Yazılım Yaşam Döngüsü Nedir?

Bir yazılım ürünü fikrinin doğuş anından yazılım ürününün geliştirilmesi, test süreci, hayata geçişi ve bakımına kadar olan tüm aşamaları kapsayan sürece “Yazılım Yaşam Döngüsü” adı verilir. Bu döngü, çift yönlü bir döngüdür. Döngünün her adımında, önceki adıma geri dönüp o adım tekrar geliştirilebilir.

Yazılım Yaşam Döngü Aşamaları Nelerdir?

1-Planlama

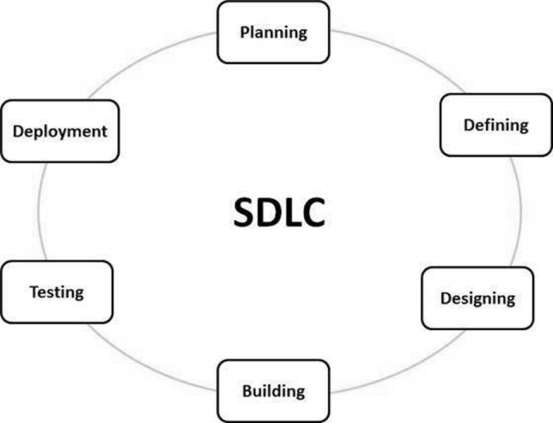
2-Analiz

3-Tasarım

4-Geliştirme

5-Test

6-Bakım



1-Planlama Aşaması

Oluşturulacak yazılım ürününe göre süre, maliyet, üretici ekip, izlenecek yöntem, kaynaklar ve yapılması gereken işlerin listesi bu aşamada belirlenir.

2-Analiz Aşaması

Bu aşamada, müşterinin istekleri net anlaşılmalı ve sistemin gereksinimleri kesin olarak belirlenmelidir. Bunlara uygun olarak da yazılım ürününün bir UML diyagramı oluşturulup proje somut hale getirilmelidir. Projenin ana mantığının ve sorunlara karşı çözümlerin oluşturulduğu bir aşamadır.

3-Tasarım Aşaması

Sistemin tasarımı yapılır. Gereksinimleri karşılayacak şekilde yazılım ürünü üzerinde tercihler yapılır. Son kez bu aşamada proje üzerinde tartışılmalı ve düzeltmeler yapılmalıdır. Bir sonraki aşama olan “Geliştirme Aşamasına” geçildiğinde akıllarda hiçbir soru işareti kalmamış olması gerekir.

4-Geliştirme Aşaması

Müşterinin tercihlerine ve proje tasarımına uygun olarak ürünün prototipi oluşturulur. Bu aşamada, veri tabanı oluşturulur, kodlama yapılır, sistem ve kullanıcı ara yüzleri oluşturulur. Kodlama yapılırken anti-pattern durumların oluşmamasına dikkat edilmelidir.

5-Test Aşaması

Test, her aşamanın sonunda yapılabilen bir adımdır fakat bahsedilen bu aşamada, sistemin geneline bir test yapılır. Entegre sorunları varsa giderilir, kodlamalarda ve tasarımda değişiklik yapılabilir. Ürünün son hali teslim edilip ürün hayata geçirilir.

6-Bakım Aşaması

Bakımlar; sistemde zamanla meydana gelen eksiklikleri ve hataları ortadan kaldırmak, sisteme yeni eklentiler sunmak üzere yapılan işlemlerdir. Doğrulama veya kusursuzlaştırma için de bakım yapılabilir. Bakım aşaması, yazılım ürününün yaşamı boyunca süregelir.

Yazılım Yaşam Döngü Modelleri Nelerdir?

