# Yazılım Gereksinimleri Mühendisliği Niçin Önemlidir?

# Yazılım Projelerinde Başarı Faktörleri
# Chaos Raporları

- Düzenli olarak yayınlanan ve yapılan araştırmalar IT (BT) projelerinin başarı/başarısızlık raporları Standish Group tarafından hazırlanır.
- Chaos raporu, IT projelerini uluslararası düzeyde değerlendirir.
- 2020'de ABD'deki zayıf yazılım kalitesinden kaynaklı başarısız projeler 260 milyar dolarlık kayıp gerçekleştirmiştir.

# Standish Group CHAOS Raporları

Yazılım projelerinin
%19'u tamamlanmadan iptal edilmekte
%47'sine uygulama esnasında itiraz edilerek değişikliğe uğramaktadır.

Projeleri başarıya götüren üç faktör;

- ❑iyi bir sponsor,
- ❑iyi bir takım ve
- ❑iyi bir çalışma ortamı

olarak belirlenir.

# Başarısızlık Nedenleri

Pek çok projen teknik olmayan nedenlerden dolayı başarısız olmaktadır. belirtilmektedir.

Sorunun çözümü için yıllardır çalışılmaktadır.

Gerçekleştirilen tüm gayretlere rağmen, yazılım projelerinin  sadece üçte biri tamamen başarılı olarak tamamlanabilmekte ve ilerleyebilmektedir.

