsayımlar önerilmiş, yazılım mimarisi kararlarının analiz aşamasında oluşturulmasında kullanılan, yarı-otomatik bir süreç şeklinde belirlenebilecek güdüler ve varsayımlar incelenmiştir [13].

Uygulama yazılım mimarisi kararları verilirken yalnızca teoride geçerli varsayımlar izlenilseydi, benzer yazılım projeleri aynı mimari ve yöntemler izlenerek üretilirdi. 2004 yılında yapılan bir çalışmada, yazılım mimarisi üzerindeki "gerçek dünya" etkileri, endüstriden ilgililerle görüşülerek incelenmiş ve değerlendirmiştir [14].

Uygulamada yazılım mimarisi kararlarının; yazılım sistemi fonksiyonel ve uygulama kümesi özellikleri, yeniden kullanılabilirlik, mevcut yazılım özellikleri, teknoloji seçimi, organizasyon özellikleri, süreç özellikleri ve mimari etkinliklere katılan kaynakların özellikleri tarafından etkilendiği belirtilmiştir, Şekil 2.



Şekil 2. Yazılım Mimari Kararlarını Etkileyen Etmenler

Yazılım mimarisi kararlarını etkileyen etmenler kapsamında, her bir etmen ile ilgili birden fazla değişken incelenebilir. Uygulamada yazılım mimarisi kararlarını etkileyen değişkenlerin incelenmesine dönük olarak Türkiye'de yapılmış bir çalışma henüz bulunmamaktadır. Tablo 1, yazılım mimarisi kararlarını etkileyen etmenler ve değişkenler ile ilgili olarak kullanılabilecek öneri bir etmen-değişken seti sunmaktadır. Yeni önerilen etmen numaraları "Y" harfi ile başlamaktadır.

Tablo 1. Yazılım Mimari Kararlarını Etkileyen Etmenler ve Değişkenler

No	Etmen	Değişken(ler)
I	Proje	Büyüklüğü (adam ay), proje niteliği, personel değişim hızı
II	Organizasyon	Yaşı, sermaye yapısı, mühendis sayısı, üst yönetim özellikleri (teknik ya da değil), üst yönetim desteği, üst yönetimin kararlara katılımı, proje ekibinin dağıtık yerlerde bulunması
III	Uygulama Kümesi	Fonksiyonel gereksinimler, entegrasyon gerekleri
IV	Süreçler	Standartlar, sertifikalar
V	Mevcut Yazılımlar	Mevcut yazılımlar
VI	Mimari Çalışma Kaynakları	Yazılım mimarı sayısı, yazılım mimar(lar)ının deneyimi, mimari kararlara yazılım geliştirme ekibinin katılımı,
VII	Yeniden Kullanılabilirlik	Bileşen sayısı, bileşenlerin yeniden kullanım sayısı
VIII	Teknoloji Seçimi	Mimari dokümantasyon, tasarım ve değerlendirme
Y1	Kültür	Yeniliklere yaklaşım, hız
Y2	Müşteri	Profili (teknik, özel/kamu, baskın)
Y3	Pazar	Ürün ömrü, ürün ailesi, pazar rekabet düzeyi

2.2. Yazılım Mimarisi Kritik Başarı Unsurları ve Fayda-Maliyet Öğeleri

Yazılım mimarisi kararlarını etkileyen etmenlerin araştırıldığı bir çalışmada, kritik başarı unsurları ve fayda-maliyet öğelerinin de incelenmesi uygun olur. Yazılım mimarisi yatırımlarının potansiyeli mimari kaynaklı fayda-maliyet ilişkilerinin belirlenmesi ve hesaplanması ile belirlenir. Kritik başarı unsurlarının göz ardı edilmesi, ya da bunlarla ilgili maliyet öğeleri hakkında gerçekçi olmayan varsayımlar yapılması sakıncalı olur [21].

Tablo 2'de yer alan yazılım mimarileri kritik başarı unsurları ve ana fayda-maliyet öğeleri USC tarafından geliştirilmiş bir çerçevenin özetidir.