1. Gelişigüzel Model

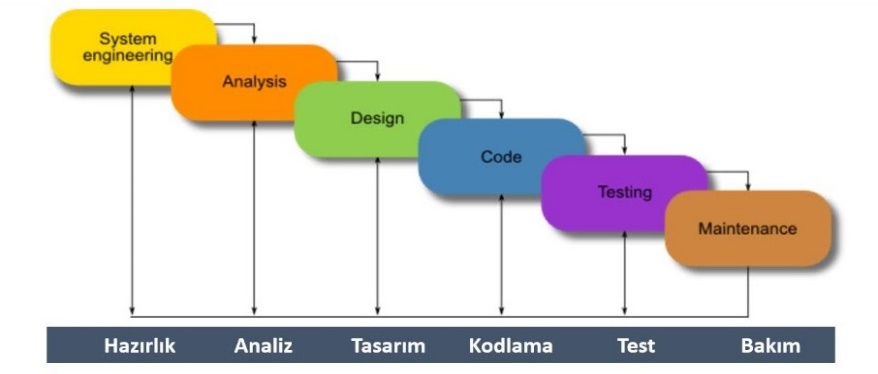
Bu yönteme model demek bile yanlış olur. Çünkü belirli adımları veya metodu yoktur. Genelde tek bir kişinin kafasına göre ilerleyerek ortaya çıkarabileceği kolay, anlaşılır ve basit yazılım ürünleri için kullanılır.

1. Barok Modeli

Bu modelde yaşam döngüsü doğrusal ilerler ve aşamalar arası geri dönüş sağlanmaz. Yazılım, geliştirip test adımlarını geçtikten sonra üretilebilir. 1970’lerde kullanılmıştır, günümüzde yaygın olarak kullanılmaz.

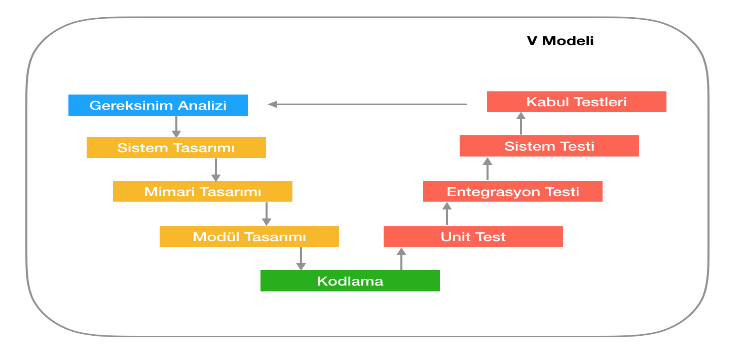
1. Şelale Modeli

Küçük projeler ve deneyimsiz personeller için daha uygun bir modeldir. Günümüzde pek tercih edilmemektedir. Modelde, her adım birbirinin üstüne eklenerek lineer bir şekilde ilerler. Her adım kendinden sonraki adım için kaynak oluşturur. Müşteri bu sürecin içine dahil edilmediği için müşterinin istekleri projenin başında net anlaşılmadığı zaman projeyi geri döndürme ve yenileme maliyeti fazla olur.



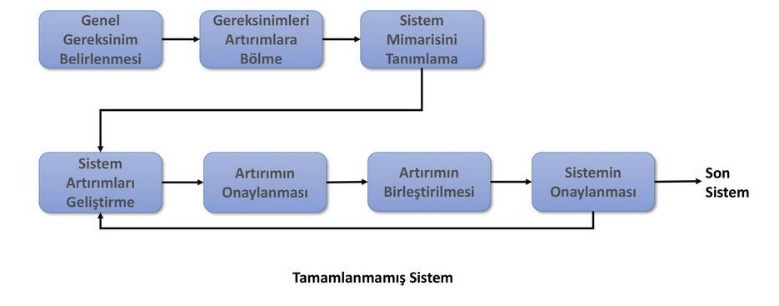
1. V Modeli

V modelinde sol kol **üretim** aşamalarını gösterirken, sağ kol ise **test** aşamalarını gösterir. Kolay-orta seviye projeler için kullanılabilir. İşin net, belirsizliğin az olduğu **bilgi teknolojileri** gibi projelerde kullanılabilir. Müşteri işin içinde olmadığından, proje bitiminden sonra yapılacak geri dönüşler projenin maliyetinin artmasına sebep olabilir.



