Takım Çalışmasının Yazılım Geliştirme Süreçlerine Etkisi: Metodolojilerin Takımlar Üzerinde Etkisi

Onur Osman GÜLE[1]

[1]Bilgisayar Mühendisliği Bölümü, Sakarya Üniversitesi Bilgisayar ve Bilişim Bilimleri Fakültesi, Sakarya, Türkiye
onur.gule@ogr.sakarya.edu.tr

Özet – Günümüzde yazılım projelerinde farklı farklı metodolojiler kullanılmaktadır. Bu metodolojiler öncelik olarak projeyi zaman ve bütçe konusunda rahatlatmak amacıyla da olsa metodolojilerin takımlar üzerindeki etkisi kaçınılmazdır. Farklı metodolojilerin ve bu metodolojilerin takımlar üzerindeki etkisi ve buna bağlı olarak takımın yazılım geliştirme sürecindeki etkileri değişebilmektedir. Bu etkiler olumlu ya da olumsuz olabilmektedir. Herhangi bir problem yaşamayan bir takımda tamamen olumlu etkiler göstermekte ve bu olumlu etkiler maddi ve manevi olarak zaman ve bütçeyi olumlu yönde ilerletmektedir. Temel etkenlerden birinde problem yaşayan projelerde ise takım ve proje yöneticisine bağlı olarak olumsuz etkiler göstermekte ve bu olumsuz etkiler ışığında maddi ve manevi olarak zaman ve bütçe olumsuz yönde ilerletmektedir. Bu çalışmada öncelikle temel metodolojiler incelenecek olup bu metodolojiler ile ilgili geçmişteki araştırmalar da incelenip bu metodolojilerin takım çalışmalarına etkileri temel alınarak projede oluşabilecek riskler ve projeyi olumlu veya olumsuz yönde ilerletebilecek etkenler ortaya konulmaktadır.

Anahtar Kelimeler —– Şelale Modeli, Yinelemeli Şelale Modeli, Çevik Yöntemler, Yalın Model, Scrum, Takım Çalışması, Metodolojilerin Etkileri

Effect of Teamwork on Software Development Processes: Effect of Methodologies on Teams

Summary – Today, different methodologies are used in software projects. Although these methodologies are primarily intended to ease the project in terms of time and budget, the impact of the methodologies on the teams is inevitable. The impact of different methodologies and these methodologies on the teams and accordingly the effects of the team on the software development process may change. These effects can be positive or negative. It shows completely positive effects in a team that does not have any problems, and these positive effects materially and spiritually advance the time and budget. In projects that have problems in one of the main factors, it shows negative effects depending on the team and the project manager, and in the light of these negative effects, it advances time and budget negatively materially and spiritually. In this study, first of all, basic methodologies will be examined, and the previous researches related to these methodologies are examined and the risks that may occur in the project based on the effects of these methodologies on team studies and the factors that can advance the project in a positive or negative way are revealed.

Keywords —– Waterfall Model, Recursive Waterfall Model, Agile Methods, Lean Model, Scrum, Teamwork, Effects of Methodologies

1. GİRİŞ

Günümüzde yazılım projelerinde kullanılan PMI standartlarına uygun metodolojilere bakıldığında işgücünün yoğun olduğu gözlemlenebilir. Yaygın olarak kullanılan metodolojiler arasında Waterfall(Şelale), Agile(Çevik) ve Scrum modelleri bulunmaktadır. Her ne kadar Waterfall modelinde yönetilen bir projenin kişisel olarak yürütülmesi mümkün olsa da çevik yöntemlerle yönetilen projelerin kişisel olarak yürütülmesi, projeyi teslim etme açısından bir hayli zor olacaktır. Scrum gibi çevik yöntemler, tamamlanması beklenen işler birbirinden ayrıldığı için takım çalışmasına daha uygun, yerine göre düşük maliyetli, düşük riskli ve yüksek verimli yöntemler olarak sınıflandırılır. Günümüzdeki girişim projelerinde ilk arananlardan biri nitelikli ve uyumlu takımlar olmaktadır. Her geçen gün zorlaşan rekabet ile tüm firmalarda verimliliğin artması ile projelerin performansının yükselmesi ihtiyacı doğmuştur. Bu durumun yarattığı arayışlar, birey odaklı projelerin takım odaklı projelere evrilmesine yol açmıştır. Her ne kadar grup çalışmalarının bazen düşük üretkenlik, zayıf karar mekanizmaları gibi olumsuz sonuçları ortaya çıkabilse de birçok güncel çalışmada takım çalışmasının eş zamanlı olarak üretkenliği ve çalışan memnuniyetini artırdığını ortaya koymuştur. Araştırmacılar takım çalışmasının daha yüksek verimlilik, daha düşük emek maliyeti getirdiğini ve bireysel performansı kesin bir şekilde artırdığını gözlemlemişlerdir.
Tüm bu araştırma çalışmaları bireysel modellenmiş organizasyon yapısından uzaklaşarak takım çalışması şeklinde modellenmiş organizasyonlara geçişin kaçınılmaz olduğunu göstermiştir. Takım oluşumu için ve takım performansının artması için bazı kavramların incelenmesi gerekliliği doğmaktadır.

2. BAŞARILI PROJELERDEKİ ÇALIŞMALAR

Başarılı projelere Google, Facebook, Microsoft gibi örnekler verilebilir. Tüm bu şirketler bir ekip ruhu çerçevesi altında doğmuşlardır ve bu çerçevede hızla büyümeye devam etmektedir. Şu anda en başarılı, büyük şirketlerden biri olan Google'ı örnek verebiliriz takım çalışmasının başarılarına. Google, 2019 yılının son çeyreğinde çalışan sayısını 118.899 olarak açıklamıştır[4]. Google, yalnızca ana görevi olan arama motoru işlevi dahilinde birçok farklı projeye imza atmıştır. Çalışan sayıları da birçok projede çalışan, yönetici, personel olarak, takım olarak bu şekilde dallanmıştır. Google, 1996'da Amerika Birleşik Devletleri'ndeki Kaliforniya eyaletindeki Stanford Üniversitesi'nde doktora öğrencileri olan Larry Page ve Sergey Brin'in araştırma projesi olarak hayata geçirildi. Google şu anda kocaman bir şirket olduğundan projeleri yönetmen bir hayli zor olmakta. Proje yönetiminde tüm projelerde olmasa da bazılarında Çevik, Scrum metodolojilerini kullandıklarını açıklamışlardır.

Şirketlerin Projelerinde Agile Metodolojisi Kullanma Oranları[5]	
Paypal	56%
Spotify	52%
Adobe	39%
Apple	26%
Microsoft	25%
Cisco	25%
Twitter	24%
Oracle	23%
Coca Cola	13%
Google	10%
IBM	9%
Tesla	9%
Facebook	8%
Amazon	8%
SpaceX	5%
McDonald's	0%

Paypal, Spotify gibi daha yenilikçi şirketler Çevik metodolojisini kullanmakta çok iyi yerlerde olurken Apple, Microsoft, Twitter, Oracle gibi şirketler daha az kullanmakta. Google, Cisco, IBM gibi ürünleri eskiden beri var olan şirketler ise bu tür metodolojilere ayak uydurmakta zorlanmakta.

2.1 TAKIM KAVRAMI

Teknolojik gelişmelerin her geçen gün ilerlediği, rekabet olramının yoğunlaşarak arttığı günümüz mevcut koşullarında işletmeleri küresel piyasalarda ileriye doğru götürmek ve rekabet edebilecek düzeye getirmek için bireysel olarak hareket etmek yetersiz kalmaktadır. Bunun için gerekli olan bilgi, beceri, deneyim ve yeteneklere sahip olan kişiler bir araya getirilerek bir organizasyon içinde takım oluşturulmaktadır. Buna bağlı olarak günümüz dinamik piyasa koşullarında problemlerin çözümünde takımlar etkin olarak kullanılmaktadır. Bu bakımdan firmalar, daha etkin ve verimli bir şekilde rekabet edebilmek adına yeniden yapılandırılarak çalışanların yeteneklerini kullanmak için takım oluşturmaya yöneldiler.

2.2 TAKIMLARIN BAŞARI KRİTERLERİ

Yapılan araştırmalara göre başarılı takım çalışmasının genel olarak 7 adet kriteri mevcuttur.

- **Aynı amaca hizmet etmek**
  Takım çalışması için bir araya getirilecek olan kişilerin aynı amaca hitap ediyor olması birinci olmazsa olmazdır. Aynı yönde koşan insanların kazandıracakları ile farklı yönlere koşan insanların kazandıracakları aynı değildir. Bu nedenle amaç birliği ilkesine dikkat edilmelidir.

- **Kuralları başında belirlemek**
  Takım içerisinde kuralların önceden açık bir şekilde belirlenmesi, iletişimin ve gelişimin daha sağlıklı bir şekilde ilerlemesini sağlayacaktır. Grup kararlarının nasıl ve hangi yöntemlerle alınacağı başta olmak üzere, diğer ekip üyeleri de görüşlerini açıkça dile getirmelidir. Kuralların belirlenmediği bir takım çalışmasında anlaşmazlıklar, sorunların zamanında çözülememesi, karar verme süreçlerinin uzaması ve ekip üyeleri ayrılıkları gibi sorunlar yaşanmaktadır.

- **Gelişim planı oluşturmak**
  Takım çalışmasının amacına nasıl ve ne kadarlık bir süreçte ulaşacağını belirlemesi açısından gelişim planı oluşturulmalıdır. Bu planda görev dağılımları ve süreçlerin belirlenmesi gereklidir. Bu sayede ekip gelişmeleri izleyebilecek bir dokümana sahip olacaktır.

