

YAZILIM GELİŞTİRMEDE ÇEVİK YÖNTEMLER

ÖZELLİK GÜDÜMLÜ GELİŞTİRME
(FEATURE DRIVEN DEVELOPMENT)

Özellik GÜdÜmlÜ Geliřtirme (Feature Driven Development - FDD)

- 1995 yılında Singapur'da 50 kiřilik yazılım geliřtirme ekinin 15 ayda tamamladıđı bir proje için Jeff De Luca tarafından tasarlanmıřtır.
- Amacı tekrarlı bir řekilde alıřan yazılımın mÜřteriye teslimatıdır.
- Yinelemeli ve artırımlı bir evik yazılım geliřtirme modelidir.
- Büyük projelerde kullanılmak Üzere tasarlanmıřtır.
- Diđer evik metodolojilerin en iyi pratiklerinin ođunu birleřtirir.
- Özellik odaklı geliřtirme, büyük geliřtirme takımlarına sahip, önceden tanımlanmıř standartları izleyen ve hızlı sürümler gerektiren projeler için idealdir.

FDD Roller

- Özellik odaklı bir geliştirme takımında altı temel rol yer almaktadır. Bu rolleri destekleyici alt roller de tanımlanabilir.
- Her rol, geliştirme süreci boyunca belirli bir işlevi yerine getirir, bir ekipte belirli bir rolde birden fazla kişi olabilir.
- **Proje Yöneticisi(Project Manager):** Tüm projenin lideridir ve tüm parçaları koordine eder, son tarihlerin tutturulmasını sağlar, iş akışındaki boşlukları belirler, vb.
- **Baş Mimar(Chief Architect):** Genel sistem için bir plan oluşturur. İşlerinin bir parçası, ekipteki insanları sistemin tasarımı hakkında eğitmek ve böylece her bir kişinin bireysel görevlerini tüm projenin bağlamı içinde etkili bir şekilde yerine getirebilmesini sağlamaktır. Baş Mimar, projeye bütünsel bir bakış açısıyla yaklaşır.

FDD Roller

- **Geliştirme Yöneticisi(Development Manager):** Tüm takımları koordine ederek görevlerini zamanında tamamlamalarını sağlar, programlama faaliyetlerinin mentörlüğünü ve liderliğini sağlar.
- **Baş Programcı(Chief Programmer):** Küçük bir geliştirme takımına liderlik eden, projenin doğru yönde ilerlemesini sağlamak için analiz ve tasarıma yardımcı olan deneyimli bir programcıdır.
- **Sınıf Sahipleri(Class Owners):** Daha küçük geliştirme ekiplerinde özellikler oluşturan bireysel geliştiricilerdir. Sorumlulukları arasında özellikleri veya sınıfları tasarlamak, kodlamak, test etmek ve belgelemek yer alabilir.
- **Alan Uzmanı(Domain Expert):** Kullanıcı gereksinimleri hakkında ayrıntılı bilgiye sahiptir ve müşterilerin çözülmesini istediği sorunu anlar.

FDD Pratikleri



Alan nesne modellemesi

- Sorun alanını anlayarak ve sorunu çözen özelliklerle çerçeveyi geliştirmeye yardımcı olan yüksek seviyeli ardışık diyagramlar oluşturarak başlanır.
- Bu modeli oluşturmaya alan uzmanlarını ve gerçek kullanıcıları dahil etmek varsayımları ortadan kaldırır ve hedef kullanıcılar için en alakalı çözümün geliştirilmesini sağlar.

Özelliğe göre geliştirme

- Her özellik, kullanıcıların hedeflerine ulaşmasını sağlayan belirli bir eylem akışını temsil eder.
- Özelliklerin 2 ile 10 gün içinde tamamlanması karmaşık hale geliyorsa, bunları birden fazla alt özelliğe bölmek çok önemlidir.
- Bu, geliştirme ekiplerinin ideal zaman dilimleri içinde özellikleri verimli bir şekilde üretmesini sağlar.