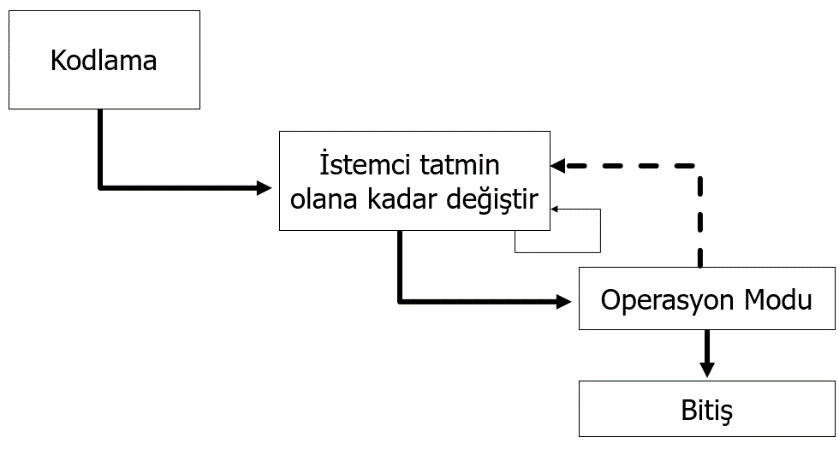
1. Artımlı Model

Kullanıcının gereksinim önceliğine göre bir plan oluşturulur. **Divide and Conquer (Böl ve Yönet)** yaklaşımı ile plan parçalara bölünür. Her parça kendi içinde geliştirilir. Zamanla parçalar birleştirilerek işlevsellik gitgide artırılır. Böylece ilerleme daha rahat gözlenir ve risk en aza indirilmiş olunur. Bir hatanın veya değişikliğin farkındalık oluşturma süresi kısa olduğu için de yenileme maliyeti düşüktür. Deneyimli personellerce kullanılabilir.



1. Kodla ve Düzelt Modeli

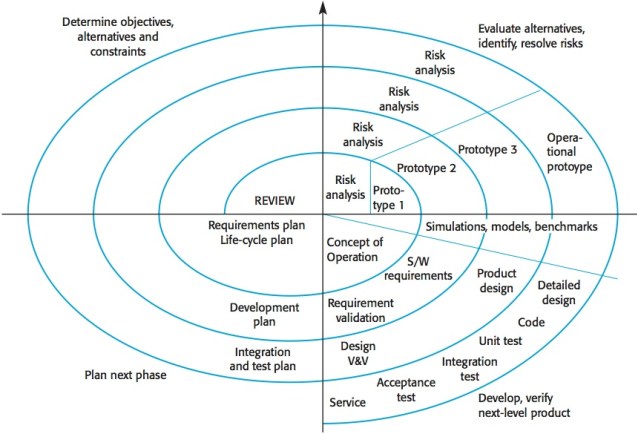
Resmi olmayan bir ürün fikri ile başlar. Yazılım hazır olana kadar müşteri isteği ile sürekli güncellenerek ilerler. Küçük projelerde veya prototiplerde kullanılabilir. Dezavantajları ise; kaynak planlaması yoktur, hatalar kolay bulunmaz ve düzeltilmesi zaman alır, kodların yeniden düzenlenmesi yüksek maliyetli olabilir.



1. Helezonik (Spiral/Sarmal) Model

Bu model, şelale modelindeki gibi düz (lineer) bir süreç yerine **yinelemeli (iteratif)** bir süreç içerir. Her yinelemenin amacı, önceki yinelemeden yararlanarak birikimli bir şekilde ilerleyip iş değerini arttırmaktır. Modelin 4 ana aşaması vardır. Bunlar **Planlama**, **Risk** **Analizi**, **Üretim**, **Kullanıcı Değerlendirme** aşamalarıdır. Planlama aşamasında, hedef belirleme ve önceki ürün ile yeni oluşturulacak ürünün entegresi yapılır. Risk Analizi aşamasında, süreç ve seçenekler göz önünde tutularak riskler değerlendirilir. Üretim aşamasında, yazılım ürünü üzerinde çalışmalar yapılır. Kullanıcı Değerlendirme aşamasında, müşteri ürünü kullanıp test eder ve geri bildirimlerde bulunur.

Bu model oldukça çağdaştır. Hataların erken tespiti, müşteriden erken dönüt alabilme gibi avantajları vardır. Fakat spiralin sonsuza kadar gidebilme gibi bir dezavantajı da vardır. Bunun yanı sıra, karışık(kompleks) bir modeldir ve çok adımdan oluştuğu için doküman kalabalığı oluşur. Büyük projelerde tercih edilmelidir, küçük projelerde çok maliyetli olabilir.