- **Verimli olabilecek takım üyelerini belirlemek**
  Takım liderleri, şirketin hali hazırda devam eden işleri ve projenin gerektirdiği yetkinlik ve becerilere sahip olan çalışanların uygunluk durumuna göre belirlemesi gerekmektedir. Zira hali hazırda diğer işlerden dolayı çok yoğun çalışan bir takım üyesi, projede verimli çalışamayacaktır.

- **Görevlerin eşit bir şekilde dağıtılması**
  Gerekli Takım çalışması sırasında görev dağılımı çok önemlidir. Her çalışana aynı oranda görev vermek kimi zaman mümkün olmayabilir. Ancak genel dağılımlar sırasında herkese eşit davranmaya çalışmak ve herkese yapabileceği görevler vermek gerekir.

- **Uyumlu olmak**
  Takım içerisindeki kişilerin birbirleri ile olan uyumları da takım çalışmasındaki kalite ve başarı ile doğru orantılıdır. Takım arkadaşları ile uyum hiçbir şekilde uyum sağlayamamış bir kişi alanında ne kadar başarılı olursa olsun çalışmayı ileriye taşıyamaz. Takım çalışmalarına başlamadan önce uyumlu bir ortam yakalamak adına geziler, yemekler ve ya aktiviteler düzenlenebilir.

- **Farklı görüşlerin olması**
  Takım üyeleri yüksek bir uyuma sahip olsa da aynı amaca hitap etse de her biri aynı yorumları yapıyor ve olaylara aynı pencereden bakıyorsa çalışmalardan yeterli verim alınamayabilir. Takım bireylerinin farklı düşüncelere sahip olması ve gruptaki herkesin görüşüne saygı duyulması verimliliği artıracaktır.

Takımlarda bulunması gereken bu 7 özelliğin her biri takımın verimini, etkisini göstermekle birlikte o takımın ortaya koyacak işlerin başarılı olma olasılığını artırmaktadır.

2.3 KİŞİSEL PERFORMANS VE TAKIM PERFORMANSI

Kişisel yapılan yazılım projelerinde veya küçük yazılım projelerinde, kişisel performans gerçekten büyük önem taşır. Fakat proje boyutu büyüdükçe ve buna bağlı olarak projede çalışan insan sayısı çoğaldıkça takım performansının önemi ortaya çıkar. Kimse takım çalışması olmadan sadece kişisel performansı ile büyük projeleri başarısızlıktan kurtaramaz. Tabii ki kişisel performans yüksek olmadan takım performansı yüksek olamaz. Ama bu bir kişinin yüksek performansının bütün takım performansını artıracağı anlamına gelmez. Esas önemli olan her bir bireyin kişisel olarak maksimum performansa ulaşıp, takım içerisinde bir sinerji yaratmasıdır. “Neden yetenekli ve özel programcıların oluşturduğu takım, gerçek bir takım olamayabilir?”. Bunun nedeni bazı programcıların birlikte çalışamamaları veya uyum gösterememeleridir. Bu yeteneklerinden veya özel olmalarından kaynaklanmaz; sorun kişisel konulardandır. Takım içindeki insanların iletişimi güçlü olmadan, gerçek bir uyum sağlanamaz.

2.4 MOTİVASYON VE PERFORMANS İLİŞKİSİ

Her meslekte olduğu gibi yazılım sektöründe de performans, motivasyondan direk etkilenmektedir. Çalışanların motivasyonu ne kadar yüksek ise performansları da o kadar yüksek olur. Kişinin motivasyonu; psikolojik faktörlere, yeteneklerine ve iletişim kabiliyetine balıdır. Bu noktada yöneticilere büyük görev düşmektedir.

Çalışanların motivasyonunu yükseltmek için yöneticiler, çalışanlarının psikolojik yönlerini keşfetmeli, sorunlarına yardım etmeye çalışmalı, maddi ve manevi yönden onların yanlarında olduklarını hissettirmelidirler. Yöneticiler çalışanların yeteneklerini geliştirmek için onları eğitimlere göndermeli ve her yönden destek olduklarını göstermelidirler. Bu motivasyon yükseltici faaliyetler gerçekleştirilirken mutlaka olumlu bir iletişim kurulmalıdır.

Bu noktada yöneticiler şirket politikalarını belirlerken yüksek performans bekleyip çalışanlarını motive mi etmeliler, yoksa önce çalışanlarını motive edip daha sonra yüksek performans mı beklemeliler? Bence bu konuda şirketler her iki yöntemi aynı anda kullanmalı ve çalışanlarına uyguladığı politika şu şekilde olmalıdır: Önce çalışanını motive etmeli, daha sonra bu motivasyondan yüksek performans beklemeli, aldığı yüksek performansa karı da çalışanını tekrar motive edecek şekilde ödüllendirmelidir. Bu döngü içinde takımın ve şirketlerin daha başarılı olabileceğini düşünüyorum.

2.5 TAKIM ÇALIŞMASININ AMACI VE ÖNEMİ

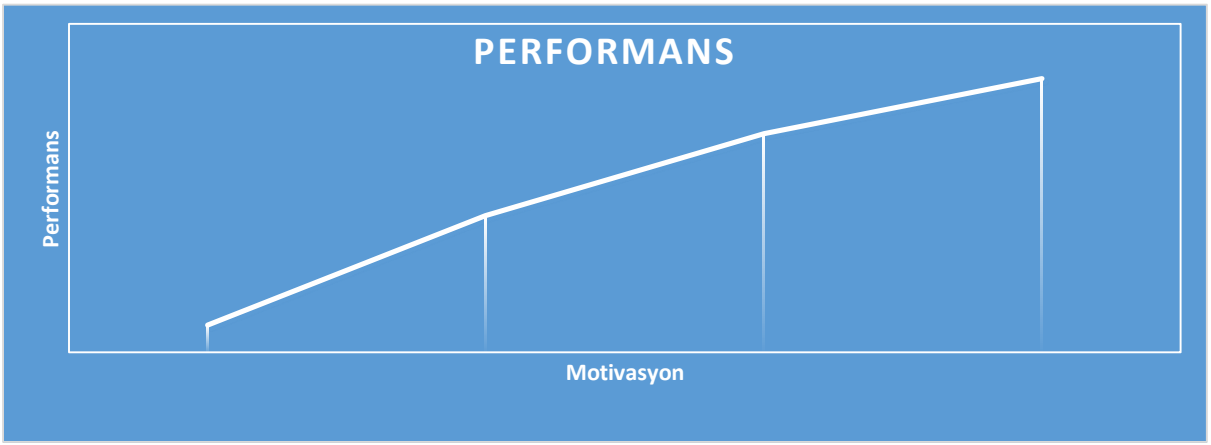
Takım çalışması, organizasyonlarda çalışanların ve yöneticilerin örgütsel hedeflerin tanımlanması, yöntem, süreçlerin belirlenmesi ve örgütlerin sürekli geliştirilmesi için bir arada çalışmaları şeklinde tanımlanabilir. Takım çalışması organizasyonlarda rekabet üstünlüğü kazanma düşüncesinin temelini oluşturmaktadır. Richter, Hemman ve Pohlandt (1999) yaptığı çalışmalarda takım içi işbirliğinde doğal öğrenme süreçlerinin, iletişimin şekline, iletişimin miktarına ve iletişimin içeriğine bağlı

olduğunu göstermiştir. Campion, Papper ve Higgs (1996) ise takım içindeki bağlılık düzeyinin tüm süreçleri etkilediğini öne sürmüştür. Günümüzün fazlasıyla dinamik çevre koşullarına ayak uydurabilmek için organizasyonlar örgütsel yönetim stratejilerinin ve yöntemlerinin güncellenmesine ihtiyaç duymaktadırlar. Bu değişim ve gelişim süreci organizasyonlarda yeniden yapılanmayı ve takım çalışmalarını zorunlu kılmaktadır. Bu nedenle takım faaliyetleri verimlilik ve performansı arttırarak stratejik amaçları gerçekleştirecek ve rekabette üstünlük yaratacak şekilde yeniden tasarımlamalıdır. Belirtilen bu sebeplerin yanı sıra örgütlerde takım anlayışına geçişin farklı nedenlerini aşağıdaki şekilde sıralamak mümkündür (Sarıhan, 1998):

- Dış çevredeki hızlı gelişmeler farklı alanlarda takım bilgi ve becerisini gerektirmesi
- Takım sinerjisinin takımı bireylerden daha güçlü yapması
- Örgütsel verimliliğin artması
- İş mükemmelliği anlayışına ulaşılması
- Bireylerin motivasyonlarının arttırılması
- Takım üyelerine birlikte ve bağımsız çalışma anlayışlarının kazandırılması
- Örgüt gerçeklerine uygun etkili fikirlerin üretilmesi
- İş tatmini ve örgütsel bağlılık duygularının gelişmesinin sağlanması
- Esnek ve yalın örgüt yapısı oluşturulması
- Çalışanların liderlik ve yaratıcılık özellikleri ile ilgili teşvik edilmeleri
- Problem çözme becerilerinin arttırılması
- Örgütsel amaçlara olan bağlılığın artırılması.

