Yazılım Yaşam Döngüsü

Ve Modelleri

Hazırlayan: Utku Özkutucu

Yazılım Nedir?

Yazılım ile Alakalı bir konuyu ele alacaksak ilk önce yazılımın ne demek olduğunu öğrenmeliyiz. Yazılım, elektronik aygıtların belirli bir işi yapmasını sağlayan programların tümüne verilen isimdir. Bir başka deyişle, var olan bir problemi çözmek amacıyla bilgisayar dili kullanılarak oluşturulmuş anlamlı anlatımlar bütünüdür, yani bir üründür ve her ürünün olduğu gibi yazılımında bir ömrü vardır. Yazılım Tasarlanır, üretilir, Kullanılır, eskir, bakım yapılır ya da çürümeye bırakılır.

Yazılım Yaşam Döngüsü Nedir? (Software Development Lifecycle)

Yazılım Yaşam Döngüsü; bir yazılımı en ucuz maliyet, en iyi kalite, ve en kısa zamanda yapmamıza olanak sağlayan yapılandırılmış bir süreçtir. Amacı müşterinin bütün ihtiyaç ve beklentilerini karşılayabilecek bir yazılım üretmektir. Yazılımın yaşam döngüsü adından da anlaşılacağı üzere döngüseldir ve aşamaları sırayla şöyledir:**planlama**,**analiz**, **tasarım**,**üretim, test, dağıtım** ve **bakım** olmak üzere 7'ye ayrılır.

Yazılım Yaşam Döngüsü Neden bu kadar önemli?

- Projenin planlanmasına ,ne kadar uzun süreceği konusunda yardımcı olur
- Projenin kontrolünü kolaylaştırır.
- Projenin yapımını hızlandırır.
- Projedeki riski azaltır.
- Müşterinin İsteklerini daha kolay anlamamızı sağlar
- Projenin yapımındaki masrafı düşürür.

Planlama:

Yazılım yaşam döngüsünün ilk ve temel adımıdır. Planlama aşamasında projeyi yapan grup para kazanacak hissedar ve kullanacak üyelerle konuşulur. Kaynak ayırma, Risk planlaması, ürünün kapsayacağı özellikler, ürünün amacı, proje zamanlaması ve projenin nasıl yapılması gerektiği ile alakalı genel bir taslak oluşturmamızı sağlar.

Analiz:

Planlamada konuşulmuş olan ihtiyaçlar, amaçlar, kaynaklar risk planlamaları, zaman gibi faktörlerin yapılacak olan ürün için teknik olarak mümkün mü analiz edilir. Planlama ile Analiz arasındaki fark: Planlama; program için genel bir taslak oluşturmaya yararken, analiz; tasarım ve üretim için bir yol haritası çizmemizi sağlar ve bunların mümkün olup olmadığını denetlememizi sağlar.

Tasarlama:

Tasarlama, projedeki en önemli aşamalardan biridir. Projenin gereksinimleri ve amaçları doğrultusunda olabilecek en uygun teknolojiyi kullanırken, en az maliyet ile çıkılmaya, en pratik yolu seçmeye, iyi bir arayüz yapmaya ve bilgisayar mimarisine dikkat edilmelidir, iyi, kolay, anlaşılabilir, pratik olan bir ürününüz olmalı, yoksa kullanıcılar sizi kullanmayabilir ya da program geliştirilirken bir sürü arıza oluşumuna sokabilir. Bu da bir süre sonra bizi içinden çıkılamayan bir duruma sokabilir, böyle durumlarda proje tekrardan yapılır, bu da ayrı bir masraf demektir.

Üretim:

Bu aşamada artık üründe bir gelişme başlar. Üretim sırasında programlama dilleri veri tabanları ve tasarım aşamasında belirlediğimiz özellikleri kullanırız. Eğer iyi bir tasarlama sürecinden geçildiyse bu aşama pek zorlu değildir, tasarlama aşamasındaki kararlardan uzaklaşılmadıkça bir sorun oluşmaz.

