Savunma Sanayi Projelerinde Çevik Yazılım Geliştirme Yöntemlerinin Kullanımı

Orhan Aksoy[1,2], Kürşat İnce[1], Uğur Suyadal[1], Selçuk Karayakaylar[1]

[1] Deniz Savaş Yönetim Sistemi Teknolojileri Merkezi, HAVELSAN A.Ş., İstanbul, Türkiye
{oaksoy, kince, usuyadal, skarayakaylar}@havelsan.com.tr
[2] Bilgisayar Mühendisliği, Gebze Teknik Üniversitesi, Kocaeli, Türkiye

Özet: Savunma Sanayi projelerinde sözleşme gereği, yüklenici firmaların ağırlıklı olarak ön tasarım, kritik tasarım gibi aşamalarda sistem seviyesi tüm proje gereksinimlerini çıkarmış ve tasarımlarını tamamlamış olmaları gerekmektedir. Bu sebeple proje yöneticileri, yazılım geliştirme yöntemini genellikle çağlayan modeli olarak belirlemektedir. Ancak, yapılacak işlerin kritikliği ve yenilikçi yönleri kullanıcı gereksinimlerinin proje başlangıcında net bir şekilde tanımlanmasını, dolayısıyla geliştirme faaliyetinin planlanmasını zorlaştırmaktadır. Çevik yazılım geliştirme yöntemi, bu zorluğu, birbiri ile yoğun koordinasyon ile çalışan bir organizasyon kurarak azaltmaya çalışır. Bu bildiride, HAVELSAN Deniz Savaş Yönetim Sistemi Teknolojileri Merkezi bünyesinde yürütülen bir savunma sanayi projesinde, kullanıcı gereksinimlerinin belirsiz olduğu bir yazılım bileşeninde çevik yöntemin uyarlanması ve uygulanması sunulmaktadır.

Anahtar Kelimeler: CMMI-DEV, Yazılım Geliştirme Modelleri, Çevik Yöntem, Çevik Yönteme Geçiş, Savunma Sanayiinde Çevik Yöntem

1 Giriş

HAVELSAN Deniz Savaş Yönetimi Teknolojileri Merkezi'nde yürütülen Savunma Sanayi projeleri, genellikle kati fiyatlı (firm fixed price) sözleşmeler üzerinden yürütülmektedir. Bu sözleşmeler gereğince, yüklenici firmalar, proje takviminin ilk yılı içerisine yayılmış ön tasarım, kritik tasarım gibi aşamalarda sistem seviyesi tüm proje gereksinimlerini kayıt altına almış ve tasarımlarını tamamlamış olmaları gerekmektedir. Bu sebeple proje yöneticileri, yazılım geliştirme yöntemi olarak, geliştirme faaliyetlerini, analiz, tasarım, kodlama, test, sürüm, bakım gibi safhalara ayıran çağlayan modelini tercih etmektedir. Ancak sistem geliştirme projelerinin kırılgan ve önceden tahmin edilemeyen doğal sonucu olarak, firmaların daha proaktif ve dinamik, geliştirilen yazılımların da esnek ve hataya dayanıklı olmaları gerekmektedir [1]. Çağlayan modeli gibi geleneksel yazılım geliştirme yaklaşımları, evrimsel olmamaları ve değişme potansiyeli çok yüksek olan sistem geliştirme sürecine adapte olabilecek derecede esneklik sağlamamaları sebebiyle eleştirilmektedir [2], [3]. Bu sakıncaları azaltmak üzere çevik yöntemler önerilmiştir [4].

Çevik yaklaşım, çağlayan modelinde olduğu gibi geliştirme sürecini, birbiri takip eden safhalara bölmek yerine, gereksinimlerin değişkenliğini baştan kabul ederek müşteri ile birlikte sürekli çalışır durumda, kısa süreli yinelemelerle yeni özellikler kazanan bir ürün yaşatmak üzerine odaklanır. Ürün, proje geliştirme süreci boyunca sistem değişikliklerine ve müşteri taleplerine adapte olur ve gelişir.

