LAPIS – (LOGO Agile Process Improvement System)

Tuğrul Tekbulut[1], Ayhan İnal[1] ve Betül Doğanay[1]

[1] LOGO Yazılım, Kocaeli, Türkiye
{tugrul.tekbulut,ayhan.inal,betul.doganay}@logo.com.tr

Özet. Çalışmamızda yazılım geliştirme süreçlerinde son yıllarda ülkemizde de uygulanmaya başlanan çevik yazılm geliştirme metodlarının derlenmesiyle elde edilen özgün metodoloji üzerinde durulacaktır. Ana faaliyet alanı yazılım geliştirmek olan şirket/organizasyonlarının (Software Intensive Organisation) en önemli bağımsız yönetim değişkeni olan zamanın yönetilmesiyle zaman yönetim kültürünün yerleştirilmesi ve çevik metodlarda tanımı bulunan Story Point kavramına yeni bir yaklaşım açıklanmaya çalışılacaktır. Geliştirme sürecinde bulunan risklerin analitik veriler sayesinde öngörülebilir ve yönetilebilir hale getirilmesi için yapılan çalışmalar sonucunda şekillenen çevik yönetim sistemine ilişkin konular ele alınacaktır.

Anahtar Kelimeler: Yazılım Mühendisliği, Çevik Yazılım Metodolojileri, İstatistiksel Süreç Yönetimi

1 Giriş

Ana faaliyet alanı yazılım geliştirmek olan "Software Intensive Organisation" niteliğindeki şirketler için masrafların yaklaşık %80'i geliştirme faaliyetlerinden oluşmaktadır. Bu yüzden geliştirme süreçlerindeki en ufak iyileştirmenin katkıları çok büyük olmaktadır.

LAPIS, farklı metodolojilere dayanan ürün geliştirme yönetim modellerinden ve yalın felsefeden etkilenmiş, tarifinde LOGO'un tüm ürün geliştirme personelinin katkı sağladığı bir proje yönetim modelidir. LAPIS, çevik metodların yaklaşımına benzer şekilde proje kapsamının proje boyunca sürekli değişeceğini en baştan kabul eden bir yönetim şeklidir. Bu yüzden olabilecek değişiklikleri bir istisna olarak görmemektedir. Değişikliklerden kaynaklanan riskleri minimize edebilmek için kısa döngülü çıktı üretme ve geri bildirim düşüncesine dayanmaktadır.

2 LAPIS Komponentleri

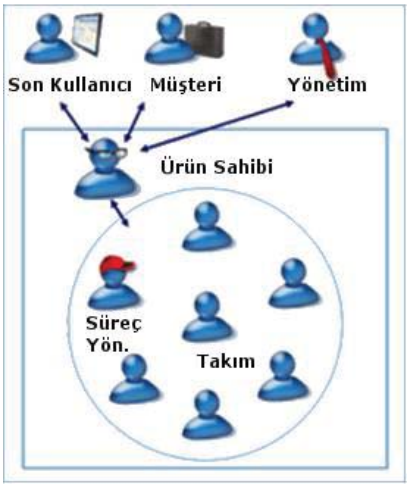
2.1 LAPIS'de Roller

LAPIS, çevik metodolojilerden biri olan "Scrum" metodolojisinde de kullanılan 3 temel rolü kendi modeline entegre etmiştir.

Ürün Sahibi (Product Owner) : Geliştirilmesi gereken tüm özellik ve fonksiyonları ürün iş yığınında (Product Backlog) derleyerek yapılan geliştirmelerden elde edilen karı (ROI - Return of Investment değerini) maksimize etmeyi hedefler. Ürün iş yığınında bulunan maddeleri önceliklendirerek sprint içeriğinin belirlendiği sprint planlama toplantısında geliştirme takımları (Development Team) ile geliştirme kapasitesini gözönünde bulundurarak yapılacak geliştirmelerin pazarlığını yapar. Geliştirme takımları tüm süreç boyunca bu pazarlığı sadece kendilerine atanmış ürün sahibi (Product Owner) ile yapmaktadır.

Süreç Yöneticisi (Process Master): Süreç adaptasyonu ile ilgili çalışmalar yürütür ve sürecin doğru işletilmesinden sorumludur.

Geliştirme Takımı (Development Team) : Sprint planlama toplantısında ürün sahibi (Product Owner) ile yaptığı pazarlık sonucu yapmayı kabul ettiği maddelerin (Sprint Backlog) sprint boyunca geliştirilmesinden sorumludur. Planlama toplantısında geliştirme takımları (Development team) madde kabul etmeme özgürlüğüne sahiptir. Ancak toplantıda kabul ettiği maddeleri sprint için belirlenen sürenin sonunda tamamlamakla yükümlüdür.



Resim 1. LAPIS'de bulunan roller ve birbirleriyle olan iletişim kanalları

LAPIS'de çevik metodolojilerinden farklı olarak aynı üründe birden fazla geliştirme takımı bulunmakta ve takım liderleri tarafından yönetilmektedir. Geliştirme takımının üyeleri "çapraz işlevli" (cross-functional) değildir. Test operasyonları test ekipleri tarafından ayrıca gerçekleştirilmektedir.

2.2 LAPIS'de Süreç

LAPIS ile zaman ekseni üzerinde işletilen süreç detayları Resim 2 de gösterilmiştir ve resim üzerinde numaralandırılan işlemlere ait açıklamalar aşağıdaki gibidir.