# Chaos Raporu İstatistikleri

## Proje Başarı Faktörleri:

Paydaş katılımı %15,9

Yönetim desteği %13,9

Açık/net gereksinimler %13

Doğru planlama %9,6

Gerçekçi beklentiler %8,2

# Chaos Raporu İstatistikleri

## Proje Zorluk Faktörleri (projenin kusurlu tamamlanmasına sebep olan faktörler)

Kullanıcı katkısının eksikliği %12,8

Tamamlanmamış gereksinimler %12,3

Gereksinimlerdeki değişiklikler %11,8

Yönetim desteğinin eksikliği %7,5

Teknoloji yetersizliği %7

# Chaos Raporu İstatistikleri

## Proje Başarısızlık Faktörleri

Eksik gereksinimler %13,1

Kullanıcı katkısının eksikliği %12,4

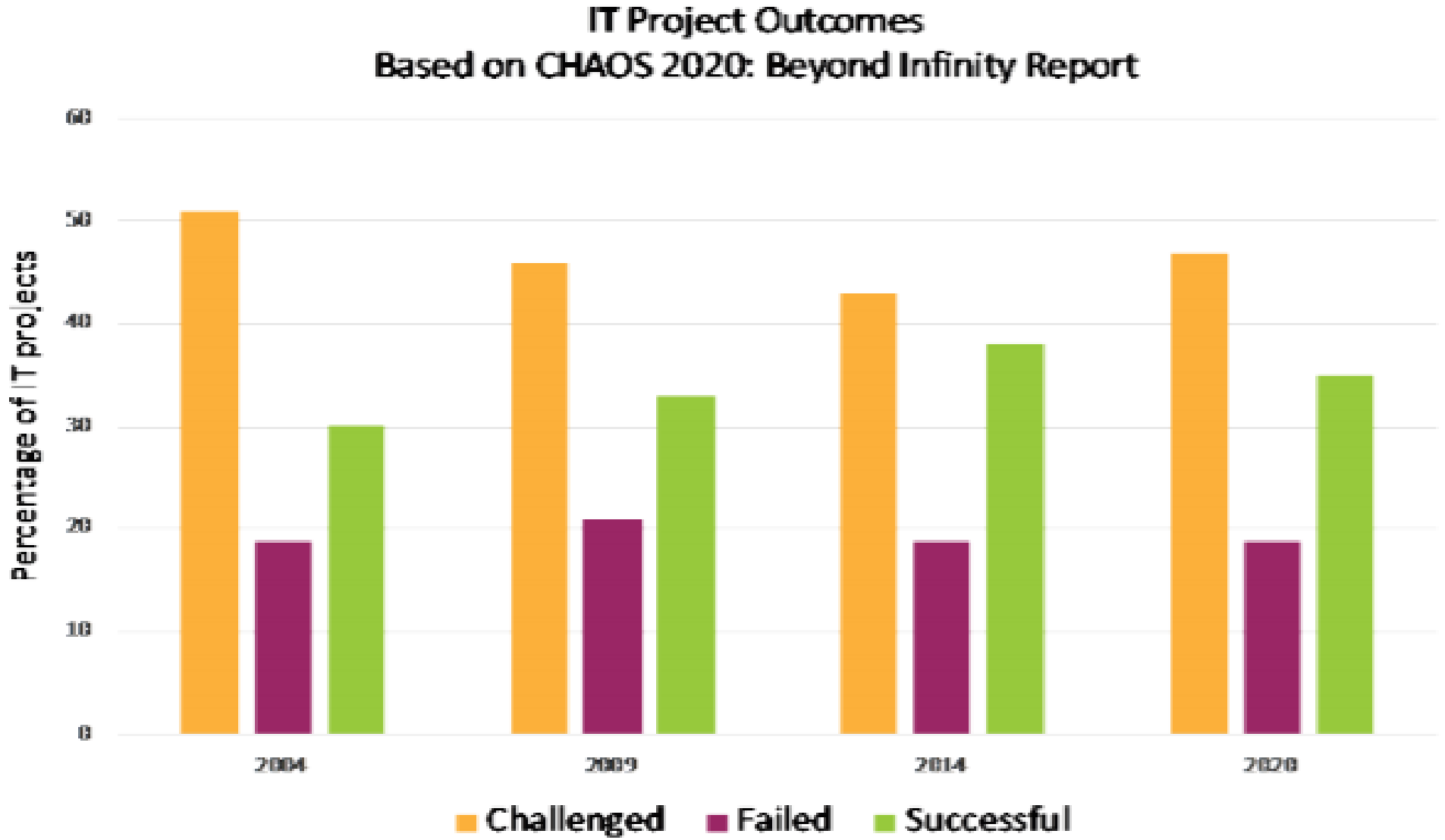
Kaynakların eksikliği %10,6

Gerçekçi olmayan beklentiler %9,9

Yönetim desteğinin eksikliği %9,3

Projelerde maliyet aşımlarının %150 ila %200, zaman aşımlarının da %200 ila %300 arasında olduğu tespit edilmiştir.

https://hennyportman.wordpress.com/2021/01/06/review-standish-group-chaos-2020-beyond-infinity/


IT Project Outcomes
Based on CHAOS 2020: Beyond Infinity Report
Percentage of IT projects
60
50
40
30
20
10
0
2004
2009
2014
2020
Challenged
Failed
Successful

# İş Analistliği ve Gereksinim Mühendisliği

- İş analisti, müşterinin sesi olarak prototipleme yöntemlerini kullanır, gereksinimleri tespit eder, modelleme ve analiz etme çalışmalarını yürütür.
- İş analisti, iş(business) ve geliştirme (development) tarafındaki paydaşlar arasında bir köprü oluşturur ve beklentinin doğru karşılanmasını sağlamaya çalışır.
- Uluslararası enstitüler tarafından iş analistliği programları açılarak sertifikalar verilmektedir.
  - PMI-PBA (Project Management Institute Professional in Business Analysis) Proje Yönetim Enstitüsü tarafından verilen iş analistliği sertifikasıdır.
  - IIBA-International Institute of Business Analysis tarafından da CBAP (Certified Business Analysis Professional) ve CCBA (Certification of Capability in Business Analysis) olarak iki seviyede iş analistliği sertifikaları verilmektedir.

# Çevik Manifesto ve  Gereksinimler Mühendisliği

Çevik manifesto 2001 yılında yayınlanmıştır.