Test:

Ürettiğimiz yazılımın planlama ve tasarlama aşamasında karar verdiğimiz beklentilerimizi ve ihtiyacımız olan özellikleri yerine getirip, getiremediği kontrol edilir. Bunların dışında performans, işlevselliği etkileyebilecek arızalar varsa diye yazılım test edilir. Test süreci kendi içinde de test türlerine ayrılır: Parça testi, kod kalite testi, entegrasyon testi, sistem testi ve kabul testi.

Dağıtım:

Dağıtım aşamasında, yazılımımız bulut altyapısına kurulur ve yapılandırılır. Kurulum tamamlandıktan sonra kullanıcılarla buluşur. Dağıtımın devam etmesi için de programın sürekli bir bakıma ihtiyacı vardır.

Bakım:

Bakım, kullanıcıların değişen ihtiyaçlarını gideren, Eskimiş özelliklerin kaldırılmasını, kullanıcı dostu hale getirilmesi, yazılımın işlevsel ve fonksiyonel kalmasını sağlayan aşamadır.

Yazılım Yaşam Döngüsü Modelleri:

Şelale (Waterfall) Modeli:

Şelale modeli, eski yazılım yaşam döngüsü modellerinden biridir. Bir sonraki aşamaya geçmek için ilk önceki aşama tamamlanmalıdır, tamamladıktan sonra bir sonraki aşamaya geçilir fakat önceki aşamaya dönüş olmaz. Her aşama, önceki aşamadan gelen bilgiye dayanır. Bu model kolay ve anlaşılabilir bir modeldir, fakat kötü bir yönü var **Esneklik.** Değişim gerektiren, uzun ve geniş projeler için uygun değildir, çünkü geriye bir dönüş yapılamayacağı için bir değişiklik yapılacağı zaman projenin bitirilmesi ve daha sonra bakım aşamasında düzeltilmeye çalışır. Bu aşamaya gelene kadar daha fazla hata ortaya çıkabilir ve bu da hataların biz çözene kadar birikmesi anlamına gelebilir.

İterative (Yinelemeli) Model:

Yinelemeli model; yazılımın planlama, tasarlama, üretme ve test aşamalarının sürekli tekrarlanması üstüne kurulmuş bir yazılım yaşam döngüsü modelidir (SDLC). Geliştirme ve değişikliklerin döngüler dolayısı ile kolayca yapılabileceği esnek bir modeldir. Hatalar çok kolay bir şekilde tespit edilebilir. Yapım aşamasında ne yapılacağı anlaşılmamış ya da yazılım çok değişebileceği yazılımlarda kullanılır. Bu kadar iyi yöne rağmen, iterative model, diğer modellere göre daha çok zaman harcayan, sürekli denetim gerektirdiğinden uzman olmayı gerektiren ve daha masraflı bir yöntemdir.

Helezonik (Spiral) Model:

Helezonik model, şelale ve yinelemeli modeli birleştiren bir yazılım yaşam döngü modelidir. En esnek modellerden biridir. Risk analizi ve yönetiminin önemini vurgulayan yazılım geliştirme modelidir. 4 Ana aşaması vardır: **Planlama**, **risk analizi**, **Mühendislik**, **Değerlendirme**. Planlamada İhtiyaçlar belirtilir. Risk analizinde, projede olabilecek arızalar ve nasıl çözülebileceği dikkatlice incelenir. Mühendislikte; yazılım tasarlanmış, geliştirilmiş ve test edilmiştir. Projeyi yapan takım bir önceki yaptıklarının üstüne yine yeni bir şeyler katar. Değerlendirme, Bu aşamada programın ihtiyaç ve amaçlar doğrultusunda çalışıp çalışmadığından emin olunur. yazılımın işlevsel, verimli ve etkili olduğunu sağlamak için kullanıcı kontrolünden geçer. Değerlendirme aşamasından sonra planlama aşamasına geri dönülür ve döngü tekrar başlar böylece bir önceki döngüde bilinenlerin üstüne yeni ihtiyaçlar veya özellikler eklenir. Spiral Model, iterative modelde olduğu gibi esnek, başlangıçta anlaşılmamış projeler veya değişikliğe gidilecek projeler için çok iyi bir seçimdir. Yine iterative'de olduğu gibi çok zaman harcatan, masraflı ve uzmanlık gerektiren bir Modeldir. İterative'den farkı, spiral modelde risk analizi daha ön plandadır.