Bireysel kod sahipliği

- Kod/Sınıf sahipliği, takım üyeleri arasında aşılması gereken bir özelliktir.
- Fonksiyonun küçük özelliklere ayrıştırılmasından sonra, bir özelliğin bir geliştiriciye atanması gerçekleşir.
- Özellik sahipliğinin yanı sıra, sınıf sahipliği de vardır. Bu da, her geliştiriciye bir sınıf atandığı ve o geliştiricinin o belirli sınıfın sınıf sahibi olacağı anlamına gelir .
- Bu, değişiklikler geldiğinde işleri kolaylaştırır; böylece ilgili sınıf sahibi hemen devreye girebilir ve sorunsuz bir şekilde yürütebilir.
- Kod sahipliği ayrıca daha yüksek kod kalitesine ulaşmaya yardımcı olur.

Özellik takımları

- Özelliklerin uygulanması birden fazla sınıf geliştirme gerektirir.
- Her sınıfın bir sahibi olduğundan, özellik ekibi tüm bu sınıf geliştiricilerinden oluşur.
- Özellik sahibi, bu sınıf sahiplerine liderlik etmesi gereken bir liderdir.
- Genellikle 3 ile 6 kişiden oluşan küçük ve dinamik özellik takımları oluşturmak iyidir.
- Bunlar özelliğin arkasındaki koda toplu olarak hak sahibidir.
- Baş programcıların rehberliği ve iş birliği, masaya en verimli çözümleri getirmeleri için etkili olabilir.

Denetimler

- Herhangi bir özelliği geliştirdikten sonra, kaliteyi kontrol etmek çok önemlidir.
- Tasarımın, kodun ve özelliğin kalitesini sağlamak için denetimler yapılır.
- Denetim uygulamaları, kaldırılacak potansiyel kusurları ortaya çıkarır ve genel modeli iyileştirmek için öngörüler toplar.
- Hem takım üyeleri hem de baş programcılar, projenin karmaşıklığına bağlı olarak denetim faaliyetlerine katılabilir.

Düzenli yapılar

- Özelliklerin tutarlı bir şekilde çalışmasını ve uygulanmasını sağlar.
- Müşterinin proje boyunca en son, doğru ve sık ilerlemeyi bilmesi için güncel olması gerekir.
- Geliştirme ekibinin her zaman kanıtlanabilir bir sisteme sahip olmasını sağlar.
- Verimli dokümantasyonla düzenli yapı uygulamaları, son kullanıcılara maksimum değeri göstermeye ve genel model geliştikçe entegrasyon hatalarını en aza indirmeye yardımcı olur.