Küçük iterasyonlarla işlerin tamamlanması  değer yaratmaya başlamıştır.

Kullanıcının  üretilmiş ara ürünü kullanması, geri bildirimlerle bir sonraki iterasyonları planlayarak artırımlı ilerleme sağlamıştır.

Müşteri ile işbirliği ön plandadır.

Geliştirme ekibi, otonom bir ekiptir, kendi kendisini yönetebilen ve organize olabilen bir ekiptir.

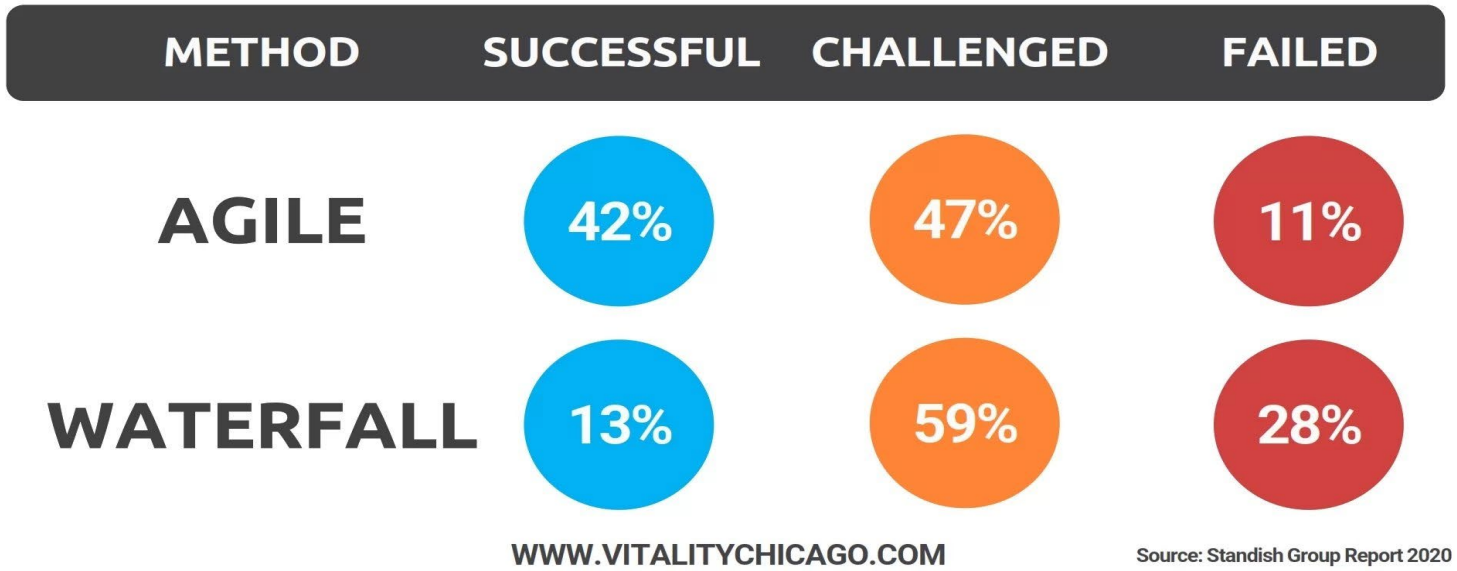
İterasyonlar çerçevesinde tamamlanacak olan işin taahütü yapılır ve ilgili zamanda işlerin tamamlanmasını sağlayacak şekilde takım olarak birarada işbirliği içinde çalışmalar yürütülür.

Çevik yaşam döngüsü, Ar-Ge projelerinde, gereksinimlerin çok sık değişme ihtimalinin bulunduğu projelerde, yazılım projlerinde oldukça ideal bir yaklaşımdır.

Çevik ve klasik yaşam döngüleri ile yönetilmiş projelerin başarı ve başarısızlık istatistiklerini de Chaos raporunda görülmektedir.