1. Çevik (Agile) Modeller

Agile Modeli; yazılım geliştirme metodu değil, yazılım geliştirme sürecidir. Bu süreç, **yinelemeli (iteratif)** geliştirmeye dayalıdır. Doküman üretme veya yazılım sürecini ayrıntılı planlama yerine yüz yüze ve sık sık yapılan toplantılar içerir. Bu toplantılar sayesinde:

-Fikirler ve düşünceler daha hızlı etkileşime girer.

-Müşterinin isteklerine daha kısa sürede dönüt verilir.

-Hız ve verim sayesinde maliyet düşük olur.

-Hataların ve yapılacak değişikliklerin yazılıma uyarlanması daha kolaydır.

-Personellerin üretme motivasyonu fazladır.

-Takım çalışması yapılmış olunur.

Bazı Agile süreci modelleri:

-Sınırsal Programlama (Extreme Programming-XP)

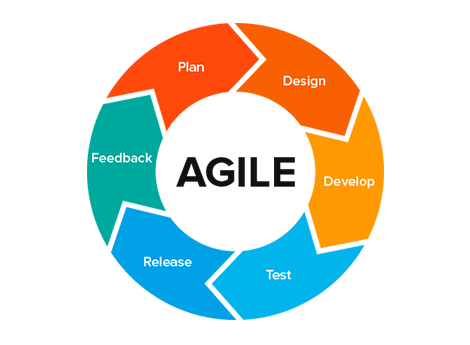
-Çevik Birleştirilmiş Süreç (Agile Unified Process)

-Scrum

-Test Güdümlü Geliştirme (Test-Driven Development)

-Çevik Bilgi Metodu (Agile Data Method)

-Özellik Güdümlü Geliştirme (Feature-Driven Programming)



Yazılım Yaşam Döngü Modellerinin Karşılaştırıldığı Tablo

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Gereksinim  Belirleme | Maliyet | Başarı  Garantisi | Uzmanlık  Gerekliliği | Maliyet  Kontrolü | Basitlik | Risk | Esneklik | Değişiklik Yapma | Yeniden  Kullanılabilirlik | Zaman | Uygulama |
| Kodla ve Düzelt | Başlangıçta | Düşük | Düşük | Düşük | Yok | Basit | Yüksek | Katı | Kolay | Yok | Çok  Uzun | Kolay |
| Çağlayan | Başlangıçta | Yüksek | Düşük | Orta | Var | Basit | Yüksek | Katı | Zor | Yok | Çok Uzun | Kolay |
| V | Başlangıçta | Yüksek | Orta | Orta | Var | Orta | Düşük | Esnek | Zor | Yok | Uzun | Kolay |
| Spiral | Belirli  Aralıklarla | Yüksek | Yüksek | Yüksek | Var | Karmaşık | Düşük | Çok Esnek | Kolay | Var | Uzun | Karmaşık |
| Artırımlı Süreç | Belirli Aralıklarla | Düşük | Yüksek | Orta | Yok | Orta | Düşük | Çok Esnek | Kolay | Var | Uzun | Kolay |
| Çevik | Belirli Aralıklarla | Çok Yüksek | Çok Yüksek | Çok Yüksek | Var | Karmaşık | Düşük | Çok Esnek | Zor | Var | Kısa | Kolay |

Sonuç?

Gördüğümüz gibi her Yazılım Yaşam Döngü Modeli’nin kullanıcına özgü avantajları ve dezavantajları vardır. Bu noktada **projenin** büyüklüğüne, karmaşıklık derecesine, risk alınabilirliğine; **proje ekibinin** üye sayısına, kapasitesine; **projeyi gerçekleştirecek kurumun** mali bütçesine, kaynaklarına bakılarak en uygun model tercih edilmelidir.