2.6 KARŞILIKLI BAĞLILIK VE TAKIM PERFORMANSI İLİŞKİSİ

Karşılıklı bağlılık kavramı belirli bir amacı gerçekleştirmek üzere birlikte çalışan takım üyeleri arasındaki etkileşimi ortaya koymaktadır. Karşılıklı bağlılık kavramının bir kolu görev bağlılığıdır. Görev bağlılığı, takımdaki her bir işi oluşturan farklı parçaların birbiri ile olan ilişkisini gösterir. Her bir faaliyetle ilgili görevli ekip üyeleri işlerini ne kadar etkin yürütürse diğer ekip üyeleri de kendi rollerini daha etkin yürütmesini sağlayacaktır. Yapılan araştırmalar görev bağlılığının grup motivasyonunu yükselttiğini, grup üyelerinin diğer üyelerin görevleri ve yürüttükleri işler üzerine de sorumluluk duymaya başlamasına bağlı olarak takım etkinliğinin yükseldiğini göstermektedir (Kiggundu, 1983).



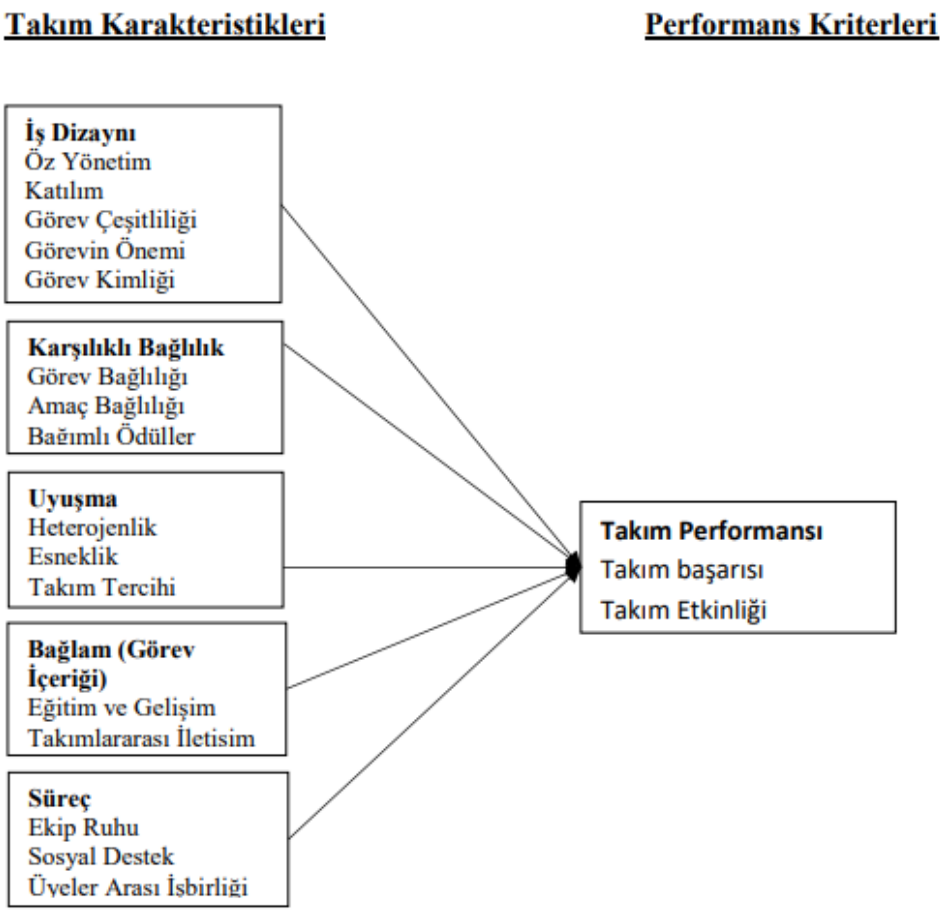
Yüksek dayanışma ve iletişim halinde olan takım üyeleri görevlerin tamamlanması ve hedef gerçekleştirme süreçlerinde diğer takım üyeleri ile çok daha yakın ilişki içinde çalışabilirler. Bunun yanı sıra kuvvetli arkadaşlık ilişkilerinin bulunduğu takımlarda karşılıklı ve sosyal etkileşimin de kuvvetlendiği bilinmektedir (Bowler ve Braas, 2006). Karşılıklı bağlılığın bir diğer şekli ise amaç bağlılığıdır. Amaç bağlılığı; ekip üyelerinin her birinin ve ekibin amaçlarının ve hedeflerinin ilişkili olmasıdır. Takımlarda amaçların üyelerinin ve hedeflerin somut ve açık olarak tanımlanmış olması grubun performansını ve etkinliğini doğrudan etkileyecektir. Araştırmalar gruplara tanımlanan hedeflerin mutlaka takım üyelerine tanımlanan somut hedeflerle desteklenmesi gerektiğini ortaya koymaktadır. Bunların yanı sıra bütünleşik geri dönüşler ve takım ödül sistemleri de karşılıklı bağlılığı ve takım performansını olumlu etkilemektedir. Bu nedenle bireysel hedef ve ödüller takım hedeflerini destekleyecek şekilde belirlenmelidir (Guzzo ve Shea, 1992).

2.7 UYUŞMA VE TAKIM PERFORMANS İLİŞKİSİ

Takım üyelerinin görevlerini ve içinde bulundukları takımı sevmelerinde ve performanslarının artmasında etkili olan faktörlerden birisi de diğer takım üyelerine ve çevresel faktörlere uyum sağlayabilmeleridir. Bir takım üyesinin diğer takım üyeleriyle kurduğu ilişkilerin güçlü olması birlikte çalıştığı kişilerle arasında yaşanabilecek potansiyel sorunları en aza indirgemektedir. Heterojenlik, etkin takımlardaki uyuşma karakteristiğinin önemli özelliklerinden biridir. Takım üyelerinin birbirine benzeşmesinden daha çok farklı yetenek, beceri ve tecrübelerinin olması etkinliğinin artmasına olanak tanıyacaktır, zira bu şekilde dizayn edilmiş gruplarda takım üyeleri birbirlerinden öğrenme fırsatına sahip olacaklardır (Gladstein, 1984). Bazı çalışmalar heterojenlik sayesinde takım üyelerinin memnuniyetinin, çatışma yönetimi becerilerinin ve iletişim yetkinliklerinin de artış gösterdiğini ortaya koymaktadır (Jackson, Brett, Sessa, Copper, Julin ve Peyronin; 1991). Etkin takımların bir diğer uyuşma karakteristiği de esnekliktir. Esneklik yeteneğine sahip takımlar, ihtiyaç anında birbirinin görev tanımını bilir ve yerine getirebilir. Bu da takımlarda etkinliği oldukça artırmaktadır (Sundstrom v.d, 1990).

2.8 SÜREÇ VE TAKIM PERFORMANSI İLİŞKİSİ

Süreç kavramı; takım üyelerinin birbirlerine karşı davranışları, bilgi paylaşımı, duygularını ifade edebilmesi, işbirliği grupları oluşturulması, takım liderinin desteklenmesi veya reddedilmesi ögelerini içerecek şekilde tanımlanır (Guzzo ve Shea, 1992). Süreçlerin belirlenmesinde takım koordinasyonu göz önüne alınmalıdır. Aynı görev için farklı üyelerin eş zamanlı çaba harcaması gereksiz kaynak tüketimi yaratacak ve takım ruhu ve enerjisini gereksiz tüketecektir. Bu nedenle iç uyumu yüksek takımlarda takım ruhu ve takım etkinliği yükselecektir. Süreç kavramı yaşayan bir olgudur. Bu nedenle zaman içinde süreç iyileştirmeleri yönünde ortaya koyulacak yaratıcı yaklaşımlar çözüme giden yolu, takım etkinliğini ve takım ruhunu olumlu etkileyecektir. Süreç kavramındaki önemli karakteristiklerden biri takımın görevi yapabilme potansiyeline duyduğu inançtır. Bu kavram karşımıza takım ruhu olarak da çıkmaktadır. Araştırmalar takım ruhuna sahip grup üyelerinin grup başarısı için çok daha fazla çalışıp çaba sarf ettiklerini göstermektedir (Guzzo, Yost, Campbell ve Shea, 1993). Süreçlere etki eden bir diğer özellik ise sosyal destek unsurudur. Birbirleriyle olumlu sosyal ilişkiler kuran üyelerden oluşan takımların birbirine daha çok yardım etme eğiliminde olduğu gözlenmektedir. Sosyal destek altında çalışan takım üyelerinin iletişim becerileri ve katılım düzeyleri yükselmekte iken stres düzeyleri azalmaktadır (Wolken ve Good, 1995). Süreçler tanımlanırken takım etkinliğinin arttırılabilmesi için iş yükü dağıtımının eşit ve adaletli yapılması gerekir. Aksi durumlar takım içinde huzursuzluklara ve bireyselleşme eğilimine sebep olacaktır. Paylaşımı arttırmak takım üyelerinin kendi performansları ile takım performansı arasında doğrudan ilişki olduğuna inanması üretkenliğe ve etkinliğe doğrudan etki edecektir (Sundstrom, vd. 1990).



2.9 YAZILIM GELİŞTİRME SÜREÇLERİ

Temel yazılım geliştirme aşamaları aşağıdaki gibidir:

1. **Planlama**: Yazılım yaşam döngüsünün ilk aşamasıdır. Temel ihtiyaçlar belirlenir, proje için **fizibilite** çalışmaları yapılır (maliyetlerin ve sistemin yararlarının tanımlanması) ve proje planlaması gerçekleştirilir.

