**SDLC iki şekilde ifade edilmektedir.**

- Software Development Life Cycle (Yazılım Geliştirme Yaşam Döngüsü)  
- Systems Development Life Cycle (Sistem Geliştirme Yaşam Döngüsü)

**Sistem Geliştirme Yaşam Döngüsü**

Sistem analistlerinin ve kullanıcı faaliyetlerinin spesifik döngüsünün kullanılmasıyla en iyi geliştirilmiş sistemin oluşmasını sağlayan analiz ve dizayn için safhalı bir yaklaşımdır. Sistem analisti problemlerin, fırsatların ve amaçların tanımlanmasına odaklanarak ilk safhada projenin geri kalanın başarısı için çok önemli bir rol oynamaktadır. Genel olarak bu safhanın faaliyetleri; müşteri yönetimi çalışmaları, elde edilen bilginin özetlenmesi, projenin sınırlarının tahminlenmesi ve sonuçların belgelemesini içerir. Bu safhanın çıktısı; problemin tanımı ve amaçların özetlenmesini içeren fizibilite raporunun hazırlanmasıdır.

**Yazılım Geliştirme Yaşam Döngüsü Nedir?**

Yazılım geliştirme yaşam döngüsü, (Software Development Life Cycle SDLC) yazılımın hem üretim hem de müşteri tarafından kullanımı süreçlerinde devam eden yazılımı geliştirmek için geçirdiği aşamaların tümüne verilen isimdir diyebiliriz. Bu süreçte zaman zaman kullanıcı ihtiyaçları ve ürün hataları ortaya çıktığı için, süreç hiçbir zaman tek yönlü olmayıp geriye dönüşlerin yapıldığı, tekrarlar ile sorunların çözüldüğü veya yeni özelliklerin elde edildiği bir döngü halini alır. Bu döngüler sayesinde yazılımın gelişimi daha rahat ve planlı bir hal alır. Bu döngüdeki bazı temel adımlar sayesinde zaman ve ekonomi yönetiminde daha hâkim olduğumuzu söyleyebiliriz.

SDLC süreci, müşteri beklentilerini karşılamayı, öngörülen zaman içerisinde projeyi tamamlamayı, maliyeti doğru bir şekilde hesaplamayı ve bu doğrultuda yüksek kaliteli yazılımlar üretmeyi amaçlayan kuruluşların olmazsa olmazları arasındadır.

Yazılım projelerinde bazı başarısızlıkların nedenleri kendini belirgin bir biçimde göstermektedir. Bunlardan başlıcaları şunlardır:

* Müşterinin isteklerini doğru analiz edememek,
* Ekiplerin görev tanımlarının belirsiz olması,
* Maliyet hesaplarındaki eksiklikler.

SDLC, saydığımız bu durumların olabildiğince az yaşanması veya mümkünse hiç yaşanmaması için uygulanması gereken adımlardan oluşan süreçtir ve birden fazla sayıda modele sahiptir. Bu modeller geçmişten günümüze değişen ihtiyaçlarla birlikte gelişmiş ve çeşitlenmiştir.

Standart bir yazılım geliştirme yaşam döngüsünün adımları şu şekildedir:

1. Planlama ve gereksinim analizi (Planning and requirement analysis)
2. Gereksinimleri tanımlama (Defining requirements)
3. Ürün mimarisini tasarlama (Designing the product architecture)
4. Ürünü oluşturma veya geliştirme (Building or developing the product)
5. Ürünü test etme (Testing the product)
6. Ürünü pazarlama ve bakım (Deployment in the market and maintenance)

**Planlama ve Gerekinim Analizi**

İhtiyaç analizi, SDLC’nin en önemli ve en temel aşamasıdır. Müşteri, satış departmanı, pazar araştırmaları ve alan uzmanlarının girdileri ile ekibin kıdemli üyeleri tarafından gerçekleştirilir. Bu bilgi daha sonra temel proje yaklaşımını planlamak ve ekonomik, operasyonel ve teknik alanlarda ürün fizibilite çalışması yapmak için kullanılır. Kalite güvencesi gereksinimlerinin planlanması ve projeyle ilişkili risklerin tanımlanması planlama aşamasında da yapılır. Teknik fizibilite çalışmasının sonucu, projeyi minimum risklerle başarıyla uygulamak için izlenebilecek çeşitli teknik yaklaşımları tanımlamaktır.

**Gereksinimleri tanımlama**

İhtiyaç analizi yapıldıktan sonra bir sonraki adım, ürün gereksinimlerini açıkça tanımlamak ve belgelemek ve müşteriden veya piyasa analistlerinden onay almaktır. Bu, proje ömrü boyunca tasarlanacak ve geliştirilecek tüm ürün gereksinimlerini içeren bir SRS (Yazılım Gereksinimi Belgesi) belgesi aracılığıyla yapılır.

**Ürün mimarisini tasarlama**

SRS, ürün mimarları için geliştirilecek ürün için en iyi mimari ile ortaya çıkabilecek referans. SRS’de belirtilen gerekliliklere dayanarak, genellikle ürün mimarisi için birden fazla tasarım yaklaşımı önerilir ve DDS – Tasarım Belge Şartnamesinde belgelenir. Bu DDS tüm önemli paydaşlar tarafından gözden geçirilir ve risk değerlendirmesi, ürün sağlamlığı, tasarım modülerliği, bütçe ve zaman kısıtlamaları gibi çeşitli parametrelere dayanarak ürün için en iyi tasarım yaklaşımı seçilir. Bir tasarım yaklaşımı, ürünün tüm mimari modüllerini (varsa) harici ve üçüncü taraf modülleri ile iletişim ve veri akışı gösterimi ile açıkça tanımlar. Önerilen mimarinin tüm modüllerinin iç tasarımı, DDS’deki detayları ile  açıkça tanımlanmalıdır.