Savunma sanayi projelerindeki plan odaklı süreçler, CMMI-Dev (Capability maturity Model Integration) süreç alanları ile yoğun olarak örtüşmektedir. Çevik yazılım geliştirme yöntemleri ve plan odaklı süreç iyileştirme modelleri arasındaki ilişkiler bir süredir yazılım dünyasının gündeminde yer almaktadır [6]. Örneğin, CMMI-Dev modeli üzerinde çevik yazılım geliştirme yöntemlerinin doğru uyarlamalarla kullanılabilmesi mümkündür [5]. Uyarlamaların başlıca gerekçesi, çevik yöntemler üzerine kurulu bir olgunluk modelinin kullanılmaması sebebiyle, çevik süreçlerin odağında proje başarısının değil, paydaşlar arasında işbirliğinin bulunmasıdır [7]. Çevik yöntemlerin doğrudan savunma sanayi projelerinde kullanımına yönelik çalışmalar da mevcuttur [8].

Savunma sistemlerinin yazılım bağımlılığı arttıkça, ABD Savunma Departmanı'nın (İng: Department of Defense, DoD), yönettiği ve alım yaptığı projelerde çevik yazılım geliştirme yöntemlerine ilgi de artmıştır [12]. Bunun arkasındaki başlıca gerekçeler, sistemlerin ve gereksinimlerin hızlı evrimleşmesi, yeni konseptlerin hızlı bir şekilde uygulanma ihtiyacı[13], projelerdeki Ar-Ge faaliyetlerinin bu sürece ayak uydurma gereği [14], dolayısıyla projelerdeki sürüm çevrimlerinin 18-36 aylık periyotlardan 30-60-90 günlük periyotlara düşürülmesidir [2007]. 2010 yılı itibariyle, DoD tarafından yapılacak tüm alımlarda, çevik yöntemin gerekleri kural olarak yayınlanmıştır [16].

Bu bildiride, HAVELSAN Deniz Savaş Yönetim Teknolojileri Merkezi (HAVELSAN DSYSTM) tarafından yürütülen bir savunma sanayi projesinde, seçilen bir yazılım bileşeni için çevik yöntemin uyarlanması ve uygulanması sunulmaktadır. Bu çalışmadaki yenilik, atıf yapılan çalışmalardaki çevik geliştirme yöntemi üzerine kurgulanmış projelerin aksine, çağlayan modeli gereklerini dikte eden bir proje sözleşmesi içerisinde çevik yöntemin uygulanmasıdır. HAVELSAN, CMMI-DEV v1.3 sertifikası gereğince, CMMI-DEV v1.3 süreç sahalarının tüm gereksinimlerini karşılamak yükümlülüğündedir. Bu modeldeki süreç sahaları, savunma sanayi sözleşme yükümlülüklerini karşılayacak şekilde firma süreçleri tarafından kapsanmaktadır. Bu sebeple, süreç uyarlamasında esas olarak CMMI-DEV v1.3 modeline uyumluluk alınmıştır. Uyarlanmış çevik süreç ile birlikte, bir yandan sözleşme aşamalarında ihtiyaç duyulan seviyede gereksinim ve tasarım bilgileri üretilirken, bir yandan çevik yöntemin yüksek verimli yazılım geliştirme mekanizması çalıştırılmıştır. Oluşturulan süreç, öncelikli olarak CMMI-DEV v1.3'in geliştirme ile ilgili süreç sahalarının ihtiyaçlarını karşılarken, bir yandan da proje yönetimini diğer süreç sahalarında desteklemiştir.

2 Çevik Sürecin Uyarlanması

Bu çalışmada uyarlanacak çevik süreçte *koşmaca*[1] *(İng.: Scrum)* modeli temel alınmıştır. Bu modelde üç rol bulunmaktadır. Ürün sahibi, süreç yöneticisi ve geliştirme takımı. Bu roller, kısa süreli geliştirme koşularıyla ürünler yaratarak, bir sonraki geliştirme koşusu için kararlarını bu ürünlerde ortaya çıkan sonuçlara göre şekillendirir. Koşular için tipik süre 2-4 haftadır. Ürün, kısıtlı bir gereksinim kümesiyle de olsa sürekli müşteriye teslim edilebilir durumda olmalıdır. Her koşunun başında yapılan koşu başlangıç toplantısında, o koşu sonunda ortaya çıkacak ürünün kapsamı, tamamlanacak iş ürünleri ve sorumluluklar belirlenir. Koşu süresince her gün yapılan kısa toplantılarda günlük faaliyetler ve karşılaşılan engeller gözden geçirilir. Koşu sonunda yapılan toplantıda ise iş ürünleri ile ilgili yapılan planlamalarla gerçekleşmeler karşılaştırılır, bir sonraki koşu planlaması için girdi sağlanır.