2. **Analiz**: Bu aşamanın amacı sistemin işlevlerini ve kesin gereksinimleri açıklığa kavuşturmak ve sonucunda bunları belirli bir formatta **dokümante** etmektir. Bu çalışma müşteri, yazılım mühendisi, sistem analisti, iş analisti, ürün yöneticisi vb. rollerin bir araya geldiği gruplar tarafından yapılabilir. İhtiyaçların net olmadığı durumlarda yazılım mühendisi ve müşteri arasında iletişim ve birlikte çalışmanın çok daha fazla olması gerekir. Çeşitli yazılım geliştirme metodolojilerinde bu aşamada kullanım dokümanları ve test plan dokümanları da oluşturulabilir.

3. **Tasarım:** Gereksinimlerin tamamlanmasıyla beraber sistem tasarım aşamasına başlanır. Yazılım ürün tasarımı, müşterinin gereksinim ve isteklerini karşılamak üzere yazılım ürününün özellikleri, yetenekleri, ve arayüzlerinin belirlenmesi etkinliğidir. İki tür tasarımdan bahsetmek mümkündür

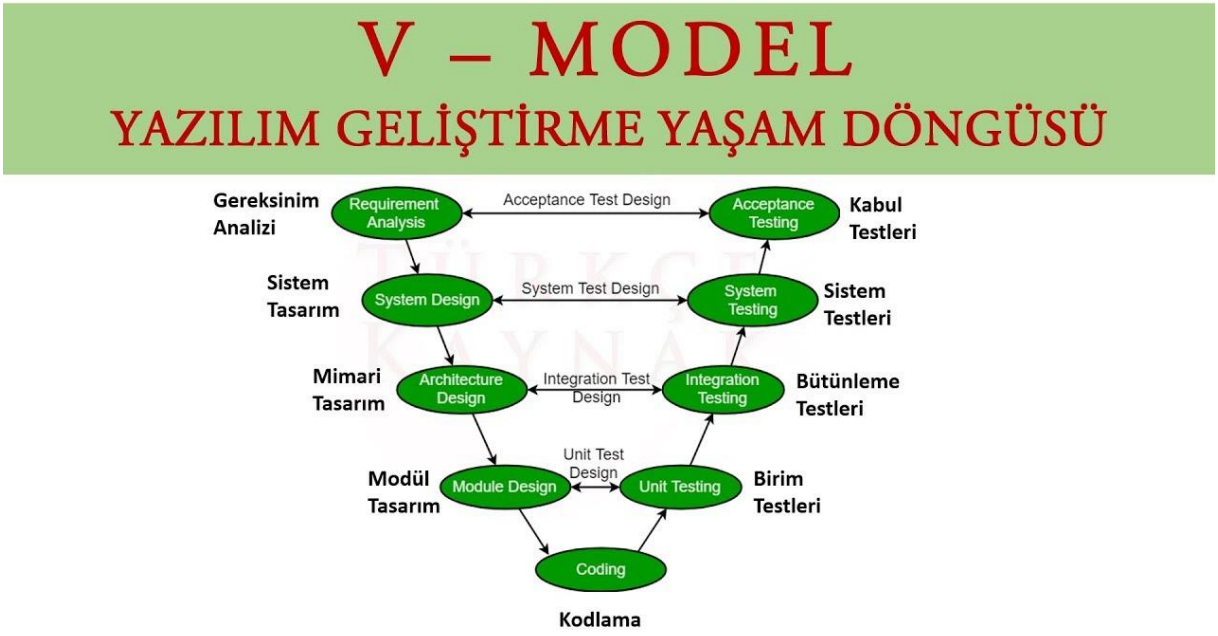
(Yüksek düzeyde tasarım — Mimari tasarım ve Detaylı tasarım). Mimari tasarım, yazılım modüllerinin genel yapıları ve organizasyon içerisindeki etkileşimleri ile ilgilenir. Sonucunda mimari tasarım dokümanları oluşturulur. Detaylı tasarım aşamasında Mimari tasarım dokümanları genelde revize edilirler. Tasarım ve analiz aşamalarının ayrımı "Problem **Ne**?/Problem **Nasıl** Çözülür?" sorularının kullanımı ile ilgilidir. Gereksinimlerin belirlendiği analiz aşaması problemin ne olduğu ile ilgilidir. Unutmamak gerekir ki sistemdeki tüm problemler yazılım ürününün tamamlanması ile çözülmeyecektir. Maalesef çoğu zaman *Ne* söylemi tasarım kararı olurken *Nasıl* söylemi de müşterinin gereksinimi olabilmektedir. Bu duruma dikkat etmek gerekir.

4. **Soyutlama**: Problemlerin çözümlerini kolaylaştırmak için nesnelerin, olayların ve durumların bazı özelliklerin görmezden gelinmesidir. Problemi basitleştirerek en önemli kısımlarına odaklanmamızı sağlar. **Modelleme** is temel tasarım aracı olup statik ve dinamik modellerden bahsetmekten mümkündür. Statik model, programın çalışması sırasında değişmeyen yönlerini ifade etmek için kullanılırken (sınıf ve nesne modelleri), dinamik model programın çalışması sırasındaki işleyişi ifade etmek için kullanılır (durum ve sıra diyagramları).

5. **Gerçekleştirim**
   5.1. **Kodlama**: Müşteriye teslim edilecek ürünü programlama aşamasıdır. Kaliteli kodlama üzerine önümüzdeki yazılarda bayağı kafa yoracağız. Şimdilik kısaca değinelim. İyi kod, okunabilirliği ve bakımı kolay olan **basit koddur**. **KISS** (Keep it simple) prensibine göre yeni mezun olmuş birisine kodunuzu verdiğinde 1–2 gün içerisinde anlayabiliyor ve değiştirebiliyorsa kodunuz iyi bir koddur. İster bir şirkette çalışın ister bireysel projeler geliştirin mutlaka belirli bir **kodlama kalite standardınız** olsun (İsimlendirme standartları, yorum satırı kullanımları, tekrar eden kodlar, dev –if koşul blokları, aşırı benzer işlevler, uzun metotlar vb.)
   5.2. **Test**: Erken test et yaklaşımı ile (**early testing**) hareket edip, analiz aşamasından itibaren test bakış açısına sahip olmamız hata yapma oranımızı ve maliyetleri (zaman, para, prestij vb.) düşürecektir. Birim testleri, duman testleri, yanlış değer testleri, kabul testleri, kullanım senaryo testleri, yük testleri, kullanıcı kabul testi, yoldan geçen adam testi, test otomasyonu gibi sürece ve duruma göre uygulanabilecek çok farklı kategoride ve derinlikte test türü bulunmaktadır.

6. **Teslim ve Bakım**: Tüm test aşamaları tamamlandıktan sonra yazılım ürünün sahaya teslim edilebilir bir versiyonu çıkartılır ve teslim aşaması gerçekleştirilir. Teslim çıktısı olarak ürün tek başına yeterli değildir. Mutlaka son kullanıcılar için kullanım kılavuzu ve versiyon fark dokümanı oluşturulmalıdır. Teslim ile birlikte bakım aşaması da başlar. Hata giderici, önleyici, altyapıyı iyileştirici, ürüne yeni özellikler ekletici gibi farklı bakım faaliyetleri mevcuttur.

Yazılımları geliştirmek için farklı modeller kullanırız. Genelde kullanılan modellerden biri V – Model'dir.



V – Model Yazılım Geliştirme Yaşam Döngüsü

2.10 YAZILIM GELİŞTİRME SÜREÇLERİNDE TAKIMIN YERİ

Öğrenci ve bireysel çalışmalar haricinde takım çalışmalarında genelde çevik yaklaşım kullanılmaktadır. Çevik yaklaşımda dezavantajlardan biri olarak, ürünün başarısı ile proje başarısı ortaktır ve bu sebeple kariyer riski doğurur. Bu, takımlar üzerinde hedefe ulaşmak için baskı oluşturur. Uç Programlama yaklaşımında ise yazılımcılar çiftler halinde çalışır, takım içi görev paylaşımı yoktur. Herkes tasarım, kodlama ve test aşamalarında görev alır.

2.11 SCRUM

Scrum, kelime olarak rugby oyununda oluşturulan küçük ekiplere verilen isimdir. Günümüzde çokça tercih edilen bir metottur. Eski yöntemlerdeki gibi projede izlenmesi gereken adımlar belirtilmemekte, bunun yerine basit ama önemli birkaç kuralı sayesinde esnek ama üretken bir işleyiş sunmaktadır. Scrum'ın sağladığı faydalardan biri de, işleyiş sürecini şeffaf hale getirerek aksaklıkları ortaya çıkarır. Böylelikle proje ekibinin hataları fark ederek sürekli olarak iyileştirme yapmalarına olanak sağlar.

Scrum'ın genel olarak çalışma mantığına gelecek olursak, yapılacak olan iş (Product Backlog) parçalara bölünür. Daha sonra işin belirli bir parçası (Sprint Backlog) ele alınır ve 2–4 hafta süreyle bu işin tamamlanması planlanır. Genelde 4 hafta olan bu süreye "Sprint" denir.

Klasik metodojilerde (Waterfall, …) kullandığımız sıralı yaklaşımda: (Sequential Approach) gereksinim, analiz, tasarım gerçekleştirme gibi aşamalar vardır. Bu metodolojiler bir aşama bitmeden diğerine geçişe izin vermez. Scrum'ın bize getirdiği overlapping (örtüşme) yaklaşımıyla ise tüm sprinte bakıldığında bu 4 aşama örtüşen bir şekilde beraber ilerler. Aşamalar bir bütünmüş gibi hareket edildiği için bir aşamada oyalanarak sıkılma olmaz ve sürekli bir ürün ortaya konulduğu için de müşteri bundan memnun olur.

