YBS Ansiklopedi

www.YBSAnsiklopedi.com

Cilt 1, Sayı 2, Ekim 2014

Yazılım Geliştirme Hayat Döngüsü (Software Development Life Cycle)

Sadi Evren SEKER

American University of Middle East, Kuwait, academic@sadievrenseker.com

1. Giriş

Her geliştirme işlemi özellikle de ürün geliştirme işlemi belirli aşamalar içerir. Yazılımların da birer ürün olduğu unutulmamalıdır. Bu bağlamda nasıl bir otomobil, bir uçak veya bir televizyonun geliştirilmesi için çeşitli aşamalardan bahsetmek mümkünse, yazılım için de benzer aşamalar mevcuttur. Genel olarak bu ürünlerin tamamını birer sistem olarak görmek ve sistem geliştirme metodolojisi ismi altında incelemek mümkündür.

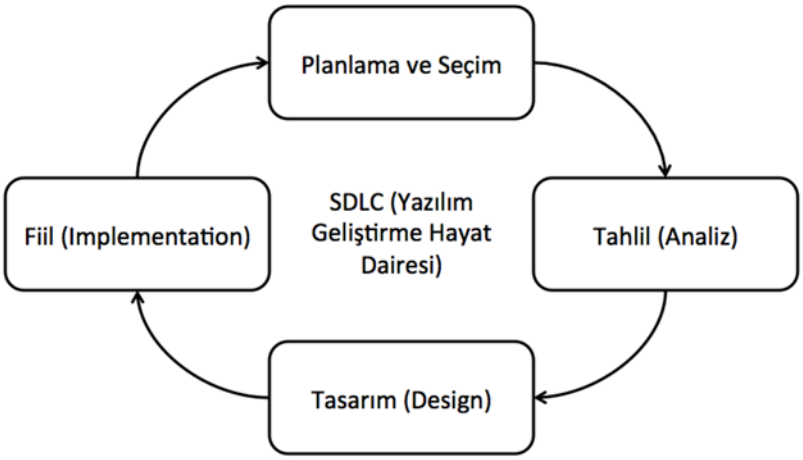
Her sistem öncelikle tasarlanır, üretilir, piyasaya arz edilir ve neticede piyasadan yok olur, yerine yeni bir sistem arz edilir ve bu daire böylece devam eder. Benzer şekilde yazılım da öncelikle bir ihtiyaçla başlar. Birileri bir bilişim sisteminin neler yapması gerektiği, ihtiyaçlara nasıl cevap vermesi gerektiği konusunda fikirle işe başlar. Bu bilişim sistemini kullanacak olan yapının altında ayrılabilecek kaynaklar belirlenir. Ardından bu yazılımın yerine ikame edileceği mevcut sistem tahlil edilir. Örneğin bir Doküman yönetim sistemi için mevcut işleyiş incelenir. Mevcut işleyişe yeni beklentiler sisteme eklenerek, sistemin tanıtımı yapılır, gerekli eğitimler verilir ve sistemin kurulumu gerçekleştirilir.

Her kurumda benzer aşamalardan geçilen yazılım geliştirme süreçleri ve bilişim sistemleri kavramlarını farklı kelimeler ve aşamalarla anlatmak mümkün olsa da genel olarak aşağıdaki maddeler altında toparlamak mümkündür.

- Planlama ve seçim aşaması
- Tahlil (analiz) aşaması
- Tasarım (Design) aşaması
- Fiil aşaması (implementation)

Yukarıdaki aşamalar şekil 1'deki gibi görselleştirilebilir ve bu dört aşama aslında bir dairenin merhaleleridir. Yani bütün aşamalar tamamlandıktan sonra tekrar başa dönülerek her şey yeniden devam eder. Bunu örneğin piyasadaki ürünlerin yaşamlarının son bulması ve yeniden yeni ürünlerin piyasaya girmesi olarak görmek mümkündür. Aslında her yazılımın da bir ömrü vardır ve bu ömür tamama erdikten sonra yazılımda gerekli yenilemeler ve değişikliklerle yeni yazılım müşterinin / kurumun hizmetine sunulur ve bu süreç bir süreklilik arz eder.

Yukarıdaki bu aşamaları aşağıdaki şekilde detaylandırmak mümkündür.



Şekil 1 *SDLC (Yazılım Geliştirme Yaşam Dairesi)*

2. Sistem Planlama ve Seçim Aşaması

Bu aşamada, iki önemli adım atılmalıdır. Birincisi birisinin yeni ve daha gelişmiş bir bilişim sistemine olan ihtiyacı belirtmesi ikincisi ise bilişim sisteminin kullanılacağı ortamdaki ihtiyaçların belirlenmesidir.