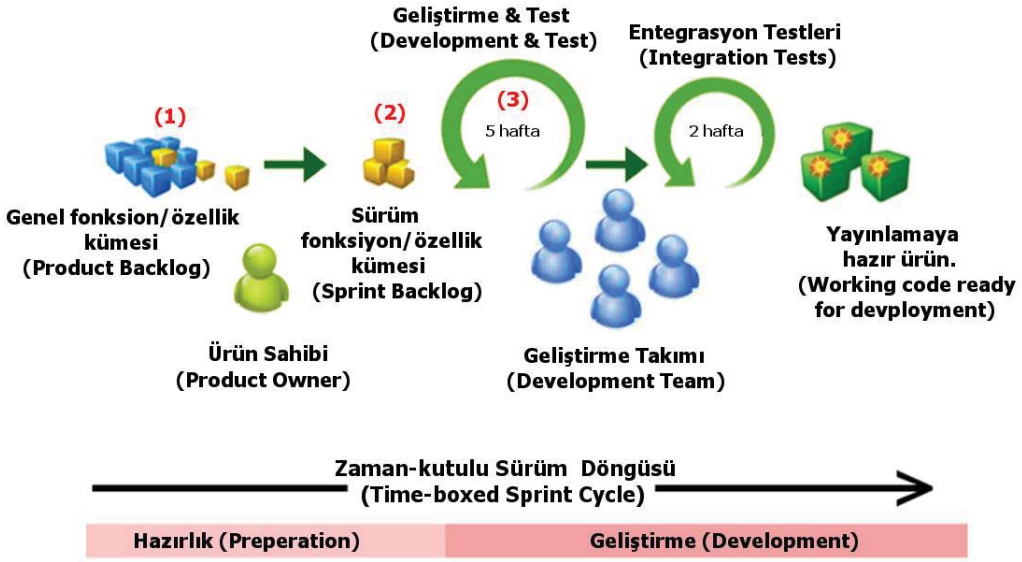
- Ürün iş yığını (Product Backlog) oluşturulur ve ürünün karını (ROI) maksimize edecek şekilde yönetilir. *(1)*

- Sprint başlamadan önce ürün sahibi, ürün iş yığınında bulunan maddeleri önceliklendirir ve sprint planlama toplantısında, geliştirme takımları ile önceliklendirilmiş maddelerden hangilerinin izleyen sprint içeriğine dahil edileceğinin pazarlığını yapar. Geliştirme takımı her madde için harcanması tahmin edilen eforlara (maddenin sahip olduğu Story Point değerlerine) göre kapasitesi kadar madde almayı kabul eder. Yapılmasına karar verilen maddeler sprint görev yığını (sprint backlog) olarak ilan edilir. *(2)*

- Sprint boyunca sprint görev yığınında bulunan maddeler geliştirilir. Sprint'in tamamlanma tarihi sabittir (5+2 hafta), başlatılan bir spring boyunca değiştirilemez (time-boxed) *(3)*

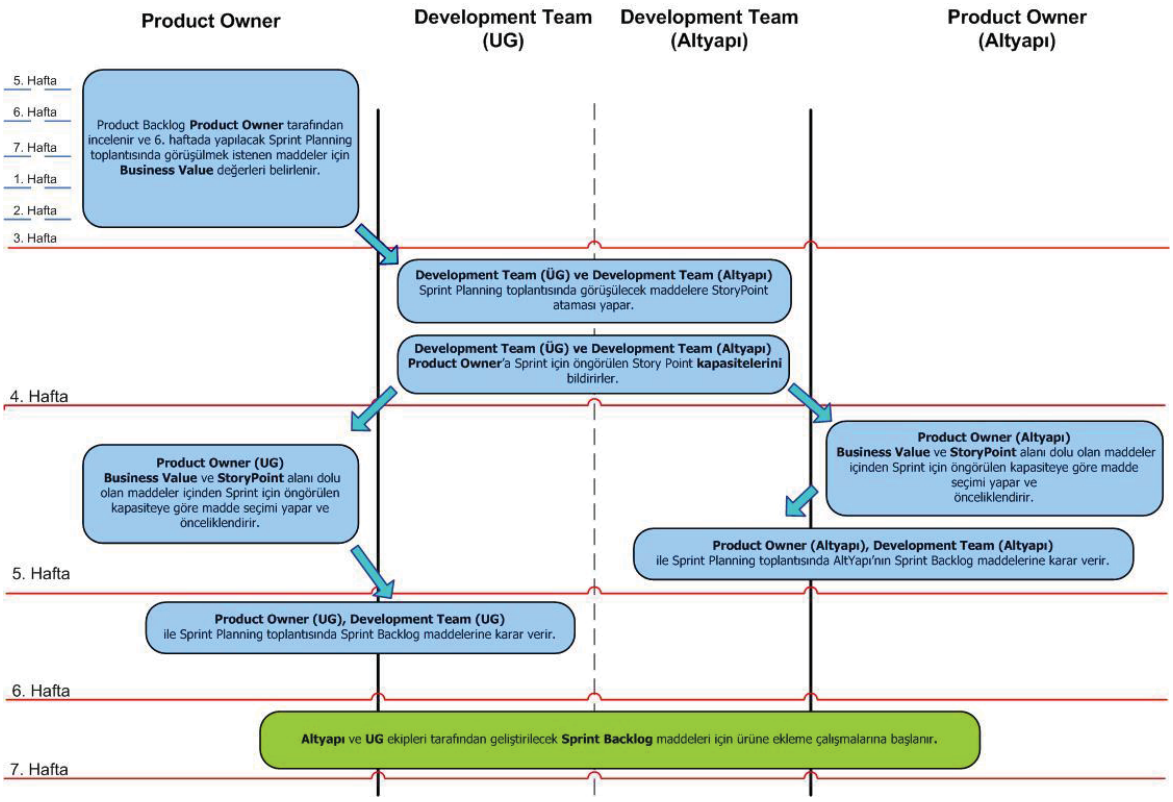
- Sprint sonunda geliştirme takımı, sprint görev yığınındaki (sprint backlog) tüm geliştirmeleri içeren bir ürün teslim eder. Ürün sahibine yapılanların demosunu sunar, (Sprint Review Meeting) sürekli iyileştirme çalışmaları kapsamında süreçte yapılabilecek iyileştirmeler sürüm değerlendirme toplantısında (Sprint Retrospective Meeting) değerlendirir ve uygulayabileceklerini bir sonraki sprint için hayata geçirir.