Scrum' da 3 temel kavram vardır. Bunlar "**Roller**" , "**Toplantılar**" ve "**Araçlar**" olmaktadır.

Rollere gelirsek ilk olarak bir **Ürün Sahibi (Product Owner)** olmalıdır. Bu kişi illa müşterinin kendisi olmak zorunda değildir. Bir müşteri temsilcisi veya Scrum ekibinin bir parçası da olabilir. Önemli olan bu kişinin müşteriyle iletişim kurabilmesi, müşterinin ne istediğini bilmesidir. Bu kişi ekiple birlikte hareket eder ve projede bizzat yer alır.

Scrum Yöneticisi' nin (Scrum Master) görevi takıma emirler verip yönetmek değildir.

Scrum Master: takıma yardımcı olan, bir problem olduğunda onu çözen, takımlar ve ürün sahibiyle olan iletişime yardımcı olan, takımın ve organizasyonun Scrum'a adapte olmasını sağlayarak üretkenliğin artmasına katkıda bulunan kişidir.

Scrum Takımı (Scrum Team) , birbirleriyle iletişim halinde olan ve programlama işini yapan gruptur. Kendi içlerinde farklı görevleri olabilir ama hedefleri ortaktır. Genellikle 5 ila 10 kişiden oluşur.

Toplantılar:

Sprint Planning de sprintlerde neler yapılacağı, gereksinimlerin listesi, takımların belirlenmesi, risk analizleri ve maliyet hesaplamaları yapılır.

Tüm sprint boyunca her gün takım üyeleri bir araya gelerek **15 dakika** boyunca ayaküstü toplantılar yaparlar. Bu toplantılara "**Daily Scrum Meeting**" denir. Bu toplantılarda önceki gün neler yapıldığı, ne gibi sorunlarla karşılaşıldığı, o gün neler yapılacağı ve nasıl yapılacağı hakkında özet şeklinde konuşmalar yapılır. Ayaküstü toplantı denmesinin sebebi, toplantı denilince akılda şekillenen karşılıklı oturup slaytlar üzerine konuşmalar değil, herhangi bir grup üyesinin masasının başında bile olsa toplanarak takım içi iletişimi kaybetmemek ve birbirine yardımcı olmak için görüş alışverişi yapılmasıdır.

Her sprint ardından bir **Sprint Gözden Geçirme (Sprint Review)** yapılır. Buraya herkes davet edilebilir ve yapılanlar gözden geçirilir. Bu toplantılar uzun sürebilir. Ardından bir sonraki sprinte hazırlık yapılır.

Üçüncü temel değer olan Araçlar 'ın **(Artifacts)** başında **Ürün Gereksinim Dokümanı(Product Backlog)** gelir. Proje boyunca tamamlanması gereken işlerin listesidir. Listeyi **product owner** yönetir. Bu liste genelde kullanıcı hikayelerinden **(User Story)** oluşur. Yani kullanıcının isteğine göre yeni gereksinimler eklenir veya çıkarılır.

Sprint Listesi (Sprint Backlog), bir sprint turunda takımın yapması gereken işlerin listesidir. Product Backlog 'ın bir alt kümesi olarak düşünülebilir. Bir sprint döngü için seçilen gereksinimler, görevlere bölünerek takımda bulunan üyelere dağıtılır.

Sprint Kalan Zaman Grafiği (Burndown Chart), bir **sprint turu**nda görevler yerine getirildikçe yapılan iş ile kalan iş arasındaki bağıntıyı ortaya çıkaran grafiktir. Belirlen sürenin yeterli olup olamadığı, işin gidişatında bir sorun olup olamadığı ve işin zamanında bitip bitmeyeceği hakkında takıma ve yöneticilere bilgi sağlar.

Scrum, uzmanlık gerektiren ve maliyeti yüksek bir modeldir. Ancak kısa sürmesi, kolay uygulanması ve başarı garantisinin yüksek olması nedeniyle günümüzde kullanımı oldukça artmaktadır. Google, Microsoft, IBM ve Yahoo gibi büyük şirketler tarafından tercih edilmektedir.

Yazılım süreç modelleri birbirlerinden tamamen ayrı değillerdir ve çoğu zaman beraber kullanılmaktadırlar. Bir proje için hangi modellerin ve hangi adımların uygulanacağını "süreç mimarı" seçer. Bütün modellerdeki güçlü ve zayıf yanları kıyaslayarak projeye en uygun olanları seçer. Bu seçim yapılırken; projenin büyüklüğü, karmaşıklığı, şirketin yapısı, zaman ve maliyet gibi kriterler göz önüne alınır. Günümüzde en çok tercih edilen ve geçiş yapılmak istenen model, Agile Modellerdir. Kısa zamanda, esnek biçimde ve yüksek başarı sağlaması tercih sebeplerinin başında gelir.

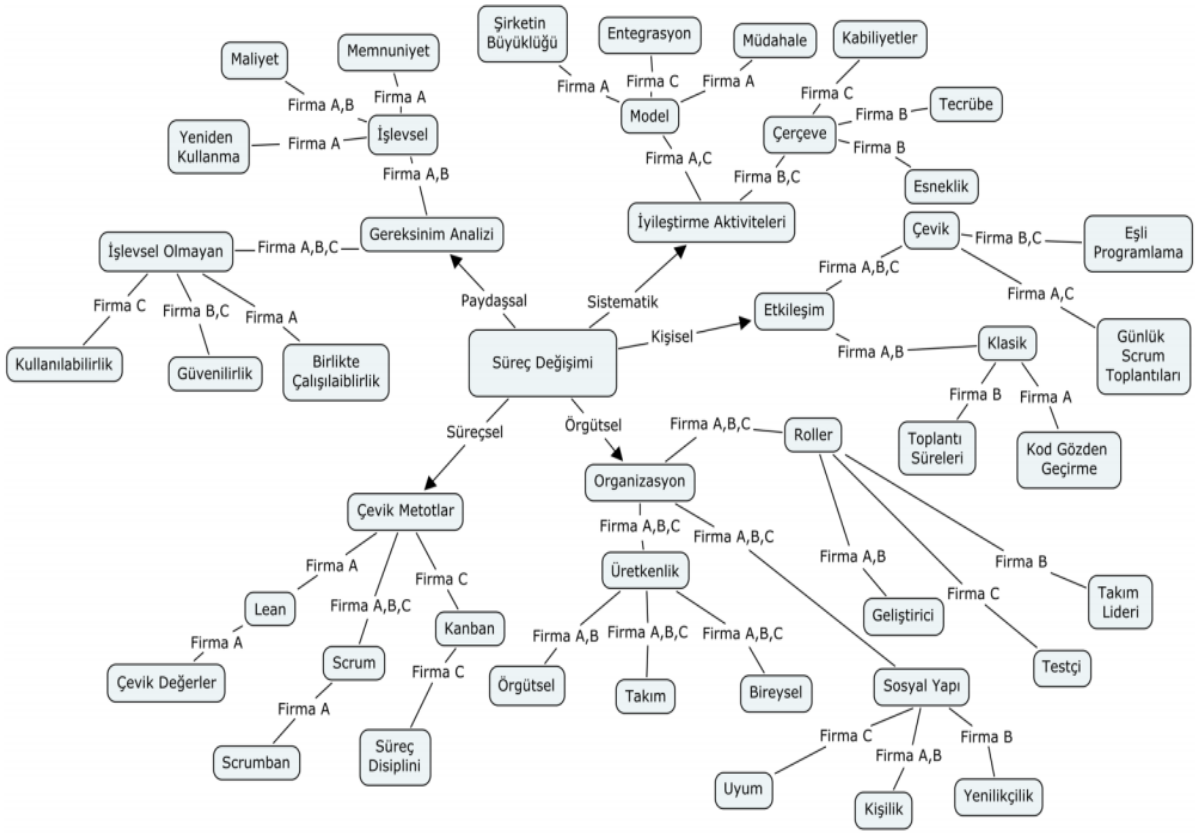
3. KARŞILAŞTIRMA

Geleneksel modeller ve yeni çevik yöntemler arasında takım çalışmasına yönelik birçok farklılık bulunmaktadır. Bu farklılıklar bazen çalışma şekilleriyle, bazen de takımdaki uyuma göre değişiklik göstermektedir.

Yazılım geliştirme süreç değişimi etkileyen faktörler, aşağıdaki tabloda belirtildiği gibidir:

Kategori	Tema	Kod
Organizasyon	Üretkenlik	• Bireysel • Takım • Örgütsel
	Sosyal Yapı	• Kişilik • Uyum • Yenilikçi
	Roller	• Geliştirici • Testçi • Lider
Gereksinim Analizi	İşlevsel	• Maliyet • Memnuniyet • Yeniden kullanma
	İşlevsel olmayan	• Kullanılabilirlik • Güvenilirlik • Birlikte çalışılabilirdik • Şirketin büyüklüğü • Entegrasyon
İyileştirme aktiviteleri	Model	• Müdahale
	Çerçeve	• Kabiliyetler • Tecrübe • Esneklik
Çevik Metotlar	Lean Scrum Kanban	• Scrumban uygulaması • Çeviklik Değerleri • Süreç Disiplini
Etkileşim	Çevik	• Eşli programlama • Günlük Scrum Toplantıları
	Klasik	• Kod gözden geçirme • Toplantı süreleri

Yazılım süreç değişimlerine yönelik görüşlerin, daha önceki araştırmalara dayanılarak firma bazlı düzenlenmiş kategori ve temaları aşağıdaki gibidir:



Firmaların kullandığı metodolojilere ve performanslarına bakıldığında, hacmi çok büyük olan firmaların çevik yöntemlere adapte olmasının daha zor olduğu gözlemlenmektedir. Ancak büyümekte olan firmalarda ise çevik yöntemler daha avantajlı olduğu tespit edilmektedir.