V Model:

Şelale modelinin gelişmiş bir halidir. Onaylama Aşaması eklenmiştir. Şelalede olduğu gibi bir aşama bitmeden sonrakine geçilmez. Avantajları: Kolaydır, Tasarım ve planlama testleri, kodlamadan önce yapıldığı için başarı şansı şelaleye oranla yüksektir. Dezavantajları: Esnek değildir, Büyük projelerde kullanılması zordur, projede ilk ürün yazılım üretim aşamasından geçince elde edilir.

Incremental Model:

Incremental modelde bütün gereksinimler Farklı Alanlara ayrılmıştır. Gereksinimleri ele alabilmek içinde birden fazla Geliştirme döngüsü kullanılır. Çoklu Şelale modeli Sistemi denilebilir. Bu döngüler de küçük ve daha kolay yönetilebilen parçalara ayrılıp **modül** adını alır. Bu modüllerin hepsi gereksinimlerin bulma, tasarlama, üretim ve test aşamalarından geçer. Programın çalışan ilk versiyonu ilk modülde gerçekleşir, bu da bize program üzerinde bir izlenim bırakır. Bir sonraki modüller program bitinceye kadar bir önceki modülün üstüne eklene eklene gelişir ve artık yazılım tamamlanır. Şelale'ye çok benzer fakat küçük modüller bize büyük bir artı sağlar. Modüller küçük ve denetimi kolay olduğu için arızalar hızlı anlanır. Özellik değiştirmek modüllere bağlı olduğu için ucuzdur, fakat proje çok denetim ve iyi bir planlama gerektirdiği için masraf Şelale modeline göre fazladır.

Çevik (Agile) Model:

Çevik Model, Incremental model'in bir türüdür. Incremental model gibi küçük gelişmeler eklenir, fakat daha hızlıdır. Çevik model Daha iyi yazılım geliştirmeyi sağlayan Agile Manisfesto üzerine kurulmuştur.Avantajları: Kullanıcı ile sürekli iletişim halinde olunur böylece istekler ve değişimler hızlı bir şekilde bildirilir. Değişime uygun ve hızlı cevap verebilmeyi amaçlar. Dezavantajları: Büyük projelerde başlarda değerlendirme yapmak zordur, Kullanıcılar ne istediğini yeterince iyi açıklayamazlarsa projenin çökmesi uzun sürmez. Uzman programcılar gerektirir. Belgelendirmeler azdır.

SCRUM Nedir?

Scrum; karmaşık problemleri, takımları yönlendirerek uyarlanabilir çözümlerle buluşturmayı hedefler. Uygulayanların verimliliğini ve işin kalitesini arttırmaya yarayan düşünme felsefesi olan, Yalın düşünmeyi kullanarak zaman ve odak kaybını engeller. Scrum inceleme ve adapte olmak için 4 alanı **Sprint** adı altında birleştirir. 4 Ana alan: Transparanlık, denetleme, adaptasyon, Scrum'un Değerleri.

Transparanlık: Yapılan süreç İşçiler kadar, kullanıcılar tarafından da görüntülenebilmelidir. Eğer Transparanlık yoksa denetim'in doğru ve dürüst olması beklenemez.

Denetim: Geliştirme süreci, sürekli ve sık sık denetlenmelidir. Bu denetim bizi istenmeyen özellikler ve hataları algılamamızı sağlar. Denetim, adaptasyonun iyi yapılmasını sağlar.

Adaptasyon: Ürün kabul edilebilir sınırların dışında veya kabul edilemez ise Uygulanan işlemler ve kaynaklar azaltılmalı, böylece ürünün daha da kötü hale gitmesi önlenilir.