- Entegrasyon testlerinin yapıldığı +2 haftasında geliştirme takımları test ekibinden gelen bug-fix taleplerini yapmakla beraber bir sonraki sürümün geliştirmesine başlar.



Resim 2. LAPIS süreci

Resim 2'de bulunan "Hazırlık (Preperation)" dönemine ait çalışmalar için yürütülen detaylı süreç takvimi Resim 3'de gösterilmiştir.



Resim 3. Sprint içerik belirleme çalışmalarına ait süreç takvimi

Geliştirme döneminde harcanan efora ait geri bildirimler, geliştirme takımlarının bulunduğu ortama yerleştirilen ekranlarda gösterilen "Yanık Grafikler" (BurnDown Chart) yardımıyla gerçekleştirilmiştir. Yanık grafikler sayesinde sağlanan sürekli geri bildirimin proje takibinine ve çalışan motivasyonuna olumlu katkıları olmuştur. Yıllar içerisinde grafiklerde izlenen iyileşmeler bunu kanıtlar niteliktedir. (Resim 5 ve Resim 6)

2.3 LAPIS'de Zaman Yönetimi

Tüm yazılım geliştirme projeleri kompleks sistemlerdir. Uzun süreli/büyük "Waterfall" projeleri yönetilmesi güç riskler barındırmaktadır. LOGO yazılım, misyon kritik (mission critical) ürünler üretmektedir. Tüm ürünler yaşayan ve üzerinde devamlı ek geliştirmeler yapılan ürünlerdir. Bu yüzden risk yönetimi son derece önemlidir. Yazılım geliştirme projeleri diğer birçok mühendislik projelerinden (Örneğin, uçak yapım projesi) farklı olarak sürekli evrilmeye imkan tanımaktadır. Bu yüzden, kısa süreli geliştirme sürelerinden sonra sonuçların hedeflenen sonuçlara uymaması durumunda yeniden tasarlanabilir ve geliştirilebilmektedir. Yapılan yanlış tasarımlardan kaynaklanan riskleri yönetmek büyük "Waterfall" projelerinde neredeyse imkansızdır. Planlamaların 1 haftalık saat dilimlerine bölünmesini öngören LOGOWeek kavramı kısa vadeli planların eksiksiz gerçekleşmesini sağlamak için, zamanında bitirilememesi riskini veya hatalı geliştirilme riskini minimize etmek için getirilmiştir.

Şirket kültürüne zaman-yönetim nosyonunun kazandırılması ve şirketin ekosistemi için ortak bir sistem saati algısının oluşturulabilmesi hedeflenmiştir. LOGOWeek ile kısa süreli "Waterfall" uygulanmaktadır.

2.4 LAPIS'de Story Point

Çevik yazılım metodolojilerinde bulunan Story Point iş birimi LAPIS'e entegre edilmiştir. Story Point bir fonksiyon/özelliğin geliştirilmesi ve hatanın düzeltilebilmesi için gerekli işi tarif eden nümerik bir değerdir. Scrum gibi çevik metodolojilerde tüm işler genelde fibonacci serisinden oluşan bir puan

kümesi ile değerlendirilmeye çalışılmaktadır. İş biriminin belirlendiği toplantılar esnasında "Scrum Pokeri" denilen bir çeşit oylama ile yapılacak geliştirmenin iş birimi tayin edilmeye çalışılmaktadır.
SIO niteliğindeki şirketler için bu yöntem uzun vadede ölçmeyi zorlaştıran ve birden fazla ekibi, özellikle farklı platformlarda geliştirme yapan ekipleri aynı ölçü birimi ile değerlendirebilmeyi zorlaştırmaktadır.
Bu yüzden LAPIS ile her maddenin iş yükünün analitik ve süreklilik arz eden veriler ile belirlenmesine olanak sağlayan Story Point tablosu yaratılmıştır.
"LAPIS Story Point" tablosu yazılım geliştirme aktivitelerini bir tablo üzerinde barındırmaktadır. Geliştirme takımları geliştirilecek olan işi tabloda bulunan aktivitelere (küçük iş parçacıklarına) bölüp herbir aktivitenin sahip olduğu Story Point değerlerini toplayıp iş için gerekli toplam Story Point değerini belirlemektedir.
Bu yöntem ilgili geliştirmenin alt geliştirmelerini ve her birinin iş birimlerini ölçmekte ve analitik değerlendirmelerde kolaylık sağlamaktadır.
Tarif edilen işin "alt işleri" kayıt altında tutulduğu için yine çevik yöntemlerde var olan "Definition of Done" kümesini içermektedir. "Definition of Done" yapılan işin ürün sahibi tarafından hangi koşullar altında bitmiş kabul edileceği detayını içermektedir.