# PROJECT SUCCESS RATES

# AGILE VS WATERFALL




WWW.VITALITYCHICAGO.COM

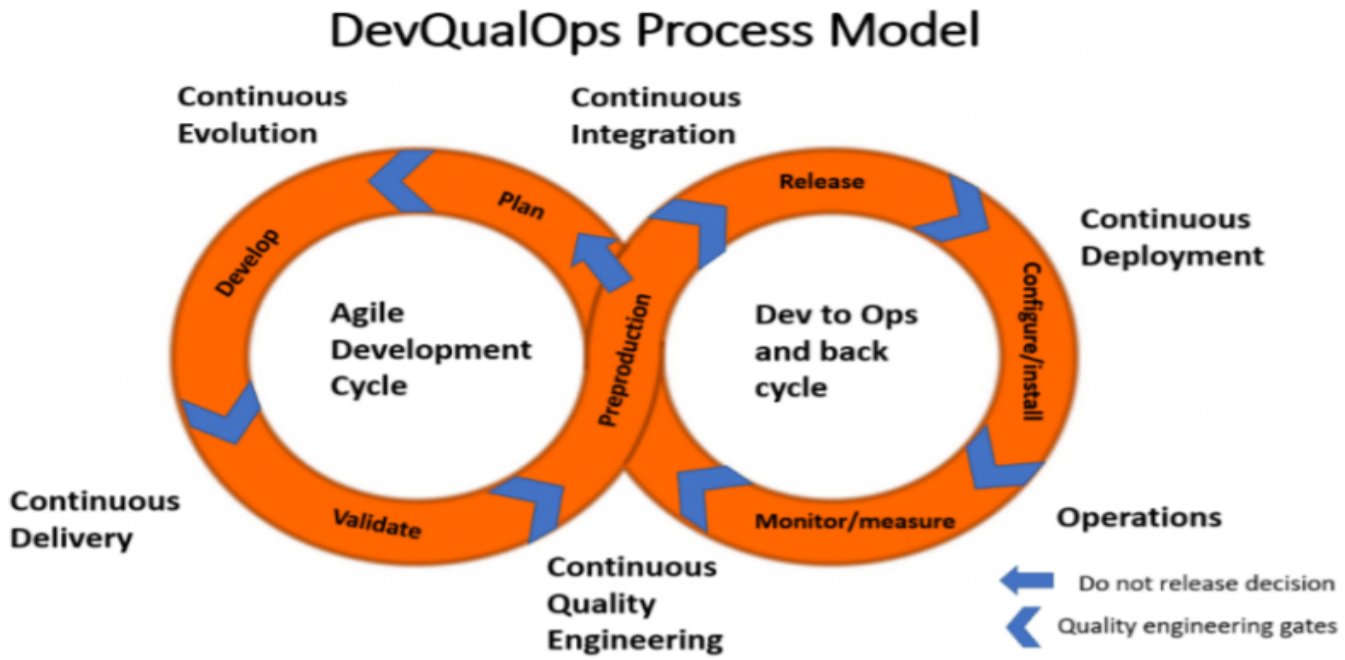
Source: Standish Group Report 2020

# The Role of the Project Manager on Modern Agile Projects

https://agilealliance.org/resources/sessions/the-role-of-the-project-manager-on-modern-agile-projects/