Tablo 2. Yazılım Mimarisi Kritik Başarı Unsurları ve Fayda-Maliyet Öğeleri

Kritik Başarı Unsurları	Ana Fayda-Maliyet Öğeleri
B1. Alan mühendisliği maliyeti	A1. Alan derinliği ve olgunluğu
B2. Mimari belirleme maliyeti	A2. Mimari olgunluk; miras ve COTS yazılım çeşitliliği
B3. Yeniden kullanılabilir bileşen maliyeti	A3. Yeniden kullanılabilirlik derecesi; diğer maliyet modelleri öğeleri
B4. Mevcut yazılım yeniden mühendisliği	A4. Mevcut yazılım yapısı, anlaşılırlığı

Kritik Başarı Unsurları	Ana Fayda-Maliyet Öğeleri
B5. Süreç yeniden tanımlama	A5. Süreç olgunluğu; süreç çeşitliliği
B6. Eğitim, ekip oluşturma	A6. Mimari yaklaşım uygunluğu; öğrenci sayısı
B7. Depo geliştirme, işlemleri	A7. Bileşen sayısı; kuruluş yerleri ve kullanıcılar
B8. Bileşen sertifikasyonu	A8. Bileşen sayısı; sertifikasyon derecesi
B9. Bileşen bakımı	A9. Uygulama durağanlığı; teknoloji, ortam
B10. Pazarlanan ürün maliyeti	A10. Ürün tipi (paket, servis); pazar büyüklüğü ve olgunluğu
B11. Maliyetten kaçınma faydaları	A11. Hattaki ürün sayısı; yeniden kullanılabilirlik derecesi
B12. Çevirim zamanı azaltma maliyeti	A12. Mimari çözümün tamlığı; personel deneyimi
B13. Kalite faydaları	A13. Mimari çözümün tamlığı; sertifikasyon seviyeleri
B14. Risk azaltma	A14. Mimari çözümün tamlığı; kullanılan süreç
B15. Pazarlanan ürün geliri	A15. Pazar büyüklüğü ve payı; fiyat yapısı

3. Örnek Proje Kapsamında Yazılım Mimarisi Kararları

Bu bölümde, 2. bölümde yer alan bilgiler doğrultusunda, örnek proje ayrıntıları sunulmakta ve değerlendirilmektedir. 2. bölümde belirtilen yazılım mimarisi ile ilgili değişkenlerin (etkileyen etmenler ile kritik başarı unsurları/fayda-maliyet öğelerinin) projedeki değerleri, Tablo 3 ve Tablo 4'de sunulmaktadır.

Tablo 3. ÖRNEK PROJE: Yazılım Mimarisi Etkileyen Etmenler ve Değişkenler

No	Etmen	Değişken(ler)
I	Proje	Büyüklüğü: 190 adam ay, süre: 2 yıl, Proje niteliği: AR-GE, Personel değişim hızı: işten ayrılan 5, yeni katılan 1 mühendis/ 2 yıl (başlangıç: 8 mühendis)
II	Organizasyon	Yaşı: 3 yıl, Sermaye: Çokuluslu, 70 bin USD, Mühendis sayısı: 8-3, Üst yönetim özellikleri: Teknik değil, Üst yönetim desteği: Tam, mali kısıtlamalarla sınırlı, Üst yönetimin kararlara katılımı: Onaylayan rolü, Proje ekibinin tümü aynı ofiste
III	Uygulama Kümesi	Fonksiyonel gereksinimler: Fonksiyonel gereksinimleri tanımlanma zorlukları, yazılım geliştirme ekibi için yeni bir alan, alan uzmanı bulma zorlukları, Entegrasyon gerekleri: Müşteri ERP, Coğrafi Bilgi Sistemi ve Veri tabanı sistemleri ile uyumlu, çeşitli raporlama araçları ile çalışabilecek, e-devlet projelerinden birden fazlası ile entegrasyon gerekleri
IV	Süreçler	Standartlar, sertifikalar: YOK
V	Mevcut Yazılımlar	Mevcut yazılımlar: YOK (firmada geliştirilmiş yazılım yok)
VI	Mimari Çalışma Kaynakları	Yazılım mimarı sayısı: 1, yazılım mimar(lar)ının deneyimi: Çok az, Mimari kararlara yazılım geliştirme ekibinin katılımı: İzleyici

No	Etmen	Değişken(ler)
		anlamında,
VII	Yeniden Kullanılabilirlik	Bileşen sayısı: ÇOK, Bileşenlerin yeniden kullanım sayısı: Sıfır
VIII	Teknoloji Seçimi	Mimari dokümantasyon, tasarım ve değerlendirme: Proje öneri dosyasında öngörülen teknoloji izlenildi. Araç seçiminde çoklukla açık kaynaklar seçildi.
Y1	Kültür	Yeniliklere yaklaşım: Yüksek, hız: Yüksek
Y2	Müşteri	Profili (Teknik değil, kamu, baskın)
Y3	Pazar	Ürün ömrü: Tahmin yok, Ürün ailesi: YOK, gelecekte ürün çeşitlemesi ile olması isteniyor, pazar rekabet düzeyi:

Tablo 4. ÖRNEK PROJE: Yazılım Mimarisi Kritik Başarı Unsurları ve Fayda-Maliyet Öğeleri

Kritik Başarı Unsurları	Ana Fayda-Maliyet Öğeleri
B16. Alan mühendisliği maliyeti: Yüksek	A16. Alan derinliği ve olgunluğu:Yüksek-Orta
B17. Mimari belirleme maliyeti: Düşük	A17. Mimari olgunluk; miras ve COTS yazılım çeşitliliği:Yüksek
B18. Yeniden kullanılabilir bileşen maliyeti: Orta	A18. Yeniden kullanılabilirlik derecesi; diğer maliyet modelleri öğeleri:Yüksek; Yüksek
B19. Mevcut yazılım yeniden mühendisliği: Düşük	A19. Mevcut yazılım yapısı, anlaşılırlığı:Mevcut yazılım yok
B20. Süreç yeniden tanımlama: Orta	A20. Süreç olgunluğu; süreç çeşitliliği:Düşük;Yüksek
B21. Eğitim, ekip oluşturma: Orta	A21. Mimari yaklaşım uygunluğu; öğrenci sayısı:Yüksek;Orta
B22. Depo geliştirme, işlemleri: Yüksek	A22. Bileşen sayısı; kuruluş yerleri ve kullanıcılar: Orta
B23. Bileşen sertifikasyonu:Yüksek	A23. Bileşen sayısı; sertifikasyon derecesi:Yüksek
B24. Bileşen bakımı: Yüksek	A24. Uygulama durağanlığı; teknoloji, ortam:Orta
B25. Pazarlanan ürün maliyeti:Düşük	A25. Ürün tipi (paket, servis); pazar büyüklüğü ve olgunluğu:Servis;Yüksek;Bilinmiyor
B26. Maliyetten kaçınma faydaları:Orta	A26. Hattaki ürün sayısı; yeniden kullanılabilirlik derecesi:1;Yüksek
B27. Çevirim zamanı azaltma maliyeti:Orta	A27. Mimari çözümün tamlığı; personel deneyimi:Yüksek;Uygun
B28. Kalite faydaları: Yüksek	A28. Mimari çözümün tamlığı; sertifikasyon seviyeleri:Yüksek;Yok
B29. Risk azaltma: Yüksek	A29. Mimari çözümün tamlığı; kullanılan süreç:Yüksek;Tanımlı
B30. Pazarlanan ürün geliri:Yüksek	A30. Pazar büyüklüğü ve payı; fiyat yapısı:Belirsiz, rekabet düşük, pay belirsiz

Yazılım mimarisini etkileyen değişkenlerle ilgili profili tanımlanan proje, TÜBİTAK-TİDEB destekli bir proje olup, mimari kararlarla ilgili planlar literatür ile uyumlu olacak şekilde proje öneri dosyası hazırlandığında oluşturulmuştur. Bununla birlikte, proje tanımının yapıldığı tarih ile, projenin tamamlandığı tarih arasında geçen sürede, uygulamada gerçekleştirilenlerle proje öneri dosyasında yer alan, literatüre dayanılarak planlanmış yazılım mimarisi ve yazılım geliştirme yöntem adımları arasında farklılıklar oluşmuştur.

Projede yazılım geliştirme yöntemi olarak planda önerilen Rational Unified Process (RUP) izlenmiş, tasarım sınıfları, sınıf diyagramlarından kaynak koda geçiş yerine, kaynak koddan geriye mühendislikle elde edilmiştir.