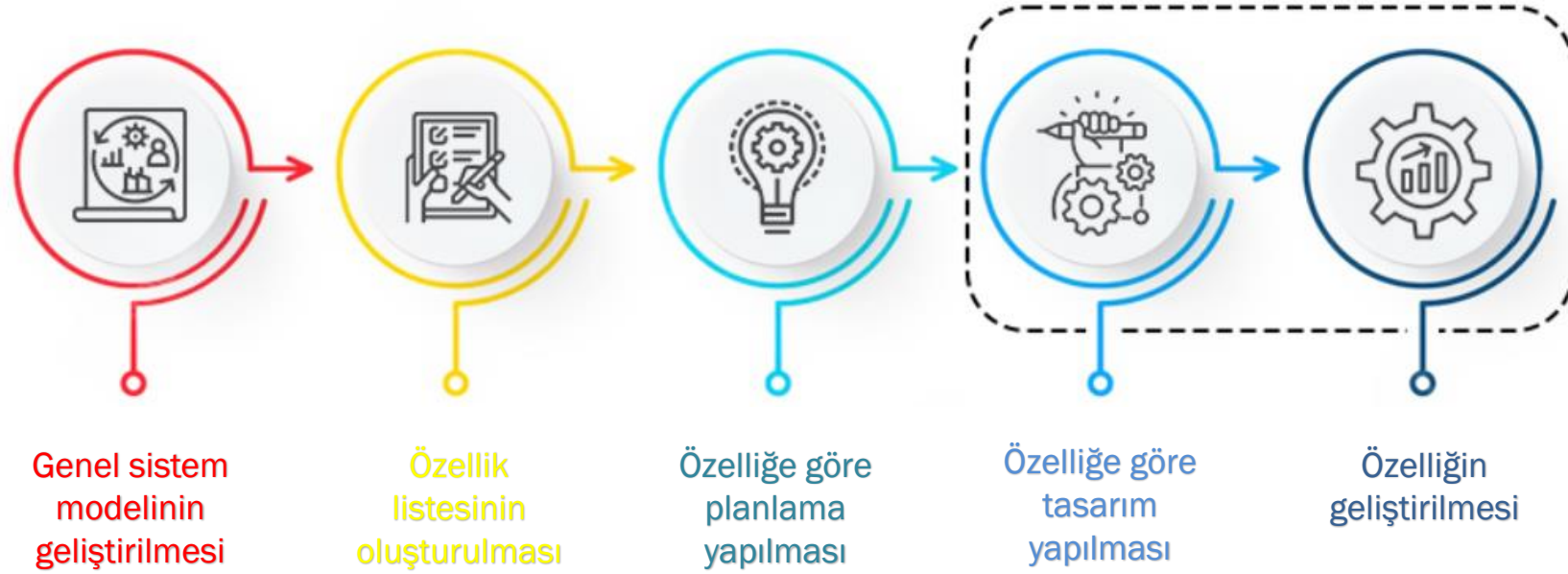
Konfigürasyon yönetimi

- Konfigürasyon yönetim sistemlerini uygulamak, tüm özellikler için kaynak kodunun korunmasına yardımcı olur ve ilgili özellikler için tüm değişiklik geçmişini kaydeder.
- Bir sınıfın geliştirilme sürecinde geçmişini korur.
- Bu, özellik takımlarının özellikleri geliştirmesine ve son kullanıcılar için daha iyi ürünler elde etmesine yardımcı olabilir.

Raporlama

- Yöneticilerin müşterilerle iletişim halinde kalması ve proje ilerlemesinin ve sonuçlarının görünürlüğünü koruması gerekir.
- Verimli raporlama çerçeveleri, proje liderlerinin genel geliştirme ilerlemesini takip edebilmeleri için görünürlüğü artırır.
- Proje takım üyeleri teslimatlar hakkında iyi bilgi sahibi olabilir.

FDD Süreci



Genel sistem modelinin geliştirilmesi

- Özellik odaklı geliştirmenin temel ilkelerinden biri, alan nesne modellemesini kullanmaktır.
- Alan nesne modellemesi, bağlantılı kavramları ve aralarındaki ilişkileri temsil etmenin bir yoludur.
- İlk adımda, alan modelinin bir taslağı oluşturulur. Bu nesne modeli, projenin taslağı olacaktır.
- Genel model, baş mimar veya projeyi yöneten başka bir profesyonel tarafından sistemin kapsamı ve bağlamı belirlenerek oluşturulur.
- Bu aşamada ekip, yazılım geliştirme projesinin genel kapsamını tanımlar.
- Hedef kitleyi, ürünün çözmesi gereken sorunları ve genel bağlamı ana hatlarıyla belirtirler.
- Önerilen modeller incelenir ve bir model veya modellerin bir birleşimi onaylanır.
- Aşamanın sonunda, takım proje hakkında net bir anlayışa ve resme sahip olur.

Özellik listesinin oluşturulması

- Projede geliştirilecek özelliklerin bir listesi oluşturulur.
- Geliştiriciler, kullanıcılar için yararlı olabilecek ve sürüm için belirli bir zaman çizelgesi boyunca tamamlanabilecek olası özelliklerin bir listesini beyin fırtınası yaparak oluştururlar.
- Her özellik yaklaşık iki haftalık bir zaman dilimi içinde yönetilebilir olmalıdır.
- Geliştirilmesi iki haftadan uzun süren özellikler daha küçük özelliklere bölünmelidir.
- Örneğin, oluşturduğunuz özellik "çevrimiçi fatura öde" ise, listedeki özelliklerden bazıları "hesap numarasını doğrula" ve "hesap bakiyesini al" özelliklerini içerebilir.

Özelliğe göre planlama yapılması

- Özellik listesi oluşturulduktan sonra, bunların hangi sırayla geliştirilip uygulanması gerektiği ve her özellikten hangi takım üyesinin sorumlu olduğunu belirleme aşamasıdır.
- Özellikler, oluşturulmalarının ne kadar sürdüğüne ve istemci için ne kadar önemli olduklarına göre sıralanır.
- Tüm takım üyeleri, potansiyel riskleri, bağımlılıkları, iş yükü kısıtlamalarını ve diğer engelleri belirlemek için bu sürece katılmalıdır.
- Bu aşamanın sonunda, her özellik bir geliştiriciye ve özellik takımına atanmalıdır.