Günümüzde en yaygın kullanılan Yazılım Yaşam Döngü Modeli, Çevik Model’dir. Hem müşterinin hem geliştirici kurumun isteklerine en rahat karşılık veren model olması sebebiyle günümüzde oldukça tercih edilmektedir. Zamanla teknoloji ilerleyip yazılım ürünleri geliştikçe ve üretim süreçleri karmaşıklaştıkça yeni modeller ortaya çıkmıştır ve çıkmaya da devam edecektir. Şimdilik biz, geçmişten günümüze kadar ortaya çıkmış ve yaygın olarak kullanılmış modelleri inceledik.

SCRUM

Scrum bir yönetim sürecidir.

Scrum, Çevik Modeller arasında en çok tercih edilen metottur. **Sprint** denilen daha kısa zamanlı alt süreçlerden oluşur. **Scrum Master (Scrum Yöneticisi)** seçilen kişi süreci yönetir, takımların motivasyonunu sağlar ve **Product Owner (Ürün Sahibi)** ile etkileşimde kalır. Yazılımın üretilmesi için gerekli olan işlerin olduğu listeye **Product Backlog** denir. **Sprint Backlog** ise o an içinde bulunulan Sprint’te yapılacak olan işlerin listesidir.

Şimdi de Scrum’ın özellikle günümüzde neden bu kadar popüler olduğuna bakalım:

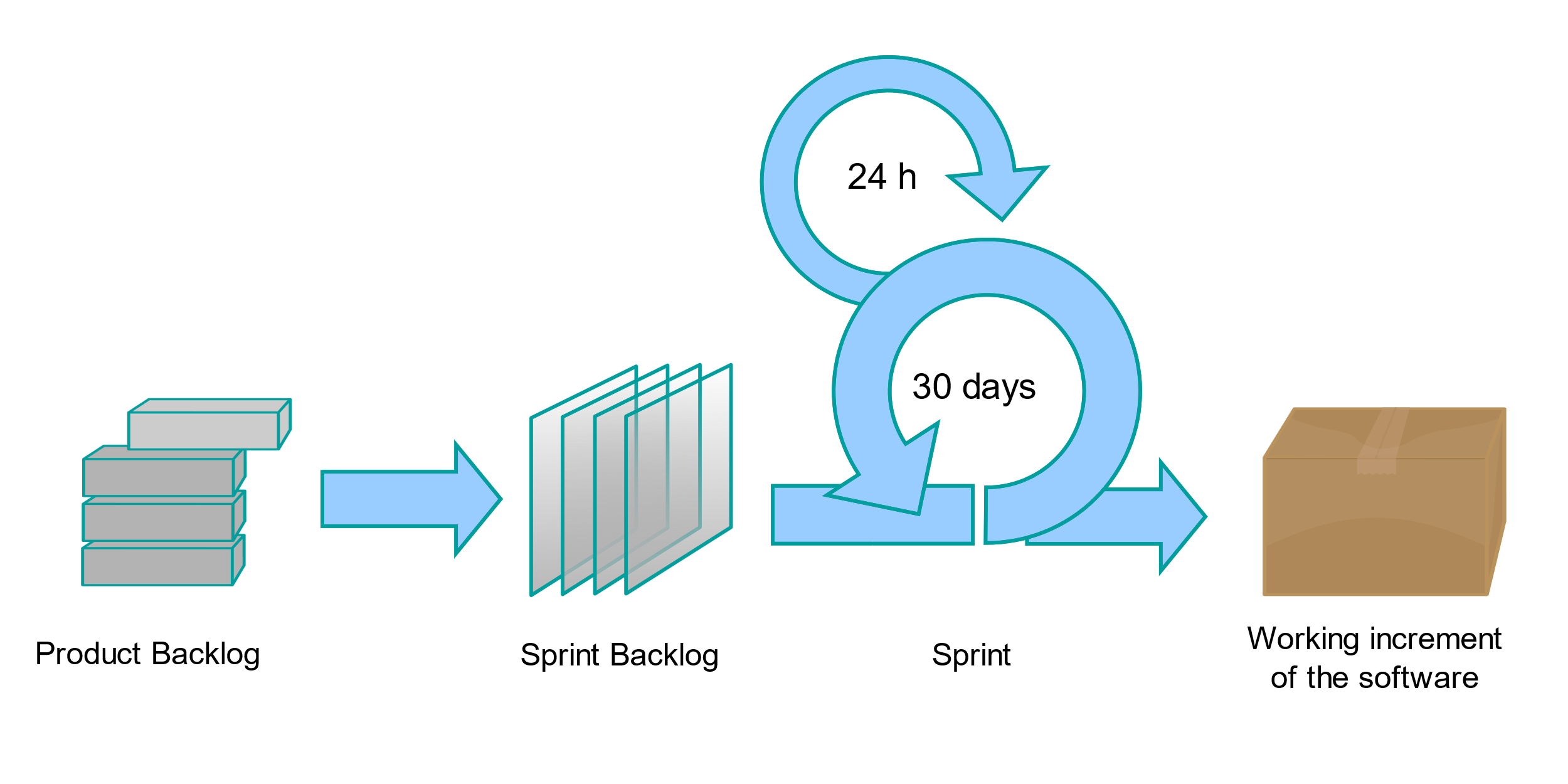
**Şeffaflık:** İlerlemenin ve sorunların **kütüphane kodlaması** üzerinde tutularak herkes tarafından gözlenebilir olmasıdır. Hem zamandan tasarruf hem daha iyi sonuç elde edilmesini sağlar.