2.4.1 Story Point Hesaplama Yöntemi

Story Point hesabı için LAPIS kapsamında geliştirilen Story Point tablosundan yararlanılmaktadır. Tablo geniş kapsamlı bir yazılım geliştirme aktivite kümesine sahiptir. Herbir aktivite için belirlenmiş taban puan (zorluk derecesi : kolay) ve zorluk dereceleri için belirlenmiş katsayılar bu tablo üzerinde tutulmaktadır.
Tüm işler, aktivite tipi, zorluk ve miktar cinsinden tanımlanarak yazılım geliştirme faaliyetine ait toplam Story Point değeri hesaplanmaktadır.

Örnek Yapılacak geliştirme faaliyetinin 2 adet 1.derece zorlukta yeni rapor içermesi durumunda aşağıdaki tabloya göre seçilmesi gereken aktivite tipi : "Rapor – yenisinin oluşturulması" , zorluk derecesi : zor (1.derece) ve adet 2 olmalıdır.

Toplam Story Point : 35x4x2 = 280

		Zorluk					
Kodlama için Story Point tablosu		***Kolay***	***Normal***	***Zor (1.derece)***	***Zor (2.derece)***	***Zor (3.derece)***	***Zor (4.derece)***
	Katsayı	*1*	*2*	*4*	*8*	*12*	*16*
Aktivite tipi	*Taban Puan*						
Rapor - Yenisinin oluşturulması	*35*			2			
Rapor - Mevcuda alan eklemek	*5*						
Rapor - Performans iyileştirme	*8*						
						Toplam Kodlama SP	**280**

Tablo1. Story Point hesaplama tablosu.

2.5 LAPIS'de "Business Value" Değeri

"Value – Based Software Engineering" konusu olan "Business Value" (BV) değeri, müşteriye ulaşan iş çıktısının müşterinin ihtiyacını karşılayan, müşteri odaklı bir iş çıktısı olmasını hedeflemektedir. LAPIS, geliştirilen her iş için verilebilecek BV değerini 0-500 rakamları arasında olacak şekilde belli bir hesaplamaya dayandırmıştır. Ürünün Sprint Backlog'unda bulunan tüm maddeler için BV değerini hesaplanması için ihtiyaç duyulan 5 soru cevaplandırılmak zorundadır.

Her bir soruya/kriter'e verilen cevabın bir nümerik karşılığı ve BV değerini etkileyecek bir katsaysı mevcuttur. Tüm cevaplardan elde edilen değerlerin toplamı, ilgili işin "Business Value" değeri olarak kabul edilmektedir. BV atama işlemi ürün sahibi tarafından yapılmaktadır.

2.5.1 BV Hesaplama Yöntemi

Sprint kapsamına alınmak istenen tüm hata/istekler için ürün sahibi tarafından BV ataması yapılmaktadır. Atama öncelik, şiddet, müşteride rastlanma sıklığı, rakipte bulunma durumu, yapılmasının kazandıracakları ve yapılmamasının kaybettirecekleri kriterleri için sunulan seçenekler için yapılan seçimlere göre yapılmaktadır. Herbir kritere ait seçeneğin nümerik karşılığı bulunmakta ve yapılan seçimlerin nümerik değer karşılıklarının toplamı hata/isteğin BV değeri olarak kabul edilmektedir.

Örneğin, "şiddet" kriteri için sunulan seçenekler ve nümerik karşılıkları aşağıdaki tablodaki gibidir. (Şiddet kriterinin BV'ye olan katkısı bir katsayı ile belirlendiği için kusuratlı değerler bulunmaktadır.) Seçenekler açıklama alanındaki açıklamaya göre seçilmekte ve sınıflandırılmaktadır. Örneğin bir hatanın "kritik" olarak kabul edilebilmesi için açıklama alanındaki "Sistem çöküyor, data kayboluyor, data bozuluyor veya kritik bir fonksiyon/özellik kullanılamaz durumda." tarifine uyması gerekmektedir.

Şiddet kriteri	Açıklama	BV hesabına etkisi
İstek - Zorunlu	Üründe mutlaka bulunması gereken, müşterilerin kritik faaliyetleriyle ilgili bir fonksiyon/özellik.	125
İstek - Faydalı	Bu fonksiyon/özellik olmadan da müşterinin işleri yürüyor ama zorluk oluyor. Olursa performans ve kullanılabilirlik artar.	62,5
İstek - Olsa İyi Olur	Ürünün temeliyle ilgili olmayan bir fonksiyon/özellik ama olması durumunda ürünü daha çekici ve kullanışlı hale getirir.	12,5
İstek - Gereksiz	Ürüne bu fonksiyon/özelliğin eklenmesine gerek yok.	0
Hata - Kritik	Sistem çöküyor, data kayboluyor, data bozuluyor veya kritik bir fonksiyon/özellik kullanılamaz durumda.	125
Hata - Majör	Önemli bir fonksiyon/özellik hatalı çalışıyor, hataya dolaylı çözüm bulunamıyor veya hata restart gerektiriyor.	93,75
Hata - Minör	Fonksiyon/özelliğin hatalı olması tahammül edilebilir etki yapıyor, hataya dolaylı çözüm var veya kullanım zorluğu yaşanıyor.	43,75
Hata - Önemsiz	Yazım hatası, terminoloji hatası, görsel bozukluk veya ürünün bir fonksiyon/özelliğini engellemeyen hata.	12,5

Tablo2. "Şiddet" kriterinin BV değerine olan katkısını içeren tablo.