Yukarıda araştırılan firmaların içindeki bir şirket ilk yıllarında klasik plan-tabanlı yazılım geliştirme yöntemlerini benimsemiştir. Ancak, müşteri taleplerinin yıllar içinde değişmesi ve piyasa şartlarının hızlı ve teknoloji odaklı üretim istekleri şirketi çevik yöntemleri kullanmaya yöneltmiştir. Şirket, Türkiye'deki diğer birçok yazılım firması gibi Scrum metodolojisini kullanmayı tercih etmiştir. Bu durumun sebepleri şu şekilde sıralanabilir:

1. Üretim için 5-8 kişilik yazılım takımı ihtiyacı,
2. Müşteri temsilcisinin takımla sürekli bir arada bulunması gerekliliği,
3. 20 ile 26 günlük koşular üzerinde ürün parçalarının üretim gerekliliği.

Şirket özellikle bazı dönemlerde her bireyin sabit bir rol ile iş yapması yerine herkesin her işi yapabilmesi ilkesine uygun bir kültür geliştirmiştir. Üretim planlaması çevik yöntemler sayesinde dört veya altı aylık teslimat dönemlerinden on veya on iki haftalık dönemlere çekilerek müşteri memnuniyeti sağlanmıştır.

3.1 TAKIMLARDAKİ PÜRÜZLER

Takımlarda uyumsuzluk, anlaşamama, iletişimsel bozukluk gibi belli başlı pürüzler olabilmektedir. Bu gibi durumlarda yöneticilerin takımdaki rolü her zamankinden daha önemli olmaktadır.
Bazen bu durum disiplinlerarası işbirlik kurulamama gibi sorunlardan da oluşabilir.
Daha önce disiplinlerarası işbirliklerin iyileştirilmesi için yapılmış olan deneylere göre takım arkadaşları aşağıdaki faktörlere göre ayrılmaktadır:

Kültür: Bu faktör mesleki kültür, organizasyon kültürü ve bireysel kültürden oluşur.

Organizasyonel Karakteristikler: Bu faktör organizasyonun ve onu oluşturan öğelerin temel karakteristiklerini, pratiklerini ve özelliklerini ifade eden ögeler, özellikler ve/veya durumlardan oluşmaktadır.

Eğitim Deneyimleri: Çalışanların lisans veya lisansüstü eğitim sırasında disiplinlerarası bağlamda edindikleri teori ve pratikler ile ilgili ögeler, özellikler ve/veya durumlardan oluşan faktördür.

İletişim ve Takım Çalışması Yetkinlikleri: Çalışanların özellikle disiplinlerarası takım çalışması bağlamında önemli ve gerekli olan iletişim ve takım çalışması yetkinliklerinden oluşan faktördür.

Teşvik ve Ödüllendirme: Çalışanların özellikle disiplinlerarası takım çalışması bağlamındaki performanslarının teşvik edilmesi ve ödüllendirilmesi ile ilgili faktördür.

Faktörler Ögeler, Özellikler ve/veya Durumlar
Süreç Yönetimi
Farklı disiplinlerden kişilerin ortak çalışması gerektiği durumlarda ortak veya üzerinde uzlaşılmış terminoloji ve prensiplerin var olması
Organizasyonda politikaların, süreçlerin, planların veya ilgili diğer düzenlemelerin işbirliğini özendirecek şekilde kurgulanmış olması
Organizasyonda tanımlı olan iş süreçleri ve ilgili süreç varlıkları
Organizasyon veya proje düzleminde kullanılan kılavuzlar veya şablonlar
Temel kuralları göz ardı etmemeyi sağlayacak kontrol listelerin varlığı ve kullanılması
Süreçlerde farklı disiplinlerin görev, sorumluluk ve etkileşimlerinin iyi tanımlanmış olması
Süreçlere ve pratiklere ilişkin olarak bir organizasyon hafızasının oluşturulması ve bunların çalışanların istifadesine sunulması
Disiplinlerdeki kavramlar, terminoloji ve/veya prensipler arasındaki farklılıklar
Kültür
Mesleki kültür (çalışanların eğitimini aldıkları bölüm dolayısıyla edindikleri & sürdürdükleri kültür)
Organizasyon kültürü (organizasyonda her türlü tutum ve davranışın arkasında yer alan gerekçeler veya değerler)
Bireysel kültür (kişilerin ait oldukları statü/sınıf dolayısıyla oluşturdukları ve sürdürdükleri kültür)
Organizasyonel Karakteristikler 1
Projelerin gerçekleştirildiği organizasyonun organizasyon şeması
Görev alınan organizasyonun geçmiş tecrübeleri ve olgunluğu
Daha önce benzer projelerin ilgili süreçlerinde aktif rol almış olmak
Çalışanların sahip olduğu tecrübe ve birikimler
Gerekli olan fiziksel alan veya koşulların varlığı ve yeterliliği
İlgili ekibin daha önce işbirliği gerektiren proje veya işlerde birlikte çalışmış olmaları
Kişilerin süreçler veya faaliyetler ile ilgili almış oldukları teorik veya pratik eğitimler
Organizasyonel Karakteristikler 2
Yöneticilerin ve/veya liderlerin tutum ve davranışları
Özel işlerinin yapıldığı alanlar ile masa başı işlerin yapıldığı ofislerin ayrı tutulması
Kullanılan bilgisayar programları veya araçlar
Takım liderlerinin bilgi ve deneyimleri
Organizasyonun veya projenin işbirliği ile ilgili beklentileri
Ekibin motivasyonu, ilgisi ve isteği
Uygulanan proje yönetimi pratikleri
Eğitim Deneyimleri
Üniversite öğrenimi sırasında (lisans veya lisansüstü eğitim sürecinde) farklı disiplinlerden kişilerin bir araya gelerek disiplinler arası veya disiplinler ötesi projeler veya çalışmalar yapmış olmaları
Üniversite veya lisansüstü eğitim sırasında disiplinler arası çalışmayı gerektiren zorunlu derslerin var olması
Üniversite veya lisansüstü eğitim sürecinde gerçek hayatta karşılaşılan disiplinler arası veya ötesi çalışmalara benzer nitelikli tasarım çalışmalarının yaptırılması ve öğrencilerin gerçek hayata hazırlanması
İletişim ve Takım Çalışması Yetkinlikleri
Kişilerin iletişim yetenekleri
Kişilerin takım çalışmasına yatkınlıkları
Teşvik ve Ödüllendirme
Çalışmalarda gösterilen işbirliği performanslarının çalışanların performans değerlendirmelerinde veya terfilerinde hesaba katılması
İşbirliğinde başarılı olan personelin ödüllendirilmesi

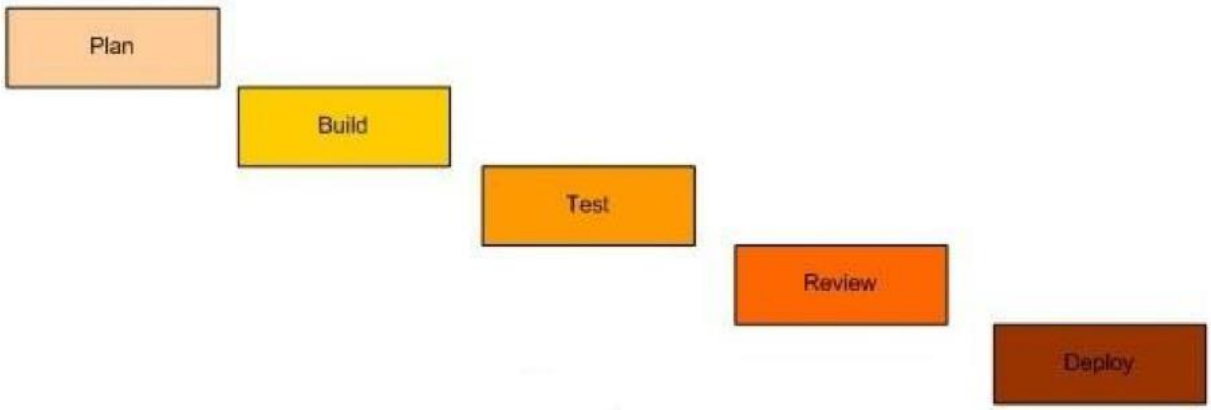
3.2 YAZILIM GELİŞTİRME METODOLOJİLERİNİN KARŞILAŞTIRILMASI

Yazılım geliştirirken en çok kullanılan genel metodolojiler aşağıdaki gibidir:

1. Waterfall (Şelale) Modeli
2. Iterative Waterfall (Yinelemeli Şelale) Modeli
3. Scrum Modeli
4. Lean (Yalın) Model

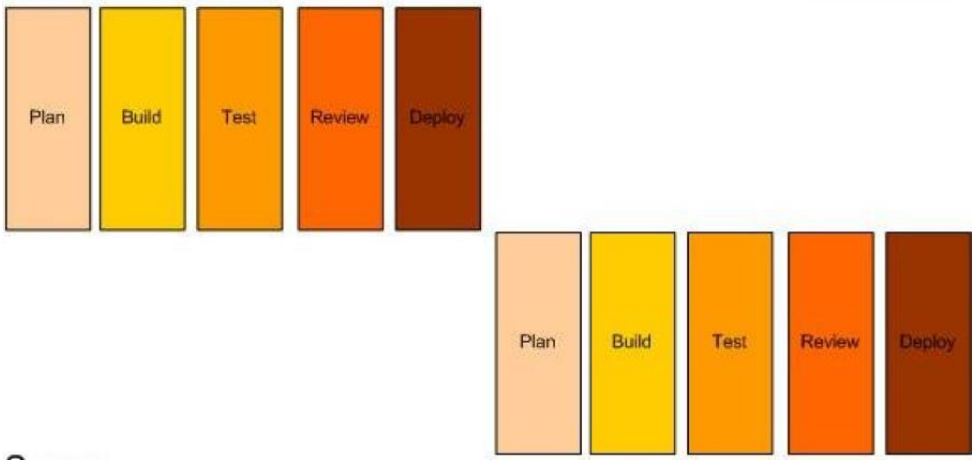
Bu metodolojilerin arasındaki farklar, şirketlerin kullanmak istediği, en işe yaradığını düşündüğü metodolojiyi seçme kararını verme aşamasında en belirleyici özelliklerdir. Bu farklar, takımlar arasındaki ilişkileri de değiştirirler.