Alistair Cockburn -co-author of the Agile Manifesto

*The New DevQualOps Process Model*



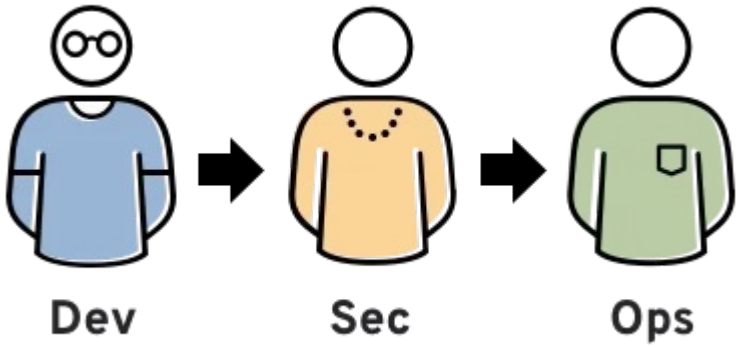

❑ Modern (günümüz) yazılım projelerinde bir diğer yaklaşım da “DevOps” tur.
“The Cost of Poor Software Quality in the US: A 2020 Report ( Düşük Yazılım Kalitesinin Maliyeti) raporu incelendiğinde ; çevik ve Devops yaklaşımlarının birbirini tamamladığı görülmektedir.

❑ Raporda sürekli değişim, entegrasyon ve devreye alma mantığının yazılım süreçlerindeki gereklilikler olarak özetleniyor.

https://www.it-cisq.org/cisq-files/pdf/CPSQ-2020-report.pdf

# DevOps ve Gereksinimlere Etkisi

- Bir çok  IT firmasında küçük, artırımlı değişiklikler sürekli test edilmekte, günlük, saatlik hatta anlık olarak sistemlerde canlıya geçişin sağlandığı görülmektedir.
  - Bu yaklaşım “DevQualOps” olarak adlandırılır.
- Agile+DevOps yaşam döngüsü boyunca uygun bir kalite düzeyi sağlanmasını hedeflenir.
- DevSecOps, güvenlikte kalitenin bir alt faktörüdür.
  - DevQualOps’un bir alt modelidir.


Dev
Sec
Ops

# DevQualOps modeli: Temel özellikleri

- Ölçülebilir olan tanımlanmış kalite hedefleri
- Yazılım kalite mühendisliğinde (Software Quality Engineering) ekibinetkisinin daha etkin ve stratejik olması
- Kalite sdüzeylerinin  ölçümü ve trend analizleri
- Sürekli süreç iyileştirmenin bir parçası olarak kusurların  önlenmesi
- Karmaşıklığı  düşürmek üzere  planlı yeniden düzenlemeler
- Her yinelemede planlanan hata bulma ve düzeltmeler (bug-fix)
- Sürecin kilit noktalarında kalite kapıları (quality gate)
- Sistemin tüm bileşenlerinin, özellikle açık kaynaklı yazılımların,  statik ve dinamik analizi

# Başla-Tanımla- Ölç -Yönet Yaklaşımı

- DMAIC Yaklaşımı: Define (Tanımla), Measure (Ölç), Analyze (Analiz Et), Improve (İyileştir), Control (Kontrol Et) aşamalarını içeren yapılandırılmış bir yaklaşımdır.
- Bu yaklaşımla şirketin kalite ve başlangıçlara yönelik "her şey yolunda" düşüncesinde olmayacaktır.
  - Kaliteyi ölçme yeteneği, kaliteli yatırımların işletmeye ve müşterilerine ne kadar iyi fayda sağladığını ölçmenin sonuçtaki testidir.
- Geliştirilecek projeler kaliteden ödün vermeden
- müşteri beklentileri ile uyumlu,
- olası değişikliklere çabuk tepki verebilen,
- adaptasyonu yüksek,
- geri bildirimlerle beslenen
- paydaş işbirliğinin ön planda tutulduğu

yaklaşım modellerini uygulayabilir.

  - ✓ Bu başarılı yazılım projelerine ilişkin önemli bir anahtardır.

# Yazılım Projelerinde Önceliklendirme Yöntemi: MOSCOW Tekniği

**M**O**SC**O**W** tekniği?

o harfi tamamen kolay okunması için konulmuş tekniktir.

4 farklı kategoriden oluşur.