3 OTEQ (Overall Team Efficiency, Quality) formülü:

Yalın Üretim'de kullanılan OEE (Overall Equipment Effectiveness) formülünden esinlenerek OTEQ (Overall Team Efficiency,Quality) formülü ile kalite faktörünü de içeren yeni bir performans metriği elde edilmeye çalışılmıştır.

OEE formülü,

OEE = Uygunluk Durumu x Performans x Kalite
(orjinal OEE = Availability x Performance x Quality)

olarak tanımlıdır. LAPIS kapsamında formül bileşenlerini uyarlayarak OTEQ formülüne varılmıştır.

(i) *"Uygunluk Durumu" (Availability) parametresi :*

Orjinal OEE'ye göre Uygunluk Durumu (Availability) parametresi,

Uygunluk Durumu = Gerçekleşen üretim süresi/Planlanan üretim süresi
(orjinal Availability = Operation Time / Planned Production Time)

olarak tanımlıdır. İnsan performansının sonsuz olabileceği kabulu ile uygunluk durumu (Availability) = 1 olarak kabul edilmiştir.

(ii) *"Performans" (Performance) parametresi :*

Orjinal OEE'ye göre Performans (Performance) parametresi,

Performans = (Üretilen parça sayısı/Üretim süresi) / İdeal üretim oranı
(orjinal Performance = (Total Pieces/Operating Time)/Ideal Run Rate)

olarak tanımlıdır. Performans parametresi uyarlanarak n adet yazılımcının t günde yaktığı Story Point olarak aşağıdaki gibi tanımlanmıştır.

$$\sum \text{Story Point} \;/\; (\sum \#\text{Yazılımcı})(\#\text{ Gün})$$
$$(\sum \text{Story Point} \;/\; (\sum \#\text{ of Coders})(\#\text{ of Days}))$$

(iii) *"Kalite" (Quality) parametresi :*

Orjinal OEE'ye göre Kalite (Quality) parametresi,

Kalite = Kaliteli(Sağlam) parça sayısı / Toplam üretilen parça sayısı
(orjinal Quality= Good Pieces/Total Pieces)

olarak tanımlıdır.

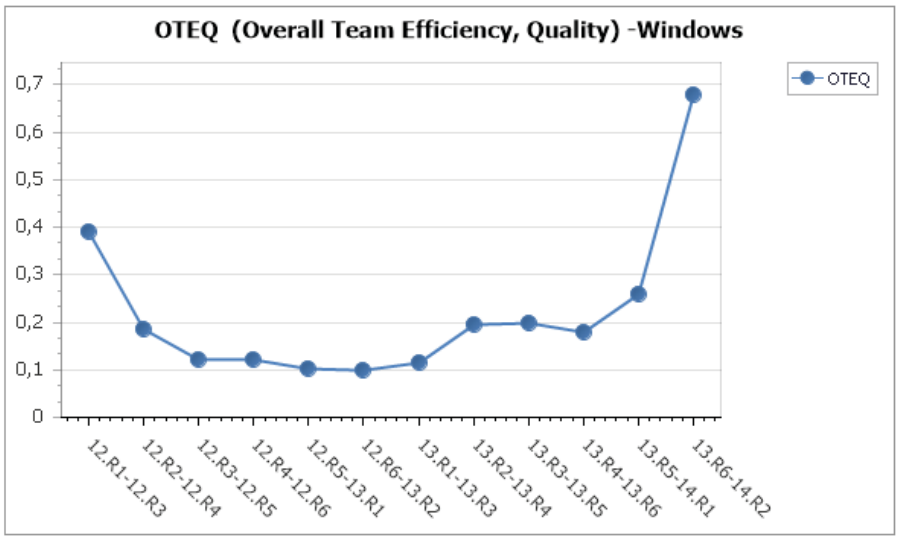
Kalite parametresi olarak müşterilerden bildirilen hatalara ait zarar puanlarının toplamı, bölen olarak formüle dahil edilmiştir. Bir hataya ait zarar puanı, hatanın yol açtığı zarara ait atanan nümerik değerdir. Bir zarar puanının alabileceği değerler 1 ile 13 arasında değişebilmektedir.

OEE formülünün OTEQ formülüne uyarlanmış hali :

$$OTEQ = (1)(\sum \text{Story Point} \;/(\sum \#\text{Yazılımcı})(\#\text{ Gün}))(1/\sum \text{Hata Zarar Puanı})$$

$$(OTEQ=(1)(\sum \text{Story Point} /(\sum \#\text{Coders})(\#\text{Days})(1/\sum \text{Damage Value of Defects}))$$

Resim 4, LOGO Yazılımın bir ürün hattından toplanan verilere göre hesaplanan OTEQ değerlerine ait grafiği içermektedir. Grafik 2012-2014 yılları arasındaki zaman dilimine ait verileri içermektedir.