Projenin fonksiyonel özellik seti ile ilgili gelişmeler, bir pilot uygulama projesi ile aşılmaya çalışılmış, teknik özellik setinin çoğu karşılanmıştır. Çok katmanlı mimaride, esnek, ölçeklenebilir, farklı sistemlerle entegrasyonu gerçekleştirilecek (CBS bağımsız, ERP bağımsız, veritabanı bağımsız, raporlama aracı bağımsız) çok dille çalışabilen, mobil ve donanım bağımsız bir yazılım oluşturulması konusunda hedeflerin sayıca çoğuna ulaşılmıştır. Ancak uygulama ve plan arasında yazılım mimarisi alanında bazı sapmalar oluşmuştur. Bu sapmalar başarısızlık olarak ele alınmamalıdır. Koşullardaki değişiklikler nedeniyle, literatürün öngördüğü "olması gereken" ile "gerçekleşen optimum" çözüm özdeş olmayabilir. Örnek uygulamadaki kararların literatür önerilerinden kaymasının ana nedenleri;

- Proje ekibinin %50 oranında küçülmesi (kaynak yetersizliği),
- Eş zamanlı pilot proje nedeniyle yazılım teslim tarihlerinde değişiklik yapılması,
- Yazılım fonksiyonel özelliklerinde değişiklik yapılması olarak sayılabilir.

Projeyi planlanıldığı şekilde 9 kişi yerine, 4 kişi ile geliştirilmesi "I. Proje", "II. Organizasyon" ve "VI. Mimari Çalışma Kaynakları" ile ilgili etmenlere ait değişkenlerin değerlerini değiştirmiştir. Projede pilot uygulama olarak çalışılan alan bilgisi sağlayan müşteri, üretilmesi planlanan ürünün teslim zamanı ve niteliğinde değişiklikler yapmış baskın müşteri rolü ile "Y2. Müşteri" etmeni ile ilgili değişkenlerin değerlerini değiştirmiştir. Bu koşullarda fonksiyonel anlamda çalışan bir projeyi hızla üretmek adına, yazılım mimarisi alanında bazı çalışmalar atlanmış, yeniden kullanılabilir bileşenler iyi tasarlanmadan, geliştirme yapılmıştır. Esnek ve bağımsız olma noktalarında ödün verilmiş, bu nedenle ürünün tahmin edilen bakım maliyeti yükselmiştir.

Projede yazılım mimarisini etkileyen etmenlerle ilgili değişiklikler, planlananlar ile gerçekleşenler arasında farklılıklara neden olmuştur. Planlanan yazılım mimarisinde yer alan "bileşenler", "birleştirenler" ve "tasarım kriterleri" ile gerçekleşenler arasında sapmalar gerçekleşmiştir. En az değişiklik "bileşenler"de, en çok değişiklik "tasarım kriterleri"nde olmuştur. Yalnız yazılım mimarisi değil, kalitesi de olumsuz etkilenmiştir.

4. Sonuç ve Değerlendirme

Bu çalışmada bir proje kapsamında uygulamada yazılım mimarisi kararlarını etkileyen etmenler ve uygulamanın planlanandan/literatürden sapma nedenleri örneklenmiştir. "Proje", "Organizasyon", "Mimari Çalışma Kaynakları" ve "Müşteri" etmenleri ile ilgili belirtilen değişikliklerin bir projede yazılım mimarisi çalışmaları üstüne yansıması özetlenmiştir.

Yazılım mimarisi ile ilgili olarak yapılan literatür çalışmalarının ivme kazandığı bir dönemde, uygulamada yazılım geliştirenlerin yazılım mimarisi kararlarını etkileyen faktörler ilgi kazanmıştır.

Türkiye'de yazılım geliştiren özel kuruluşların yazılım mimarisi kararları verirken kullandıkları kriterler, bu kararları etkileyen teknik ve yönetsel etmenler ile yazılım mimarisiyle ilgili kritik başarı ve maliyet faktörleri profil çalışması henüz yapılmamıştır. Gelecekte önerilen çerçevede anketler hazırlanıp, yazılım mimarları ve mimari kararları onaylayanlarca hangi kriterlerin göz önüne alındığı incelenebilir. Projelerin büyüklükleri ve çeşitlerine göre bu etmenlerde farklılık olup olmadığı, firmanın özelliklerine göre bu kararlarla değişiklik olup olmadığı ya da yazılım mimarisi ile ilgili literatür ve son gelişmelerin uygulamada çalışanlarca izlenme oranları belirlenebilir. Yazılım mimarisi ile ilgili unsurlarla ilgili fayda-maliyet analizinin nasıl yapıldığı ile ilgili olarak bilgiler derlenerek, öncelikler araştırılabilir ve yeni çerçeve modelleri oluşturulabilir.

5. Teşekkür

Sayın Halit Oğuztüzün'ün (ODTÜ) güdüleyici desteği için derin teşekkürlerimi sunarım.

Kaynakça

[1]. Carnegie-Mellon Software Engineering Institute, "How Do You Define Software Architecture?", http://www.sei.cmu.edu/architecture/definitions.html, 1.Haziran.2005.