2.1 Çevik Manifestonun dört önemli değerine yaklaşım

Çevik manifesto, dört önemli değeri yöntemin merkezine yerleştirir:

- Bireylerin ve etkileşimlerinin, süreçler ve araçlardan daha değerli olduğu
- Çalışan bir yazılım ürününün, detaylı dokümantasyondan değerli olduğu,
- Müşteri ile işbirliğinin, sözleşme müzakerelerinden değerli olduğu,
- Değişime cevap vermenin bir planı takip etmekten değerli olduğu.

HAVELSAN'ın CMMI-DEV ve savunma sanayi sözleşmeleri gereği sorumlulukları sebebiyle bu dört önemli değer, belirli ölçülerde uygulanabilmiştir. Bireyler ve etkileşimler, koşu planlamalarında ve geliştirme faaliyetlerinde öne çıkarılmakla birlikte, planlama odaklı şirket süreçlerinin gerekleri de yerine getirilmiştir.
Aynı şekilde, her koşu sonunda çalışan bir yazılım sunulması odak noktasında tutulmakla birlikte, şirket süreçleri gereği, mühendislik dokümanları da çalışan yazılım bileşenleri ile birlikte üretilmiştir.
Sözleşme gereği proje kurgusu sebebiyle geliştirici ekibin müşteri ile doğrudan iletişimi mümkün olmadığı için müşteri ile işbirliği, proje kurgusunun izin verdiği ölçüde işbirliği toplantıları çerçevesinde gerçekleştirilebilmiştir.
Çevik yöntemin değişime kolay cevap veren evrimsel yapısı, sözleşme gereği projenin ilk safhalarında sistem seviyesi gereksinim kümesinin kesinleştirilmesi sebebiyle tam olarak uygulanamamış, ancak yazılım bileşeni gereksinim kümesinin belirlenmesinde kullanılabilmiştir.

2.2 Rollerle ilgili uyarlamalar

Ürün Sahibi Rolü: .

[1] Bu bildiride *Scrum* terimlerinin Türkçeleştirilerek kullanılmasında www.scrumturkey.com sitesinden faydalanılmıştır.

Sözleşme gereği proje kurgusu sebebiyle geliştirici ekibin müşteri ile doğrudan iletişimi olmadığı için ürün sahibi rolü, bir ekip üyesi tarafından üstlenilmiştir. Ürün sahibi, tüm süreç boyunca ekibi doğru yönlendirebilmek için, çevik yöntemden farklı olarak, tüm sözleşme gereksinimlerini karşılayacak kullanım senaryolarını hazırlamak üzere sistem mühendisliği faaliyetleri gerçekleştirmiştir. Bu yöntem ile sözleşme yükümlülükleri gereği ön tasarım ve kritik tasarım aşamalarında tamamlanması gereken sistem seviyesi gereksinim analizi ve tasarım faaliyetleri de eksiksiz olarak gerçekleştirilmiştir.

Süreç Yöneticisi Rolü: .

HAVELSAN'ın matris yapılı mikro organizasyon yapısı gereğince, geliştirme faaliyetleri, ilgili mühendislik alanlarına göre takımlarca gerçekleştirilmektedir. İlgili bileşenin gerçekleştirilmesi de tek bir takım sorumluluğundadır. Dolayısıyla, ilgili yazılım bileşeninin sözleşme ile kayıt altındaki proje takvimine uygun olarak tamamlanması, takım liderinin proje yöneticisine olan sorumluluğudur. Bu sebeple, çevik yöntemdeki süreç yöneticisi rolü, takım lideri tarafından üstlenilmiştir.

Çevik yöntem gereği,

Koşu planlamaları, tüm ekip tarafından anlaşarak yapılması gerekirken, bu projede süreç yöneticisi, proje takviminin koşu planlamalarına yansıtılmasını sağlamıştır. Öte yandan, ürün özelliklerinin önceliklendirilmesi, çevik yönteme uygun olarak ürün yöneticisi tarafından gerçekleştirilmiştir.

Takım içerisinde bir hiyerarşi olmaması gerekirken, sözleşme gereği, belirli gereksinimlerin takvim içerisinde tamamlanması gerektiğinden, takımın faaliyetleri takım lideri tarafından organize edilmiştir.

Ürün başarısı, yalnızca ürün sahibinin sorumluluğunda iken, bu projede, süreç yöneticisi ile paylaşılmıştır.

Geliştirme Takımı Rolü: .

Kaynak kısıtları sebebiyle ürün sahibi ve süreç yöneticisi de geliştirici rolü üstlenmiştir.