Resim 4. Windows ürün hattı OTEQ grafiği

3.1 Zarar Puan hesaplama yöntemi

Müşteriler tarafından bildirilen tüm hatalar ortaya çıkma tarihlerine göre "yeni" veya "eski" hata olarak sınıflandırılmaktadır. Bildirilen hata, müşteri tarafından bildirilme tarihinden bir yıl önceki sürümde de var ise "eski" olarak kategorize edilmektedir. Eğer hata bir yıl önceki sürümde gerçeklenemiyorsa hata "yeni" olarak kabul edilmektedir.

"Zarar Puanı" hesabına sadece "yeni" hatalar dahil edilmekte ve ürün sahipleri (Product Owner) tarafından "Business Value" atama işlemi kapsamında verilen "Şiddet" (Severity) değeri dikkate alarak aşağıdaki tabloya göre nümerik bir değer atanmaktadır.

Business Value - Şiddet (Severity)	**ZP değeri**
Hata - Kritik	13
Hata - Majör	8
Hata - Minör	3
Hata - Önemsiz	1
İstek - Zorunlu	5
İstek - Fayda	0.0001
İstek - Olsa iyi olur	0.0001
İstek - Gereksiz	0.0001

Tablo3. Zarar puanı tablosu.

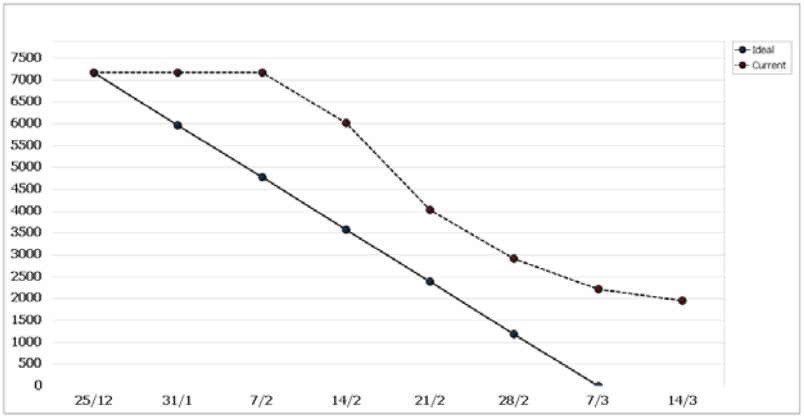
BV değeri sadece ürün sahibi tarafından sürüme dahil edilmek istenen hata/istekler için belirlenmektedir. Sürüme dahil edilmeyen ve bu sebepten dolayı ZP değeri

olmayan hatalara hesaplamanın yapıldığı dönem için elde edilen ortalama ZP değeri atanarak OTEQ hesabına dahil edilmektedir.

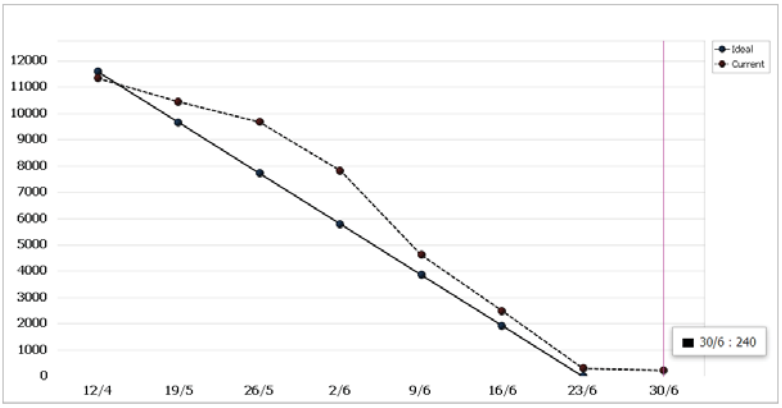
4 LAPIS kapsamında yapılan ölçümler

Tarif edilen sistemin süreçlere ve katkıda bulunduğu bazı ana başlıklara ait somut veriler bulunmaktadır. Yapılan ölçümlerin sonuçları sisteme girdi olarak değerlendirilerek sürekli iyileşmeye önemli etkileri olmaktadır. LAPIS kapsamında bulunan bazı ana başlıklar ve elde edilen iyileşmelere ait örnekler aşağıdaki gibidir.

(i) Tüm ürün geliştirme çalışanlarına sürekli geri bildirimin sağlanmasıyla elde edilen performans artışı ve sürüm yayınlama tarihlerindeki olası sarkamaların azaltılması.



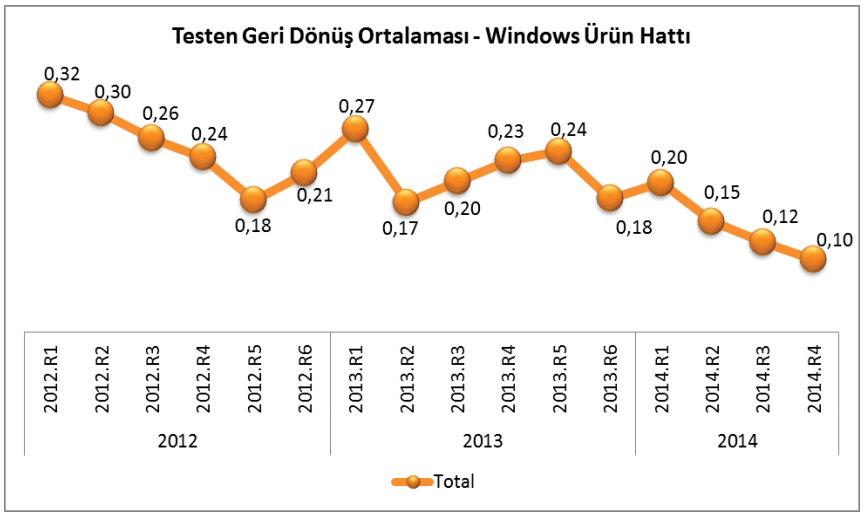
Resim 5. 2011. R2 sürümüne ait "Yanık Grafik"