[2]. Bass, Klemen, Kazman, Software Architecture in Practice (2nd edition), Addison-Wesley, 2003.

[3]. Recommended Practice for Architectural Description of Software-Intensive Systems, ANSI/IEEE Std 1471-2000.

[4]. Baragy, J., Reed K., "Why Is It So Hard to Define Software Achitecture?", Proceedings IEEE Software Engineering Conference, s. 28-36, 3-4 Aralık 1998.

[5]. Barargy, J., Reed, K., "Why We Need A Different View of Software Architecture", IEEE Proceedings of the 23rd International Conference on Software Engineering, s. 121-134, 12-19 Mayıs 2001.

[6]. Shaw, M., "The Coming-of-Age of Software Architecture Research", IEEE Proceedings of the 23rd International Conference on Software Engineering, s. 656-664, 12-19 Mayıs 2001.

[7]. Barosi, L., Heekel R., Thöne S. ve Varro D., "Style-Based Refinements of Dynamic Software Architectures", Proceedings of the Fourth IEEE Working IEEE/IFIP Conference on Software Architecture, s. 155-164, 12-15 Haziran 2004.

[8]. "Role of The Software Architect", Worldwide Institute of Software Architects - WWISA, http://www.wwisa.org/wwisamain/role.htm, 30 Mayıs 2005.

[9]. Mattman, C. A., Crichton, D. J., Hughes, S. J., Kelly, S. C. ve Ramirez, P. M., "Software Architecture for Large-Scale, Distributed, Data-Intensive Systems", Proceedings of the Fourth Working IEEE/IFIP Conference Software Engineering Conference, s. 255-264, 12-15 Haziran 2004.

[10]. Özgürce, E., http://bilisim.milliyet.com.tr/detay.asp?id=1249, Yazılım Mimarisi konulu yazı, 16 Mayıs 2005.

[11]. Issamy, V., Tartanoglu, F., Jinshan, L., ve Sailhan, F., "Software Architecture for Mobile Distributed Computing", Proceedings of the Fourth IEEE/IFIP Conference on Software Architecture, s. 233-242, 12-15 Haziran 2004.

[12]. Drongowski, P. J., "Software Architecture in Realtime Systems", Proceedings of the IEEE Workshop, s. 198-203, 13-14 Mayıs 1993.

[13]. Perez-Marinez, J.E., Sierra-Alonso, A., "Heuristics for the Transition from Analysis to Software Architecture", Proceedings of the Fourth IEEE/IFIP Conference on Software Architecture, s. 311-314, 12-15 Haziran 2004.

[14]. Mustapic, G., Wall, A., Norström C. ve Crnkovic I., "Real World Influences on Software Architecture – Interviews with Industrial System Experts", Proceedings of the Fourth IEEE/IFIP Conference on Software Architecture, s. 101-111, 12-15 Haziran 2004.

[15]. Roshandel, R., Schmert, B., Medvidovic, N., Garlan, D. ve Zhang, D., "Understanding Tradeoffs Among Different Architectural Modeling Approaches", Proceedings of the Fourth IEEE/IFIP Conference on Software Architecture, s. 47-56, 12-15 Haziran 2004.

[16]. Tekinerdogan, B., "ASAAM: Aspectual Software Architecture Analysis Method", IEEE Proceedings of the Fourth IEEE/IFIP Conference on Software Architecture, s. 5-14, 12-15 Haziran 2004.

[17] Zhu, L., Babar, M. A. and Jeffery, R.,"Mining Patterns to Support Software Architecture Evaluation", Proceedings of the Fourth IEEE/IFIP Conference on Software Architecture, s. 25-34 , 12-15 Haziran 2004.

[18]. Barber, K. S., Holt, J., "Software Architecture Correctness", IEEE Software, Kasım/Aralık 2001.

[19]. Kong, J., Zhang, K., Dong, J. ve Song, G., "A Graph Grammar Approach to Software Architecture Verification and Transformation", IEEE Proceedings of the 27th Annual International Computer Software and Applications Conference (COMPSAC'03), 2003.

[20]. Licthner, K., Alencar, P., ve Cowan D., "A Framework for Software Architecture Verification", Proceedings of Software Engineering Conference, s. 149-157, 28-29 Nisan 2000.

[21]. Boehm, B., "Software Architectures: Critical Success Factors and Cost Drivers", IEEE Proceedings of the 16th International Conference on Software Engineering, s. 364,16-21 Mayıs 1994.