Proje için

hangi özellikler olmazsa olmaz,

hangi özellikler kritik olmayan ama önemli olan,

hangi özellikler memnuniyet verici

hangi özelliklerin olmasa da olur şeklinde bu 4 farklı kategori oluşturur:

Must Have

Should Have

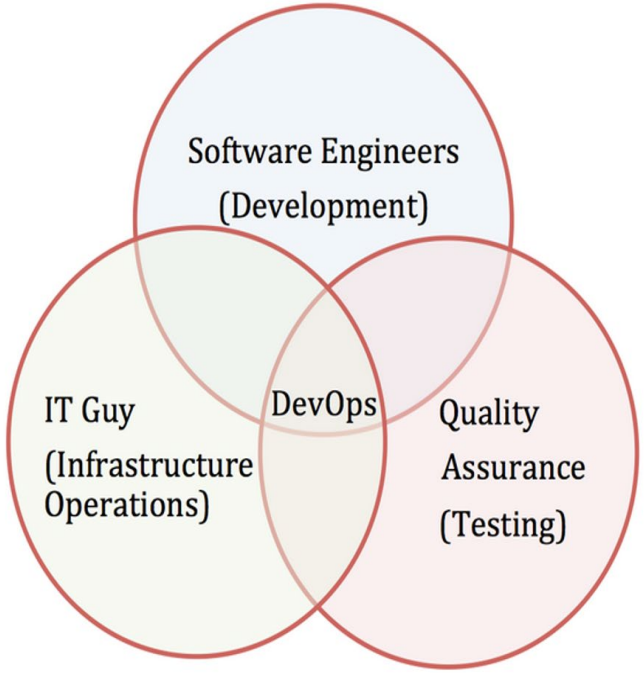
Could Have

Won’t Have

# Proje Yönetiminde OSCAR Modeli

# DevOps nedir? Ne değildir?

- ❑ DevOps is not simply combining Development & Operations teams
- ❑ DevOps is not a separate team
- ❑ DevOps is not a tool
- ❑ DevOps is not a one-size-fits-all strategy
- ❑ DevOps is not just automation

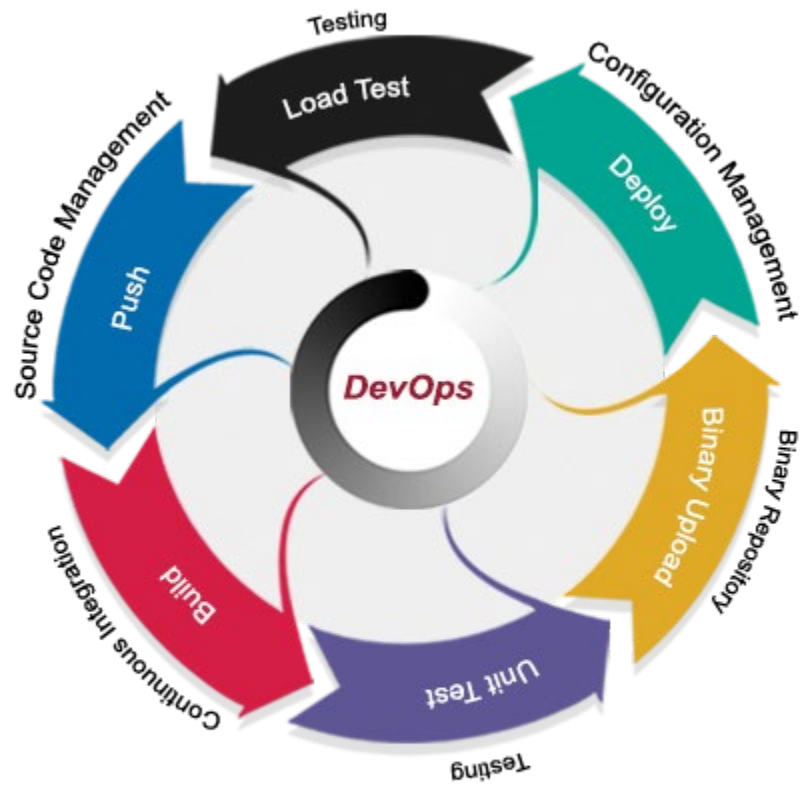


# DevOpsTanımlamaları


- DevOps is the philosophy of unifying Development and Operations at the culture, practice, and tool levels, to achieve accelerated and more frequent deployment of changes to Production.
  - Culture=behaviour, teamwork, responsibility/accountability, trust...
  - Practice=policy, roles, processes/procedures, metrics/reporting...
  - Tools=shared skills, toolmaking for each other, common technology platforms...