Resim 6. 2014. R4 sürümüne ait "Yanık Grafik"

Resim 5 ve Resim 6 'da sunulan yanık grafiklerdeki kesikli çizgi gerçekleşen storypoint tükemini kesiksiz çizgi ise idealinde beklenen tüketimi göstermektedir. 2013.R1 sürümüne ait yanık grafikte gerçekleşen tüketimdeki iyileşme net bir şekilde görülmektedir.

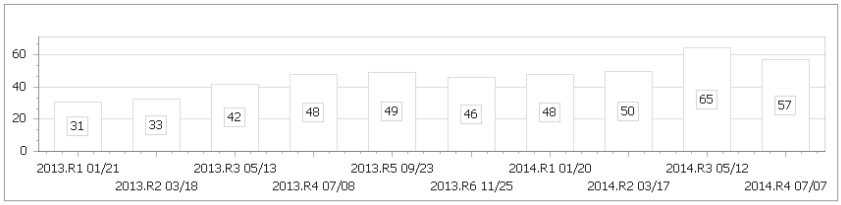
(ii) Sprint boyunca sürüm içeriklerindeki değişikliklerin minimize edilerek yazılım geliştirme kalitesinin iyileştirilmesi. Geliştirme esnasında bölünen işlerin azalmasını sağlayarak konsantrasyonun artırılması.



Resim 7. Sürüm bazında geri gönderilme ortalamaları (adet)

Resim 7, yazılım ekipleri tarafından test ekiplerine gönderilen ve bug-fix için yazılım ekiplerine geri gönderilen ortalama madde sayısına ait verileri göstermektedir (Return count). Geri gönderilme rakamlarında azalma olduğu gözlemlenmiştir. Yazılım ekipleri tarafından test edilmek üzere test ekiplerine gönderilen 100 maddeden 10 madde hata içermektedir. Çalışma başlangıcında bu oran 100'e maddeye 32 madde gibi bir değere sahipti.

(iii) Ortalama storypoint tüketimine ait verilerin değerlendirilerek personel ihtiyacına yönelik tahminlerin yapılabilmesi.



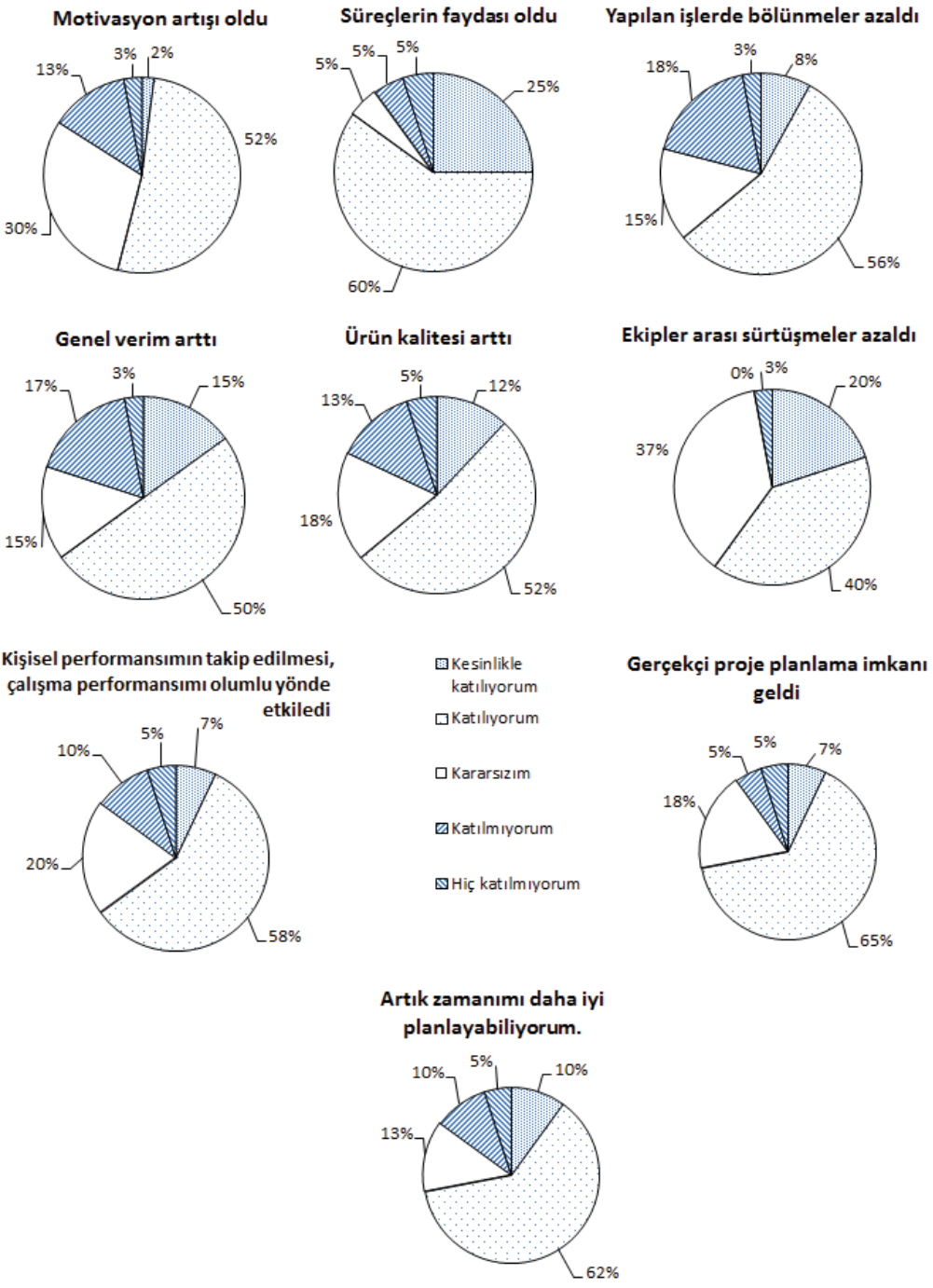
Resim 8. Günlük ort. adam/storypoint tüketimi – Windows ürün hattı