Bu iki adım atılırken aşağıdaki yan amaç ve sorular gündeme getirilmelidir:

- Mevcut sistemdeki problemlerin çözülmesi için kullanılan yöntemler ve bu yöntemlerin iyileştirilmesi
- Sistemin iyileştirilmesi için ortamdaki beklenti düzeyi ve buna olan inanç

3. Bilişim Sisteminin Gerekliliği ve Sisteme Yapacağı Katkı Düzeyi

Yazılım Geliştirme Yaşam Döngüsü (Software Development Life Cycle (SDLC)).

- **Planlama:** Bu aşama, planlamanın bütün gereklerini tamamlar. Zaman kaynak ayrımı (Seker ve Diri, 2010), personel, kurumun mevcut bilişim seviyesi ve yeni sistemin kullanılabilirliği, uzun vadede sistemin bakımı ve sürdürülebilirliği gibi çok sayıda parametre bu aşamada incelenir. Bu aşamanın tamamlanmasının ardından bir uygunluk raporu (feasibility report) hazırlanarak kurumdaki irade sahiplerine (yönetim) sunulur ve bilişim sisteminin yapacağı olumlu katkıların maliyet analizine göre katma değeri olduğu kesinleştirilir.
- **Sistem Tahlili (Analiz):** Bu aşamada sistem analizi yapılır ve geliştirilmekte olan bilgi sisteminin etkilediği diğer birimler ve mevcut işleyiş tahlil edilir. Örneğin geliştirilmekte olan sistem ödeme emirleri ile ilgili olsun, bu durumda girilen emirlerin satış, defteri kebir, stok kayıtları gibi çok sayıdaki etkilediği diğer kayıt tahlil edilerek bu sistemlerle olan etkileşimi çalışılır.
- **Sistem Tasarımı (Design):** Bu aşamada sistemin bütün bileşenleri ve genel yapısı tasarlanır. Çok çeşitli tasarım yöntemleri ve aşamaları olmakla beraber genelde mantıksal bir tasarım aşamasından geçilerek sistemin hedefleri ve etkileşimde olduğu alt bileşenleri arasındaki uyum gözetilir. Ardından bu mantıksal tasarımın fiziksel tasarıma çevrilmesi aşaması gelir. Fiziksel tasarıma çevrilim aşamasında, programlama dilinden kullanılacak olan teknolojilere ve sistemin bütün parçalarına kadar olan tasarımı tamamlanır. (veri tabanı tasarımı, kullanıcı ekranları, veri akış yolları gibi).

• **Uyarlama ve işletme aşaması (implementation and operation):** Bu aşama, şimdiye kadar yapılan bütün kağıt üzerindeki işlemlerin gerçeğe dönüştürüldüğü ve yazılım halini aldığı aşamadır. Kodlama, kod kontrolleri, testleri, kurulumu, yönetimi gibi işlemlerin tamamı bu aşamada yapılır.

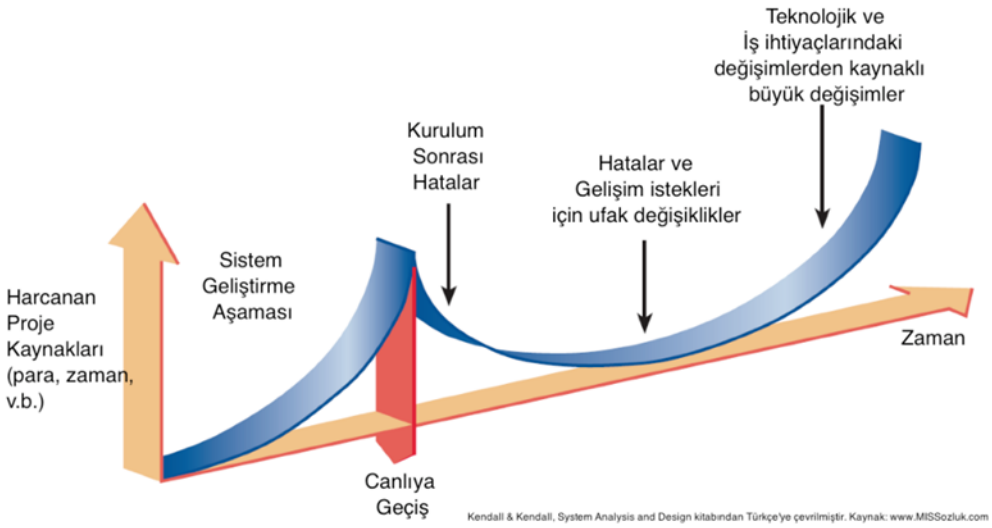
Yukarıdaki 4 aşama kapsamında SDLC adımlarının çıktıları aşağıdaki şekilde özetlenebilir:

Aşama	**Neticeler**
Planlama ve seçim safhası	Sistem öncelikleri, Veri, ağ, donanım ve bilgi sisteminin mimarisi, Seçilmiş olan proje için detaylı çalışma planı, Sistemin geçerlilik alanı için özellikler, Sistemin iş akışı açısından önemi
Sistem Tahlili	Mevcut sistemin tanımı, Mevcut sistemin düzeltilmesi, iyileştirilmesi, değiştirilmesi için öneriler, Muhtelif sistemler için açıklamalar ve alternatiflerin değerlendirilmesi, Yeni teknoloji gereksinimlerinin belirlenmesi ve geçiş için öneriler
Sistem Tasarımı	Sistemin bütün alt bileşenleri için detaylı birer tasarım
Uyarlama ve işletme	Kod, Dokümantasyon, Eğitim ve destek kaynakları, yeni sürüm için kod, eğitim ve destek aşamalarındaki değişiklikler.

Yukarıdaki 4 adımlı yaklaşımın yanında Kendall and Kendall tarafından sunulan 7 adımlı yaklaşımdan söz etmek de mümkündür. Bu yaklaşım aşağıdaki adımlardan oluşur (bkz. System Analysis and Design, Kendall and Kendall):

1. Problem, fırsatlar ve hedeflerin belirlenmesi
2. İnsan seviyesi enformasyon ihtiyaçlarının belirlenmesi
3. Sistem ihtiyaçlarının (isterlerinin) belirlenmesi
4. Şimdiye kadar olan adımlar ışında elde edilen isteklere cevap veren bir sistemin tasarlanması
5. Yazılımın geliştirilmesi ve dokümantasyonu.
6. Sistemin testleri ve bakım adımlarının belirlenmesi.
7. Sistemin gerçeğe alınması ve değerlendirilmesi.

Yine Kendall ve Kendall kitaplarında bu geliştirme sürecindeki kaynak harcama miktarlarını şekil 2'de gösterildiği gibi modellemektedir.



Şekil 2 *Yazılım Projeleri Kaynak Değişim Grafiği*

Şekilden de anlaşılacağı üzere, sistemin geliştirilme aşamasında sürekli artan (üstel olarak artan) proje maliyeti varken, projenin canlıya geçmesinin ardından bu maliyet düşmeye başlar. Zamanla sistemdeki istek ve ihtiyaçların ölçeğinin artması ile proje maliyetleri de artmaya başlar. Hatta sistemin geliştirme aşamasındaki maliyetlerin bile üzerine çıkabilir. Bu yüzden projenin tasarımının ve analizinin ileride çıkabilecek ihtiyaçları en verimli şekilde karşılayacak yeterlilikte olması çok önemlidir.

Ayrıca yukarıdaki SDLC yaklaşımı için alternatif olarak sunulan yöntemler veya araçlardan da bahsetmek mümkündür.

Örneğin bilgisayar destekli yazılım mühendisliği araçları (computer allied software engineering, CASE tools), ortak uygulama tasarımı (joint analysis design, JAD), hızlı uygulama geliştirme (rapid application development, RAD), katılımcı tasarım (participatory design, PD), atik yöntemler (agile methodologies) (Seker, 2015) gibi farklı birer yazı konusu olan yöntemler kullanılabilir.

SDLC klasik bir yaklaşım olup, yazılım mühendisliği alanındaki şelale modeline benzetilebilir. Bu modelin yanında çevik geliştirme ve nesne yönelimli geliştirme gibi yöntemler de kullanılabilir.

Kaynaklar

Kendall E. K. ve Kendall J. E. (2013), System Analysis and Design, 9th ed. Prentice Hall, 2013, ISBN: 0133023443

Seker, S.E. and Diri, B. (2010), TimeML and Turkish Temporal Logic, International Conference on Artificial Intelligence, ICAI'10, v. 10, pp. 881-887

Seker, S. E. (2015), Çevik Yazılım Geliştirme (Agile Software Development), YBS Ansiklopedi, v. 2, is. 1, pp. 16- 18