**Ürünü oluşturma veya geliştirme**  
SDLC’nin bu aşamasında gerçek gelişme başlar ve ürün oluşturulur. Programlama kodu, bu aşamada DDS’ye göre üretilir. Tasarım ayrıntılı ve organize bir şekilde gerçekleştirilirse, kod üretimi çok fazla güçlük çekilmeksizin gerçekleştirilebilir. Geliştiriciler, kuruluşları tarafından tanımlanan kodlama yönergelerine uymalı ve kod üretmek için derleyiciler, tercümanlar, hata ayıklayıcılar vb. Programlama araçları kullanılmaktadır. Kodlama için C, C ++, Pascal, Java ve PHP gibi farklı üst düzey programlama dilleri kullanılır. Programlama dili, geliştirilen yazılımın türüne göre seçilir.

**Ürünü test etme**

Bu aşamada, modern SDLC modellerinde olduğu gibi, genellikle tüm aşamalardan bir alt küme, test etkinlikleri çoğunlukla SDLC’nin tüm aşamalarında yer alır. Bununla birlikte, bu aşama, ürün SRS’de tanımlanan kalite standartlarına erişene kadar, ürün kusurlarının raporlandığı, izlendiği, sabitlendiği ve tekrar test edildiği test aşamasına işaret eder.

**Ürünü pazarlama ve bakım**

Ürün test edildikten ve kurulmaya hazır hale getirildiğinde, uygun piyasada resmen piyasaya sürülür. Bazen ürün dağıtım işlemi, o kuruluşun işletme stratejisine göre kademeli olarak gerçekleşir. Ürün ilk önce sınırlı bir segmentte serbest bırakılabilir ve gerçek iş ortamında test edilebilir (UAT- Kullanıcı kabul testi). Ardından, geri beslemeye dayanarak, ürün, olduğu gibi bırakılabilir veya hedefleme pazar segmentinde önerilen geliştirmelerle bırakılabilir. Ürün piyasaya sürüldükten sonra bakım, mevcut müşteri tabanı için yapılır.

SLDC aşamalarına ek olarak bahsedilmesi gereken bir diğer husus ise bu aşamalarda görev alanlar. Standart bir SDLC takımında project manager (PM), business analyst (BA), developer (Dev), quality analyst (QA) gibi roller bulunur. Farklı SDLC modellerinde bu rollerden daha fazla rol bulunabilir ancak temelde saydığımız bu rollerin olması gerekir.

**Proje yöneticisi (PM)**

**Takımdaki herkesin rolünü bilmesinden ve yerine getirmesinden sorumludur. Proje planının geliştirilmesi sürecinde proje müşteri ile yakın ilişki kurması gerekir. Takım içerisindeki iletişimi sağlar. Proje riskini yönetir. Proje çizelgesini hazırlar ve proje bütçesini yönetir. Yazılım anlamında teknik bilgisinin yüksek olması şartı yoktur. Daha çok yönetici yeteneklerinin iyi düzeyde olması beklenir. Ancak PM’nin teknik bilgi seviyesi arttıkça yazılım projeleri daha başarılı olma olasılığı da artabilir.**

**İş analisti (BA)**

Bir projeyi analiz eden, proje süreçlerini tasarlayan ve iş modelini değerlendiren kişidir. Teknik bilgisi yüksek olmayan PM ile developer ekibi arasında bir köprü görevi görür. BRD ve FRD'yi oluşturur ve bir takım functional requirement topladıktan sonra use cases oluşturur.

**BRD**, projelerde iş/talep sahibinin proje tamamlandıktan sonra gerçekleştirmeyi istediği ihtiyacın ne olduğunu ve neden yapıldığını anlatan dökümandır. Bu ihtiyaçlar projenin kapsamını belirler, dolayısıyla ihtiyacın ve beklenenin ne olduğunun net bir biçimde anlatılması gerekmektedir. Doküman, etkilenen tüm paydaşların katılımı ile doldurulmalıdır. Ekipler arasında bir anlaşma mahiyetindedir.

**FRD** , iş ihtiyaçlarını karşılamak için gereken gereksinimleri ayrıntılandıran resmi bir belgedir. Belge, bir sözleşmenin amacına hizmet eder, böylece müşteri, ürünün yetenekleri için kabul edilebilir bulduğu şeyleri kabul edebilir. İşlevsel gereksinimler belgesi, ürün geliştirme için temel bir belgedir.

**Geliştirici (Dev)**

Yazılımın kodlamasından sorumludur. Kendilerine aktarılan software requirement dokümanını analiz eder, gereken soruları sorar ve gereken programın oluşumunu sağlar. Beklentileri ve gereksinimleri karşılayacak yüksek kalitede kod yazarlar.

**Kalite analisti (QA)**

Oluşturulan projenin arzu edilen çıktı çerçevesinde son kullanıcının (end-user) beklentilerini karşılayacak seviyede olmasını sağlar. Projedeki hatalar kalite analisti tarafından bulunmalıdır ki developerlar bu hataların üzerinde calışıp sorun teşkil etmeyecek ürün ortaya koyabilsinler. Test planları stratejilerinin oluşturulması, test vaka ve senaryolarının hazırlanması, test verilerinin bir araya getirilmesi ve test caselerin execute edilmesi gibi sorumlulukları vardır.