HAVELSAN'ın CMMI-DEV doğrulama ve geçerli kılma süreç alanları kapsamındaki sorumlulukları sebebiyle geliştirme takımına bir test mühendisi kalıcı olarak dâhil edilmiştir.

2.3 Uygulama ile ilgili uyarlamalar

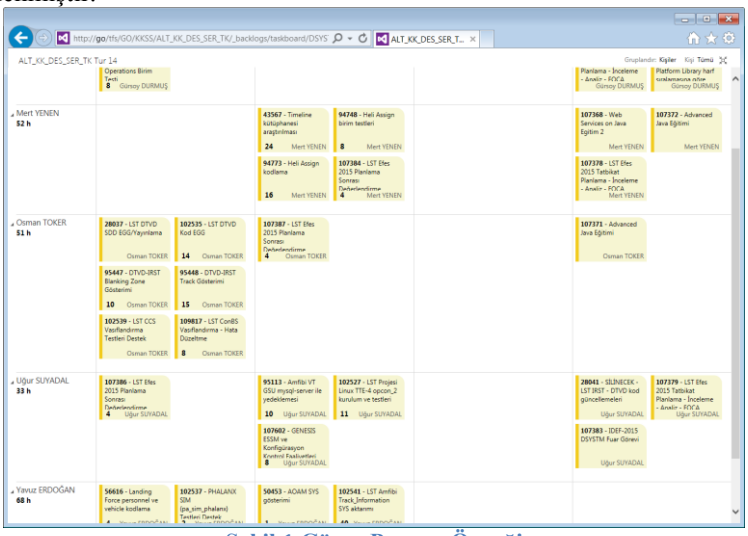
HAVELSAN'ın CMMI-DEV geliştirme, entegrasyon, doğrulama ve geçerli kılma süreç alanları ile ilgili sorumlulukları sebebiyle, mühendislik dokümanları da yazılım bileşenleri gibi ürün olarak değerlendirilmiştir.

Çevik yönteme göre iş öğelerinin belirlenmesi için, her koşu başlangıcında, ürün sahibi tarafından belirlenen kullanıcı hikayelerinden (İng.: User Story) oluşturulmaktadır. Nihai ürünün özellikleri, sözleşme gereği proje başlangıcında müşteri tarafından onaylanan sistem seviyesi gereksinim dokümanları ile kayıt altında olduğu için, kullanıcı hikâyeleri, bu gereksinimler referans alınarak oluşturulmuştur.

3 Çevik Sürecin Uygulanması

Çevik yöntem uygulanırken kurumsal bulutta yer alan HVL-GO (HAVELSAN Geliştirme Ortamı) uygulama yaşam döngüsü hizmetlerinden yararlanılmıştır. HVL-GO çekirdeğinde yer alan MS TFS (Microsoft Team Foundation Server) 2013 çevik yöntemi aşağıdaki şekilde desteklemiştir [11]:

a. Kullanıcı hikâyeleri ürün sahibi tarafından *User Story* iş öğesi kullanılarak kayıt altına alınmıştır
b. Yapılacak işlerin detaylandırılması için her kullanıcı hikâyesi ürün özelliklerine bölünmüş ve bunlar da *Backlog* iş öğesi kullanılarak kayıt altına alınmıştır. *Backlog* iş öğelerinden oluşan ürün özellikleri listesi proje takviminin izlenmesi için kullanılmıştır. İş öğeleri, taşıdıkları öncelik derecelerine göre sınıflandırılmıştır.
c. Sistem seviyesi tasarım *Configuration Item* iş öğesi olarak kayıt altına alınmıştır. Bunlar sistem seviyesi gereksinimlerle ilişkilendirilerek gereksinim ataması ve izlenebilirlik sağlanmıştır.
d. Her koşu ile birlikte ürün özelliklerinden öncelikli olan bir kısmının gerçekleştirilmesi koşmaca yönteminin en önemli prensiplerinden biridir. Proje takvimi göz önüne alınarak Takım Liderinin (Süreç Sahibi) tercihleri iş listesinin oluşturulmasında etkili olmuştur. Koşu açılış toplantılarında, seçilen ürün özellikleri görevlere detaylandırılarak *Task* iş öğesine dönüştürülmüştür.
e. Günlük toplantılarda görev panosu (Şekil 1) kullanılarak görevlerin durumu izlenmiştir.