Şelale modeli, klasik yazılım geliştirme yaklaşımı olarak da bilinmektedir. Bir faz bitmeden, diğer faza geçilmemesi bu metodolojinin en önemli kuralıdır. Önceki fazlar mükemmel olmadan diğer faza geçilemeyeceğinden maliyeti yüksek, verimi ise düşüktür. Önceki fazlarda değişiklik minimum olduğundan takımlarda anlaşmazlık ortaya çıkabilir.



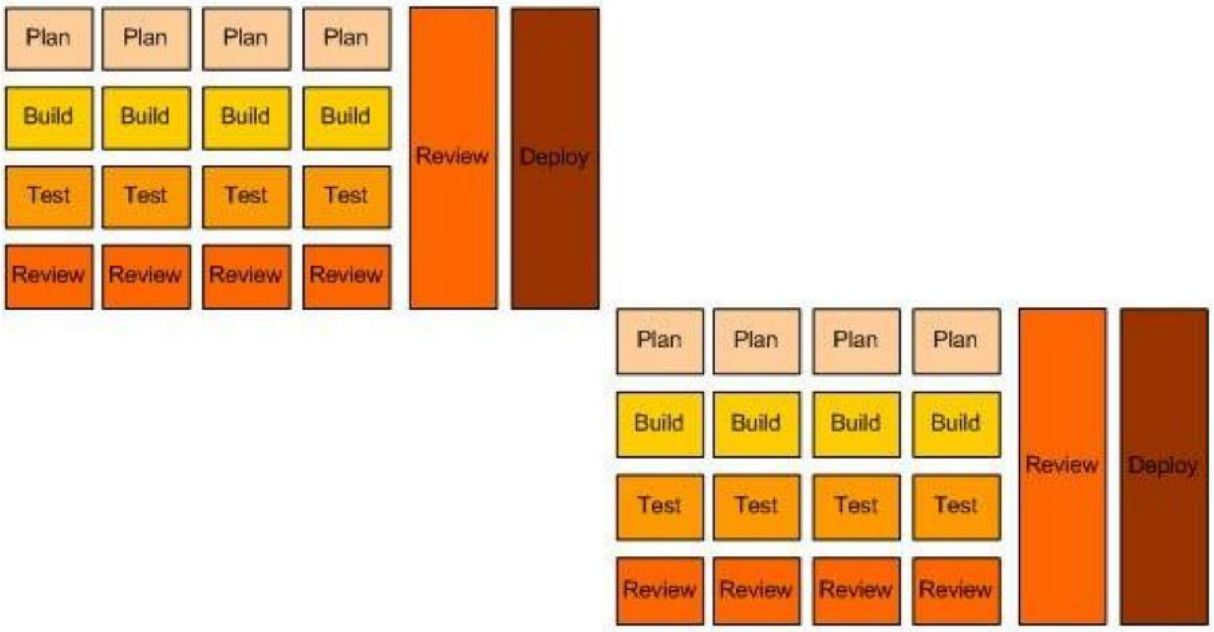
Şelale Modeli

Yinelemeli şelale modeli, şelale modeline göre daha az riskli olmasına rağmen çevik yöntemlere göre yine daha riskli ve verimliliği düşüktür. Herhangi bir fazda problem olduğunda genellikle ilk fazdan itibaren sorun kontrol edilir. Tüm takım hedefe uyma konusunda kalan işlere odaklanmalıdır. Ancak takımın hızı ve kalan işler konusunda bir bilgi yoktur. Sprintin sonuna kadar süreci izlemek konusunda herhangi bir yol yoktur.



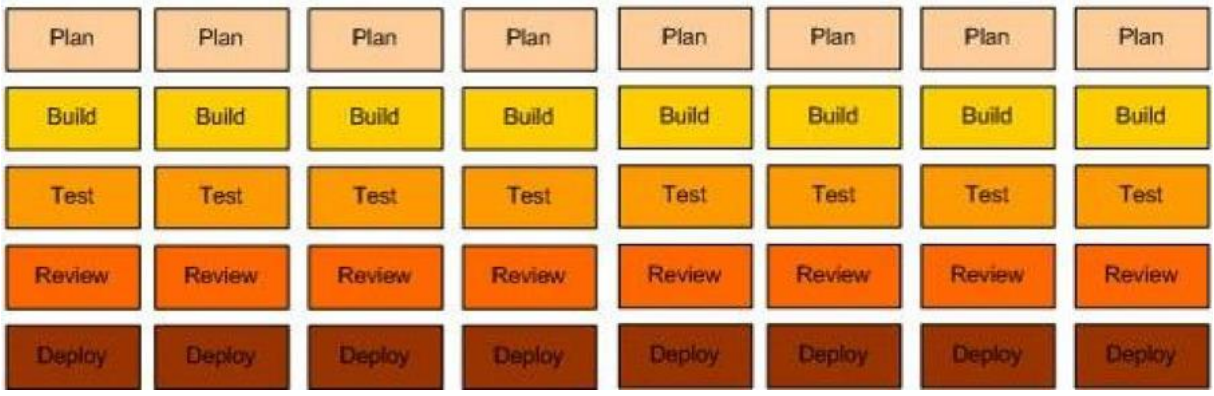
Yinelemeli Şelale Modeli

Scrum modeli ise şelale modeline göre çok daha az risklidir ancak maliyeti biraz daha yüksektir. Bu modelde takım tam olarak test edilmiş, bağımsız, değer oluşturan küçük özelliklere odaklanır. Böylece risk dağıtılarak, eğer bir özellik geliştirilirken sorun çıkarsa bunun diğer parçaları geliştirmeye engel oluşturma riski azaltılmış olur. Bu modelde yine sprint adı verilen iterasyonlar oluşturulur ve her iterasyonun sonucunda istenirse uygulama yayına alınabilir.



Scrum Modeli

Yalın metodolojisinde ise küçük parçalara bölünmüş özelliklere odaklanma konusunda Scrum modeline benzese de bu özelliği daha öteye taşımaktadır. Yalın modelde bir sonraki özelliğin planlama, geliştirme, test ve devreye alım süreçleri belirlenmeden önce yapılacak olan özelliğin planlama, geliştirme, test ve devreye alım aşamaları belirlenir. Böylece risk seviyesi özellik kırımına kadar inerek mümkün oldukça azaltılmış olur.
Yalın modelin temel yaklaşımı israfı engellemektir. Bu sebeple gerekli olmayan hiçbir özellik geliştirilmez.



Yalın Model

3.3 TAKIMLARIN ÖZELLİKLERİNE GÖRE DOĞRU METODOLOJİ SEÇİMİ

1. Waterfall (Şelale) Modeli

Şelale genellikle projenin kapsamını çok iyi anlayan büyük, plan güdümlü ekipler tarafından kullanılsa da, kendini yalnızca projeyeme veremeyen geliştirme ekipleri, daha modern yöntemlerin esnekliği ve çevikliği ile daha iyi sonuçlar bulacaktır.

2. FDD (Özelliğe Dayalı Geliştirme) Modeli

FDD, hali hazırda çalışan yazılımları sık sık sunma hedefine odaklanmakta ve özellikle müşteri odaklı bir yaklaşım olup, daha küçük geliştirme ekipleri için iyi bir seçimdir. FDD'nin sağlam yapısı, proje odaklı ve kırıcı onarım türlerini dengeleyen ekipler için daha az arzu edilir kılmaktadır.

3. Agile(Çevik) Model

Agile yaklaşımını kullanarak ekipler, her biri belirli bir süre ve teslim edilebilirler listesi içeren, ancak belirli bir sırada olmayan kısa sprintler veya iterasyonlar geliştirir. Sprintler sırasında, ekipler çalışan yazılım (veya başka bir somut, test edilebilir çıktı) sağlama hedefine doğru çalışırlar.

Agile, çeşitli departmanlardan ve müşterilerden gelen iç geri bildirimlerin yanı sıra ekip güçlerine ve verimliliğe odaklanan işbirliği ağırdır. Ekiplerin sürekli çalışma, test edilmiş, önceliklendirilmiş özellikler sunarak elde ettikleri Agile yaklaşımı ile müşteri memnuniyeti en yüksek önceliktir.

4. Scrum Modeli

Agile yaklaşımını uygulamanın başka bir yolu olan Scrum, Agile'nın temel inanç ve felsefelerinden ekiplerin ve geliştiricilerin yoğun ve günlük olarak işbirliği yapması gerektiğinden ödünç alır.