Scrum Değerleri: Scrum çalışanların 5 değer altında daha efektif olmasını sağlar: Bağlılık, odaklanma, açıklık, saygı, cesaret.

Scrum Takımları: Scrum takımı; ürün sahibi, geliştiriciler, Scrum uzmanı içerir. Sadece ürüne odaklanan profesyonel bir takımdır. Geliştiriciler: Scrumda kullanılabilecek bir sistem üretmeye çalışırlar. Sprint için bir plan olan **Sprint Backlog**'u üretirler. Ürün Sahibi: Ürünün değerini maksimum hale getirmeye çalışır. Scrum Master: Scrum Takımının efektifliğinden ve projenin herkes tarafından anlaşıldığından sorumludur.

Sprint Nedir: 1 ile 4 hafta arası takımın potansiyel olarak tamamlanmış bir ürün yapmak için ürün üzerinde çalışılan dönemidir. Sprint Planning, daily Scrum , sprint review, sprint retrospective gibi ürüne ulaşmayı sağlayan çalışmalar sprint içerisinde olur. Scrum bir ayı geçmemeli yoksa risk artışları, odak kayıpları ve karmaşıklık artışları görülebilir.

Sprint Planning: Planlama aşaması sprintin başında gerçekleştirilir. Sprintte ne değerlidir, ne üzerine çalışılmalı gibi sorular cevaplanır.

Daily Scrum: Daily Scrum 15 dakikalık bir toplantıdır. Amaç karmaşıklığı azaltmak, hedefe doğru ilerlenmiş diye kontrol etmek ve çalışanların odaklarını yenilemektir. Sabah her gün aynı saatte yapılır. Eğer Ürün sahibi ve scrum uzmanı product backlog üzerinde çalışıyor ise onlar da bu toplantıya katılmalıdır.

Sprint Review: Sprint Review'in amacı sprint'in iyi geçip geçmediğini analiz etmek ve bundan dolayı oluşabilecek gelecekteki adaptasyonları belirlemektir. Ürünün hazır olup olmadığı belirlenir

Sprint Retrospective: Amacı Kaliteyi ve efektifliği arttıracak yollar aramaktır.

Sprint bileşenleri: Product Backlog; ürün hedefini belirler, üründe neyin geliştirilmesi gerektiği belirlenir ve buraya kaydedilir. Sprint Backlog; Sprint aşamasında neyin neden yapıldığını, product backlogda seçilen bu ürünün neden geliştirilmesi gerektiği ile alakalı bir karara varılır. Tamamlanmış olan ürün için nasıl bir dağıtım aşaması izleneceği de burada karar verilir.

Kaynakça:

- https://www.synopsys.com/glossary/what-is-sdlc.html
- https://www.roberthalf.com/blog/salaries-and-skills/6-basic-sdlc-methodologies-which-one-is-best
- https://www.tutorialspoint.com/sdlc/sdlc_overview.htm
- https://www.linkedin.com/pulse/yazılım-yaşam-döngüsü-nedir-veysel-ugur-kizmaz/?originalSubdomain=tr
- https://www.bilimozu.com/post/yazılım-yaşam-döngüsü-modelleri-her-üründe-olduğu-gibi-yazılımların-da-bir-yaşam-döngüsü-var-mıdır
- https://www.javatpoint.com/software-engineering-sdlc-models
- https://medium.com/architectural-patterns/yazılım-geliştirme-modelleri-62915545c51e
- https://tryqa.com/what-is-agile-model-advantages-disadvantages-and-when-to-use-it/
- https://tryqa.com/what-is-v-model-advantages-disadvantages-and-when-to-use-it/
- https://tryqa.com/what-is-incremental-model-advantages-disadvantages-and-when-to-use-it/
- https://tr.wikipedia.org/wiki/Atik_yazılım_geliştirme
- https://en.wikipedia.org/wiki/Lean_thinking
- https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf#zoom=100
- https://www.scrum.org/learning-series/what-is-scrum

https://www.linkedin.com/in/utku-özkutucu-15b705269/

https://github.com/UtkuOzkutucu

https://medium.com/@utku__ozkutucu