Şekil 1 Görev Panosu Örneği

Günlük toplantılar eksiltmeli iş bitirme çizelgesi (İng.: Burndown chart) (Şekil 2) izlenmesi için kullanılmıştır.

Şekil 2 Eksiltmeli İş Bitirme Çizelgesi Örneği

f. Koşu kapanış toplantıları da görev panosu kullanılarak yapılmıştır.
g. Koşu süresinde ortaya çıkan yazılım hataları *Bug* iş öğesi kullanılarak kayıt altına alınmıştır. Kurumsal uyarlamada *Bug* iş öğesi *Task* iş öğesi ile eş düzeyde olduğundan bunlar da görev listelerine alınarak yönetilmiştir.

4 Koşu Verileri

Geliştirme ekibi çalışmalarına Mayıs 2014'te başlayıp makale yayın tarihine kadar 15 tur koşulmuştur. Toplamda bir yılı aşan bu çalışmada elde edilen koşu verileri Tablo 1'de gösterilmiştir.

Tablo 1 15 Tura Ait Koşu Verileri

	Tur Süresi (Gün)	Ürün Özelliği Sayısı	Görev Sayısı (adet)	Plan. (Toplam)	Grçklşn (Toplam)	Plan. (Ort.)	Grçklşn (Ort.)	Plandan Sapma (Ort.)
Tur 1	18	6	45	436	436	9,69	10,37	7,0%
Tur 2	21	6	33	488	583	14,79	18,20	23,1%
Tur 3	20	4	24	316	392	13,17	16,33	24,1%
Tur 4	15	3	27	416	595	15,41	22,04	43,1%
Tur 5	10	—	16	209	285	13,05	17,80	36,4%
Tur 6	12	10	50	460	508	9,20	10,81	17,5%
Tur 7	13	8	35	435	566	12,41	16,16	30,1%
Tur 8	20	10	53	712	734	13,43	14,39	7,1%
Tur 9	20	9	48	496	546	10,33	12,69	22,8%
Tur 10	21	9	40	467	670	11,68	16,75	43,5%
Tur 11	20	4	31	565	623	18,23	22,23	22,0%
Tur 12	20	9	44	626	729	14,23	17,35	21,9%
Tur 13	18	3	23	418	515	18,17	22,39	23,2%
Tur 14	18	5	28	468	620	16,71	23,85	42,7%
Tur 15	20	14	49	693	738	14,14	15,70	11,0%
Ort.	17,7	7,1	36,4	480,3	569,1	13,64	17,14	25,0%

Bu tabloda her tur için

Tur süresi: Planlanan iş günü,
Ürün özelliği: İşlem gören ürün özelliği sayısı,
Görev sayısı: Tamamlanan görev sayısı,
Planlanan Toplam: Görevlerin adam x saat tahminleri toplamları,
Gerçekleşen Toplam: Görevlerin adam x saat gerçekleşme toplamları,
Planlanan Ortalama: Görevlerin adam x saat tahminleri ortalamaları,
Gerçekleşen Ortalama: Görevlerin adam x saat gerçekleşme ortalamaları,
Plandan Sapma: Planlanan ortalamadan sapma değerleri göstermektedir.

Her gün için, 1 süreç yöneticisi 5 saat, 4 yazılım mühendisi 32 saat, 1 test mühendisi 3 saat, toplamda 40 saat kapasite kullanılmıştır.

5 Sonuçlar

Bu bildiride, HAVELSAN DSYSTM bünyesinde, sözleşme gereği projenin ilk aşamalarında sistem seviyesi gereksinimlerin analizinin ve tasarımının tamamlanmış olması gereken bir proje içerisinde, kullanıcı gereksinimlerinin belirsiz olduğu bir yazılım bileşeninin geliştirme faaliyetinde çevik yöntemin uyarlanması ve uygulanması sunulmuştur. Bu kapsamda, çevik yöntemdeki roller için yapılan uyarlamalar gerekçeleri ile açıklanmış, yöntemin uygulanması, kullanılan altyapıyı da içerecek şekilde detaylandırılmıştır.

Tablo 1'de yer alan tur verilerinin değerlendirmesi aşağıda sunulmuştur:

Tur süresi: Genel olarak 20 günlük tur süreleri planlanmıştır. Kurumsal takvime bağlı olarak tur sürelerinin 10, 12 ve 13 olarak gerçekleştiği üç kısa tur bulunmaktadır. Ortalama tur süresi 17,7 gün olarak hesaplanmıştır ki bu değer Koşmaca metodunun 2-4 hafta olan önerisine uygundur.