Adam başı günlük ortalama storypoint tüketim verisi, yıllık stratejik geliştirme planlarına önemli bir girdi parametresi ve eleman alımı için tetikleyici niteliğe sahiptir. LAPIS öncesi nümerik değerler olmadan tahminler yapılmaya çalışılıyordu.

(iv) LAPIS'in ilk zamanlarında sprint planning toplantıları ortalama 1 gün sürerken artık 2-3 saatlik kısa süreli toplantılarla sürüm içeriği belirlenebilmektedir.

5 Anket sonuçları

Ürün geliştirme ekiplerine LAPIS'in hayatlarına getirdiği değişiklikleri ölçmeyi amaçlayan bir anket düzenlendi. Toplamda 39 kişinin katılımı ile gercekleşen anket sonuçları aşağıdaki gibidir.



Resim 9. LAPIS anket sonuçları

	Kesinlikle katılıyorum	Katılıyorum	Kararsızım	Katılmıyorum	Hiç katılmıyorum
Motivasyon artışı oldu	2%	52%	30%	13%	3%
Süreçlerin faydası oldu	25%	60%	5%	5%	5%
Ekipler arası sürtüşmeler azaldı	20%	40%	37%	0%	3%
Yapılan işlerde bölünmeler azaldı	8%	56%	15%	18%	3%
Genel verim arttı	15%	50%	15%	17%	3%
Ürün kalitesi arttı	12%	52%	18%	13%	5%
Gerçekçi proje planlama imkanı geldi	7%	65%	18%	5%	5%
Artık zamanımı daha iyi planlayabiliyorum.	10%	62%	13%	10%	5%
Kişisel performansımın takip edilmesi, çalışma performansımı olumlu yönde etkiledi	7%	58%	20%	10%	5%

Tablo 4. LAPIS anket sonuçları

6 Sonuç

LAPIS ile ürün geliştirme faaliyetlerini kapsayan sürecler ölçülebilen, kestirilebilen ve üzerinde mutabakat oluşturulabilen sistemlere bağlanmıştır. Yapılan iyileştirmeler tüm ekosistemi etkilemiş ve şirket çıktılarına güven artışını sağlamıştır.
Şirket ekosistemine takvim yıllarının başlarında tüm yıl için planlanmış sürüm çıkışlarına ait tarihler duyurulmaya başlanmıştır. Yazılım sektöründe -neredeyse ön kabul olan- proje sürelerindeki sarkmaların önüne geçilmiştir. Bu sistemle son 3 yılda 35 iş günlük standart proje dilimlerinde ortalama 0,3 günlük sapma süresi sağlanmıştır.

Standart bir yazılım iş birimi (Story Point) geliştirilerek, kapasite planlama, maliyet hesaplama, teslimat tarihi belirleme gibi yazılım sektöründe sorun olan konular analitik olarak hesaplanabilir, izlenebilir hale getirilmiştir. Şirketin tepki süresinden kaynaklanan müşteri memnuniyetinin arttığı gözlemlenmiştir. Sistemli çalışma kültürünün iş ortaklarına ve iş ortakları üzerinden müşterilere yansıması rekabet gücünü olumlu yönde etkilediği gözlemlenmiştir.

Kaynakça

1. “Microsoft Secrets” Michael A. Cusumano, Richard W.Selby

2. “The Toyota Product Development System” James M. Morgan , Jeffery K.Liker

3. “Lean – Agile Software Development Achieving Enterprise Agility” Alan Shaloway, Guy Beaver, James R. Trott

4. “Agile Product Management with Scrum – Creating Products that Customers Love” Roman Pichler

5. ”Coaching Agile Teams – A Companion for Scrum Masters, Agile Coachers, and Project Managers in Transition” Lyssa Adkins

6. “Continuous Integration – Improving Software Quality and Reducing Risk ” Paul M. Duval with Steve Matyas, Andrew Glover

7. “Winning at new Products – Creating Value through Innovation ” Robert G. Cooper

8. “ Continous Delivery – Reliable Software Releases Through build, Test, and Deployment Automation” Jez Humble , David Farley