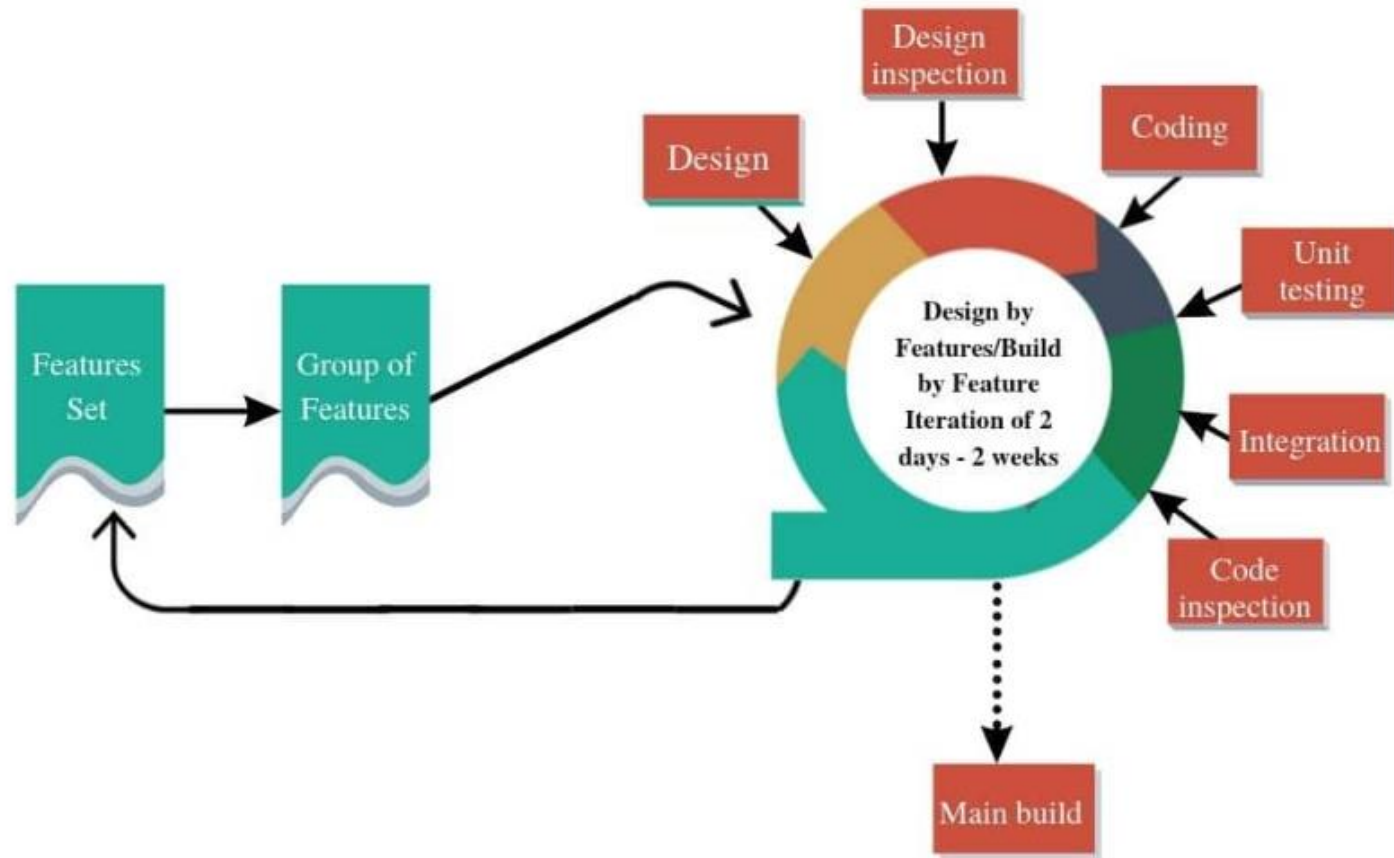
Özelliğe göre tasarım yapılması

- Her özellik için bir tasarım paketi oluşturulur.
- Baş programcı, her iki haftalık yinelemeye hangi özelliklerin dahil edileceğini belirler ve ayrıca sınıf sahiplerini ve özellik önceliklerini belirler.
- Tasarım aşaması, tüm ekip tarafından yapılan bir tasarım incelemesiyle sona erer.
- Alan uzmanı, oluşturulan şeyin çözülen müşteri sorununu ele aldığından emin olur.
- Her özelliğin ayrıntıları üretilir, incelenir ve sonlandırılır.