Scrum ile yazılım, ekibin ön ve merkez olduğu yinelemeli bir yaklaşım kullanılarak geliştirilir; daha küçük ekiplerdeki deneyimli ve disiplinli çalışanlar, kendi kendini örgütleme ve kendi kendini yönetmeyi gerektirdiği için bu yöntemle en fazla başarıyı bulabilir.

Ekip üyeleri başlangıçta nihai hedefleri daha küçük hedeflere ayırır ve yazılım oluşturmak ve sık sık (genellikle iki hafta süren) göstermek için sabit uzunluklu yinelemeler (sprintler) kullanarak çalışırlar. Toplantılar Scrum yaklaşımında önemli bir rol oynar ve her sprint sırasında ilerlemeyi takip etmek ve geri bildirim toplamak için günlük planlama toplantıları ve demolar yapılır. Bu artımlı yöntem hızlı değişiklikleri ve gelişimi teşvik eder ve karmaşık projelere değer katar. Scrum, daha geleneksel yazılım geliştirme metodolojilerinin yapısını ve disiplinini modern Agile'ın esnekliği ve yinelemeli uygulamalarıyla birleştirir.

5. XP (Ekstrem Programlama) Modeli

Genel olarak XP, basitlik de dahil olmak üzere adımlardan ziyade bir dizi değeri izler (gerekli olanı geliştirin, başka bir şey değil); iletişim (ekipler yazılımın her parçası üzerinde birlikte çalışmalı ve birlikte çalışmalıdır); tutarlı geribildirim; ve saygı.

Ekstrem Programlama, geliştiricilerin öncelikle müşterinin kullanıcı hikayelerini, belirli özelliklerin gayri resmi açıklamalarını planlamasını ve anlamasını gerektirir. Diğer uygulamalar şunları içerir: işlerin zamanlanmasını ve yinelemelere bölünmesi. Hatasız tasarım, kodlama ve sık sık test yapın, bu da hatasız yazılım oluşturmaya yardımcı olur. İşlevi en iyi anlamak için geri bildirimleri dinleyin ve ardından daha fazlasını test edin.

6. Lean (Yalın) Model

Yalın model, bir zamanlar üretim alanından ilke ve uygulamaları içeren ve bunları yazılım geliştirme de dahil olmak üzere çeşitli endüstrilere uygulayan bir iş akışı metodolojisi ve zihniyetidir. Agile, geliştirme en iyi uygulamalarının pratik uygulaması için mükemmel bir metodoloji olsa da, bu uygulamaların kuruluş genelinde ölçeklendirilmesi veya geliştirme türü çalışmaların dışında uygulanması için talimatlar içermez.

Bu nedenle takım düzeyinde Agile uygulayan birçok kuruluş, ölçekte yenilik yapmaya yardımcı olmak için Yalın felsefeleri, uygulamaları ve araçları dahil etmeye başlar. Lean'in temel ilkeleri - bütünü optimize etmek, israfı ortadan kaldırmak, kalite oluşturmak, bilgi oluşturmak, bağlılığı ertelemek, hızlı bir şekilde teslim etmek ve insanlara saygı göstermek - kurum genelinde karar verme sürecini, potansiyel sorunları ortaya çıkarmaya ve sürdürmeye yardımcı olacak şekilde yönlendirmeye yardımcı olabilir sağlıklı bir örgüt kültürü. Yalın düşünme ve Çevik yazılım geliştirme uygulamalarının en iyi şekilde birleştirilmesi, yalnızca geliştirme organizasyonuna değil, bir bütün olarak sisteme fayda sağlayan sağlıklı, sürdürülebilir bir inovasyon kültürü oluşturabilir.

4. SONUÇ

Yazılım çalışmalarında takım, ekip gibi topluluklardaki performansları etkileyen faktörler, yine o takımdaki roller olmaktadır. Takımdaki bazı kilit roller takımın motivasyonunu ve doğrudan performansı etkilemektedir. Takımdaki motivasyon eksikliklerini çözüme kavuşturup performansı en üst noktada tutmak elzemdir. Takım üyeleri arasında uyuşmazlık olmamasına özen gösterilmeli, takım üyeleri de ilişkileri iyi tutmalıdır. Takım yöneticileri, takımla ilgili detayları bilmeli ve sürekli iletişim halinde olmalıdır. Yöneticiler haricinde takım üyeleri de aynı amaç için çalıştıklarını unutmamalı, yöneticilerine yardım etmelidirler. Genel olarak kullanılan metodolojiler, projeyi daha uyumlu, hızlı, verimli ve daha az maliyetli yapmaya çalışsa da takımlar bu metodolojilerde direkt olarak etkilenmektedirler. Elbette ki farklı metodolojilerde çalışan takımlar farklı etkileneceklerdir. Yine elbette ki bazı takımlar bu metodolojilere uyum sorunları yaşayabilmektedirler. Ancak genel olarak bakıldığında çevik yaklaşımlar, takımlar için çok daha iyi olduğu kanısı mevcuttur. Takımların yazılım geliştirme süreçlerindeki etkisi, metodolojilere bağlı olarak değişse de takım içerisindeki uyum, yazılım geliştirme sürecini doğrudan etkilemektedir. Uyumsuz takımlar, diğer takımlara göre yenik başlarlar. Tabii ki bu takımlar zaman içinde uyum gösterebilmektedir ancak bu geçen zaman projeye maliyet olarak yansıyacaktır. Minimum maliyet için takımların belli teorik ve pratik bilgilerinin üst düzey olmasına, takım uyumu sergilemesine dikkat edilmesi elzemdir. Geçmişte araştırmalara konu olan projelerde gerek kullanılan metodolojiler olsun, gerek takım içerisindeki uyum olsun, gerekse proje yöneticilerinin çalışmaları olsun bazı problemler doğmuştur. Bu problemleri oluşturan temel etkenler aşağıdaki gibi sıralanabilir:

- Doğru metodolojiyi seçememek
- Takımdaki teorik ve pratik bilginin azlığı
- Takım içerisindeki bilgi donanımının farkı
- Takım içerisindeki uyumsuzluk
- Proje yöneticilerinin takımı tanımaması
- Proje yöneticilerinin takımla yeterince ilgilenmemesi
- Proje yöneticilerinin projeyi tamamen anlamaması

Bu temel etkenlerden herhangi biri bir projede mevcutsa o projede bir problem çıkması çok muhtemeldir. Hiçbir projede problem istenmeyip, her problem maddi ve manevi olarak projede kayıp olarak görülmektedir.
Eğer proje içindeki takım ve proje yöneticileri bu temel etkenlerin üstesinden kolaylıkla gelebiliyorsa, üzerinde çalışılan projenin başarıya çok yakın olduğu söylenebilir.

5. KAYNAKÇA

[1] Pusula360 Yazılım Teknolojileri ve Danışmanlık, Başarılı Takım Çalışmasının 7 Özelliği, https://www.pusula360.com/blog/detay/22/basarili-takim-calismasinin-7-ozelligi

[2] Mehmet İnce, Aykut Bedük, Enver Aydoğan, Örgütlerde Takım Çalışmasına Yönelik Etkin Liderlik Nitelikleri, Selçuk Üniversitesi ,2004

[3] Ercan Ergün, Mustafa Emre Eyisoy, Takım Çalışması Özelliklerinin Takım Performansına Etkisi Üzerine Bir Araştırma, Business & Management Studies: An International Journal, Aralık 2018

[4] Alphabet, Alphabet Announces Fourth Quarter and Fiscal Year 2019 Results, https://abc.xyz/investor/static/pdf/2019Q4_alphabet_earnings_release.pdf, Aralık 2019

[5] Scruminc, Is Your Company Agile?, https://www.scruminc.com/company-agile/, Kasım 2014

[6] Tunca SELBES, Yazılım Takımlarında Başarı, Meteksan Sistem, Simülasyon ve Görsel Sistemler, ODTÜ

[7] Richter, P., Hemman, E., & Pohlandt, A, Objective task analysis and the prediction of mental workload: Results of the application of an action-oriented software tool(REBA). New approaches for modern problems in work psychology, 1999

[8] Ahmet Ilhan, Ercan Ince, Takım Çalışması ve Takım Etkinliğini Belirleyen Faktörlerin Ölçülmesi, Gaziantep Üniversitesi, Ocak 2015

[9] Murat Yılmaz, Üç yazılım firmasında yazılım süreç değişimlerinin gözlenen etkileri: Endüstriyel keşif vaka çalışması, Bilgisayar Mühendisliği Bölümü, Mühendislik Fakültesi, Çankaya Üniversitesi, Ankara, Türkiye, Ocak 2018

[10] Mustafa Değerli, Pınar Kaygan, Sistem Projelerinde Elektrik & Elektronik Mühendisleri ve Bilgisayar Mühendislerinin İşbirliklerinin İyileştirilmesi, TÜBİTAK – İleri Teknolojiler Araştırma Enstitüsü & ODTÜ – Enformatik Enstitüsü Ankara, Türkiye, 2016

[11] PEM360, Waterfall, Iterative Waterfall, Scrum ve Lean Yazılım Geliştirme Metodolojileri Arasındaki Farklılıklar, https://www.pem360.com/blog/Agile/Waterfall-Iterative-Waterfall-Scrum-ve-Lean-Yazilim-Gelistirme-Metodolojileri-Arasindaki-Farkliliklar/167

[12] PEM360, Agile ve Waterfall Yaklaşımları Karşılaştırılması, https://www.pem360.com/blog/Agile/Agile-ve-Waterfall-Yaklasimlari-Karsilastirilmasi/179

[13] PlanView, Top 6 Software Development Methodologies, https://blog.planview.com/top-6-software-development-methodologies/