**Kısa Sürede Çok İş:** Ar-Ge çalışmaları 1 gün ile 30 gün arası sürer ve bu süreç içinde her gün 15-30 dakika arası toplantılar yapılır.

**Müşteriye Yönelik:** Müşteriyle kağıt üzerinde anlaşılmasındansa, müşteri ile birlikte çalışmayı amaçlar. Müşterinin istek ve tercihlerine hızlı geri dönüş yapılır.

**Iterative(Yineleme):** Ürün geliştirilirken yinelemeli bir süreç tercih edilir. Her yineleme adımında, önceki sürümün üzerine katkılar yapılarak yeni ve daha iyi bir sürüm elde edilir.

**Refactoring(Yeniden Üretme):** Refactoring, yinelemeli süreç ile birlikte daha rahat kullanılabilir olmaktadır. Her yineleme adımında, kodlar istenilen şekilde yazılıp refactoring yapılır. Böylece Şelale Modeli’ndeki gibi proje tamamlandıktan sonra “Keşke şöyle olsaydı” demeye gerek kalmadan vaktinde düzenleme yapılır.



Kaynaklar:

<https://fikirjeneratoru.com/yazilim-proje-yonetimi-yontemleri/>

<https://e-bergi.com/y/cevik-modelleme-ve-cevik-yazilim-gelistirme/>

<https://webdosya.csb.gov.tr/db/cbs/icerikler/salihsoylu_tez_v10-20180925134450.pdf>

<https://systemanalize.wordpress.com/gelistirme-metotolari/#:~:text=Art%C4%B1r%C4%B1msal(Incremental)%20Geli%C5%9Ftirme%20Modeli&text=Proje%20s%C3%BCresi%2C%20d%C3%B6ng%C3%BC%20olarak%20tan%C4%B1mlanan,sayede%20yaz%C4%B1l%C4%B1m%20art%C4%B1r%C4%B1msal%20olarak%20geli%C5%9Ftirilir>.

<https://www.mshowto.org/agile-ve-scrum-nedir.html#close>

<https://bilginc.com/tr/blog/184/scrum-nedir>

<http://www.aspmvcnet.com/tr/m/yazilim-muhendisligi/yinelemeli-model-iterative-model.html>

<https://stratejico.com/scrum-nedir-scrum-master-ne-yapar>

<https://www.burakavci.com.tr/2016/11/scrum-sprint.html>

<http://goktugturan.com/blog/product-backlog-nedir/#:~:text=Product%20Backlog%20ile%20Sprint%20Backlog,ekibin%20beraber%20olu%C5%9Fturdu%C4%9Fu%20bir%20listedir>.

Medium’da makalem: <https://ozdamarsevval.medium.com/yaz%C4%B1l%C4%B1m-ya%C5%9Fam-d%C3%B6ng%C3%BCs%C3%BC-nedir-582cc82c1bdd>

**AD-SOYAD:** ŞEVVAL ÖZDAMAR

**BÖLÜM:** BİLGİSAYAR MÜHENDİSLİĞİ

**ÖĞRENCİ NUMARASI:** 200601044