# DEVOPS
## LIFE CYCLE

- ✓ Push Code
- ✓ Fetch Changes
- ✓ Run Unit Tests
- ✓ Build Artifacts
- ✓ Store Artifacts
- ✓ Provision environment
- ✓ Deploy Your Build
- ✓ Run Load & Functional Tests
- ✓ Dev -> QA -> Staging -> Production



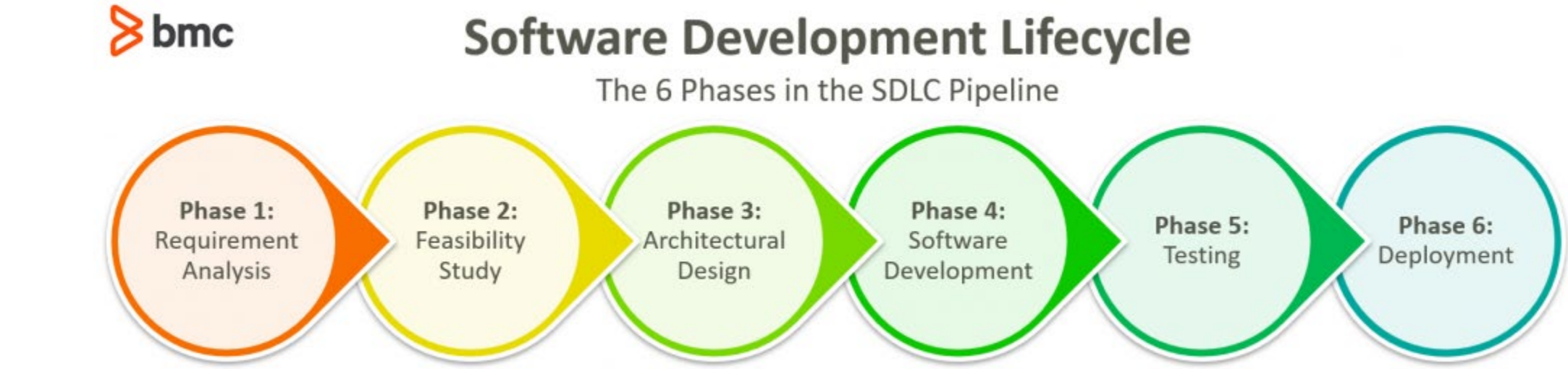

**DevOps - Spanning across entire delivery pipeline**

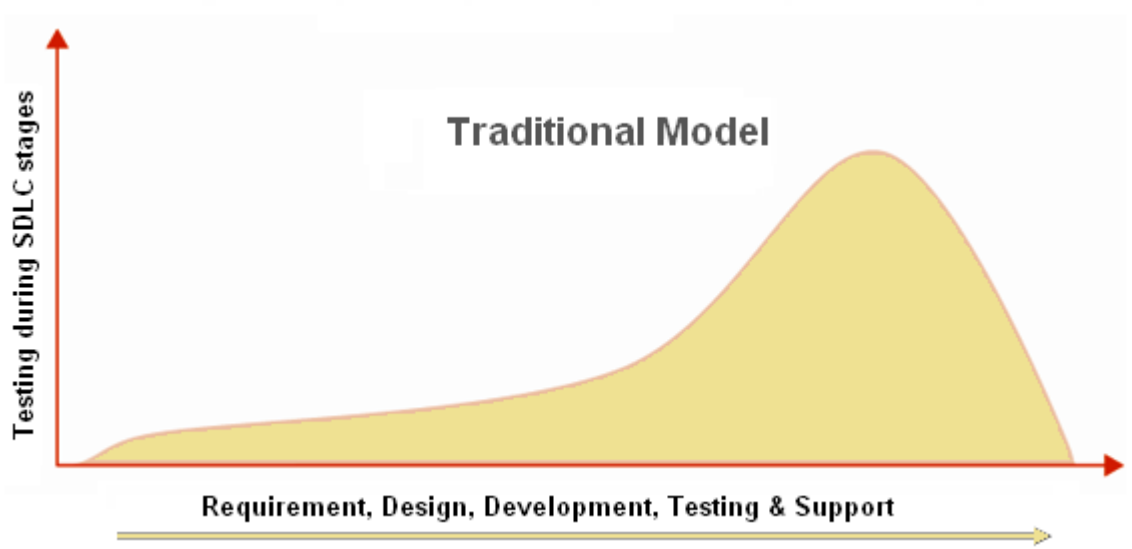
Continuous Integration | Continuous Delivery

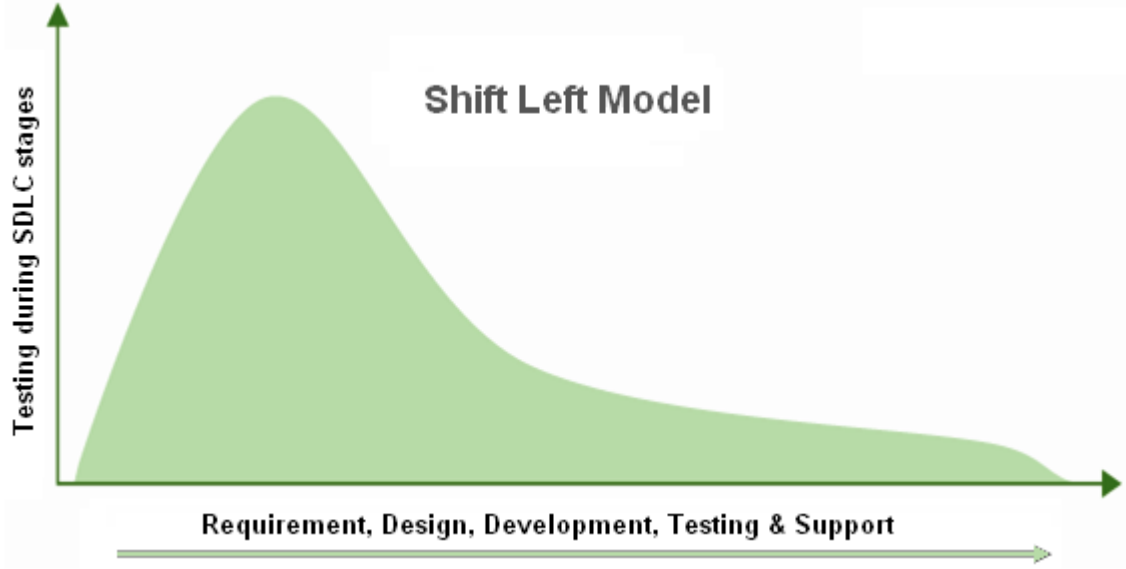
# DevOps Lifecycle

Utilizing a DevOps lifecycle, products can be continuously deployed in a feedback loop through:

- Infrastructure Automation
- Configuration Management
- Deployment Automation
- Infrastructure Monitoring
- Log Management
- Application & Performance Management




Yazılım Geliştirme Yöntemlerinin Geleneksel ve Çevik olarak yapılan sınıflandırmasında SDLC'nin yürütümünde Gereksinimlerin Yeri ve Test Süreçlerindeki Dönüşüm

https://www.bmc.com/blogs/what-is-shift-left-shift-left-testing-explained linkinden alınmıştır.