Turda işlem gören ürün özelliği sayısı: Bir turda ortalama 7,14 ürün özelliği (İng: backlog) planlanmıştır. Tur5'de Tur4'den kalan işler olduğundan yeni ürün özelliği açılmamıştır.

Turda kapatılan görev sayısı: Bir turda kapatılan ortalama görev sayısı 36,4 olarak hesaplanmıştır. Ortalama tur süresi göz önüne alındığında, bir günde kapatılan ortalama görev sayısının 2,1 olduğu görülmektedir.

Tur başına planlanan-gerçekleşen ortalama görev saatleri: Tur boyunca tamamlanan işlerin plan saatleri ile gerçekleşmelerinin ortalaması göz önüne alındığında, gerçekleşmelerin planlanandan ortalama %25 daha fazla olduğu görülmektedir. Genel olarak, ulaşılan detay seviyesinden dolayı, görev sayısı 30 ve üzerinde olan turlarda sapma oranı azalmaktadır. Geliştirme ekibinin saha çalışmaları yapması gereken turlarda, uzun süreli az sayıda görev planlandığı gözlenmekte ve gerçekleşmelerde sapma %43'e kadar artmaktadır. Bunun sebebinin, proje kurgusu gereği yapılan saha çalışmalarının belirsizlik barındırması olduğu değerlendirilmektedir.

Sonuç olarak, günümüzdeki hızlı değişen ortam şartlarında en uygun yazılım geliştirme yaklaşımının çevik yöntem olmasından yola çıkılarak [10], sözleşme gereklerinin çağlayan modeli işaret ettiği projelerde dahi yazılım geliştirme faaliyetlerinde bu yöntemin uyarlanarak uygulanabildiği deneyimlenmiştir.

Kaynaklar

1. Livari, J., Livari, N.:The relationship between Organizational Culture and the deployment of Agile Methods, Information and Software Technology, 53, 509-520 (2011)
2. McMahon, P.E.: Bridging Agile and Traditional Development Methods: A Project Management Perspective, The Journal of Defense Software Engineering, 16-20 (2004)
3. Nerur, S. Mahapatra, R., Mangalaraj, G.: Challenges of Migrating to Agile Methodologies, Communication of the ACM, 48(5), 73-78 (2005)
4. Tanner, M., Willingh, U.: Factors leading to the success and failure of agile projects implemented in traditionally waterfall environments (2014)
5. Top, Ö. Ö., Demirörs, O.: CMMI ve Çevik Yazılım Geliştirme Yöntemlerinin Birlikte Uygulanabilirliği, 7. Ulusal Yazılım Mühendisliği Sempozyumu (2013)
6. Boehm, B., Turner, R.: Balancing agility and discipline: Evaluating and integrating agile and plan-driven methods (2004)
7. Yin, A., Figueiredo, S., da Silva, M.M.: Scrum Maturity Model: Validation for IT organizations' roadmap to develop software centered on the client role, The 6th conference on software Engineering Advances, 21-29 (2011)
8. Hantos, P.: Agile Software Development in Defense Acquisition: A Mission Assurance Perspective. AEROSPACE CORP EL SEGUNDO CA, 2012.
9. Manifesto for agile software development, http://agilemanifesto.org
10. Eng, Siv Fern: Agility and Discipline: A Case Study in Incorporating More Balance to a Software Engineering Process, Portland state University, 2014.
11. Macit, Y., Tüzün E., İnce K., Aytekin A. İ: Büyük Ölçekli Bir Organizasyonda Uygulama Yaşam Döngüsü Yönetimi Uygulama Deneyimi, 8. Ulusal Yazılım Mühendisliği Sempozyumu (2014)
12. Dahmann, J., Gregorio, D., Modigliani, P.,: Systems engineering processes for agile software development, IEEE International Systems Conference, 351-355, 2013
13. Mordecai, Y., Dov, D.: Agile modeling of an evolving ballistic missile defense system with object-process methodology, IEEE International Systems Conference, 839-846, 2015
14. Oxenham, D.: Agile approaches to meet complex system of engineering challenges: A defense perspective, IEEE International Systems Conference, 1-6, 2010
15. Cohan, S.: Successful integration of agile development techniques within DISA, Agile Conference (AGILE), 2007.
16. National Defense Authorization Act For Fiscal Year 2010, Section 804, www.gpo.gov/fdsys/pkg/PLAW-111publ84/pdf/PLAW-111publ84.pdf