Özellğin geliştirilmesi

- Tasarım onaylandıktan sonra, artık kod yazmaya başlanır.
- Özellik tasarımını desteklemek için gereken tüm öğeler uygulanır. Bu öğeler, kullanıcı ara yüzleri veya veri tabanı sorguları gibi öğeleri içerebilir.
- Her şey takım üyeleri tarafından tamamlandıktan sonra özellik test aşamasına geçer ve süreç listedeki bir sonraki özellik için tekrar başlar.
- Tasarım iyileştirildikten sonra, tamamlanan özellik müşteriye teslim edilmek üzere resmi yapıya eklenir.

FDD



FDD Avantajları

- 
- Gerçek son kullanıcıyı dahil etme fikri, daha kullanıcı merkezli ürün geliştirmeyi garanti eder.
 - Alan modelleme ve planlama, öngörülebilir ve kontrollü bir geliştirme yol haritası sağlar.
 - İyi tanımlanmış adımlara sahip yapılandırılmış yaklaşım, daha fazla ölçeklenebilirlik sunarak onu büyük ölçekli ve karmaşık projeler için ideal hale getirir.
 - Etkili ve net dokümantasyona vurgu, geliştirme ekipleri için gelişmiş görünürlük, üretkenlik ve verimlilikle birleşir.
 - Proje gereksinimlerini küçük ve yürütülebilir görevlere bölmek, geliştirme döngüsü boyunca kodu yönetmeyi kolaylaştırır.

FDD Dezavantajları



- Karmaşık takım yapısı onu küçük projeler için daha az uygun hale getirir.



- Takım oluşturma ve eğitim, zaman ve bütçe açısından bir zorluk olabilir.



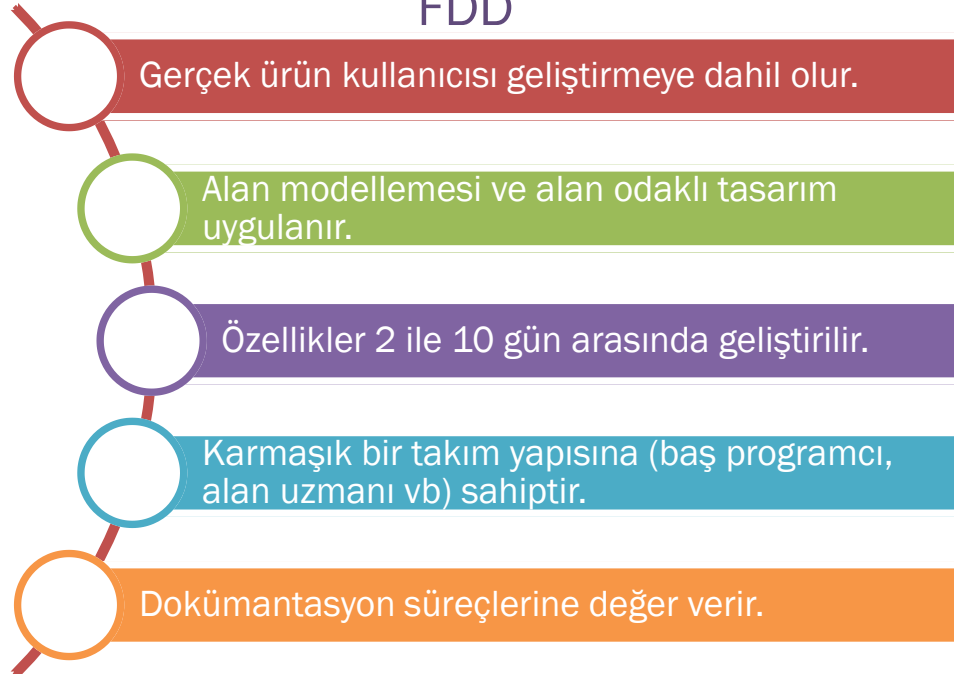
- Bağımlılıklar, teknolojik uzmanlık ve birbiriyle ilişkili özellikler geliştirme yaşam döngüsünü engelleyebilir.



- Baştan itibaren iyi planlanmış bir yaklaşımı vurguladığı için, devam eden özellik değişiklikleri yapmak için yalnızca biraz esneklik sunar.

FDD – Scrum Karşılaştırma

FDD



SCRUM

