Bölüm 8

Yazılımın Sınanması (Testing)

İçerik

- Geliştirme testleri
- Test güdümlü geliştirme
- Sürüm testleri
- Kullanıcı testleri

Program Testi

- Testler, programın kendinden yapması beklenenleri yaptığını göstermek ve kullanılmaya başlamadan önce hatalarını saptamak amacıyla yapılır
- Yazılımı test ederken, program yapay verilerle çalıştırılır
- Test sonuçları hatalar ve anormalliklerin tespiti veya sistemin fonksiyonel olmayan özellikleri hakkında bilgi almak için kullanılır
- Hata varsa ortaya çıkar, yoksa çıkmaz.
- Test, büyük bir doğrulama ve onay sürecinin bir parçasıdır

Program Test Hedefleri

- Yazılım geliştiriciye ve müşteriye yazılımın gereksinimleri karşıladığını göstermek
  - Kişiye özel yazılım için bu, gereksinim dokümanındaki her gereksinim için en az bir test yapılması anlamına gelir. Genel yazılım ürünleri için ise sistem özelliklerinin hepsi ve özelliklerin kombinasyonları için testler olması anlamındadır
- Yazılımın davranışının yanlış, istenmeyen şekilde veya tanımını karşılamadığı durumları tespit etmek
  - Hata testleri, sistemin göçmesi, diğer sistemlerle istenmeyen etkileşimlere girmesi, yanlış hesaplamalar ve veri bozulması gibi hataların ortaya çıkarılması ve kökünden yok edilmesi ile ilgilenir

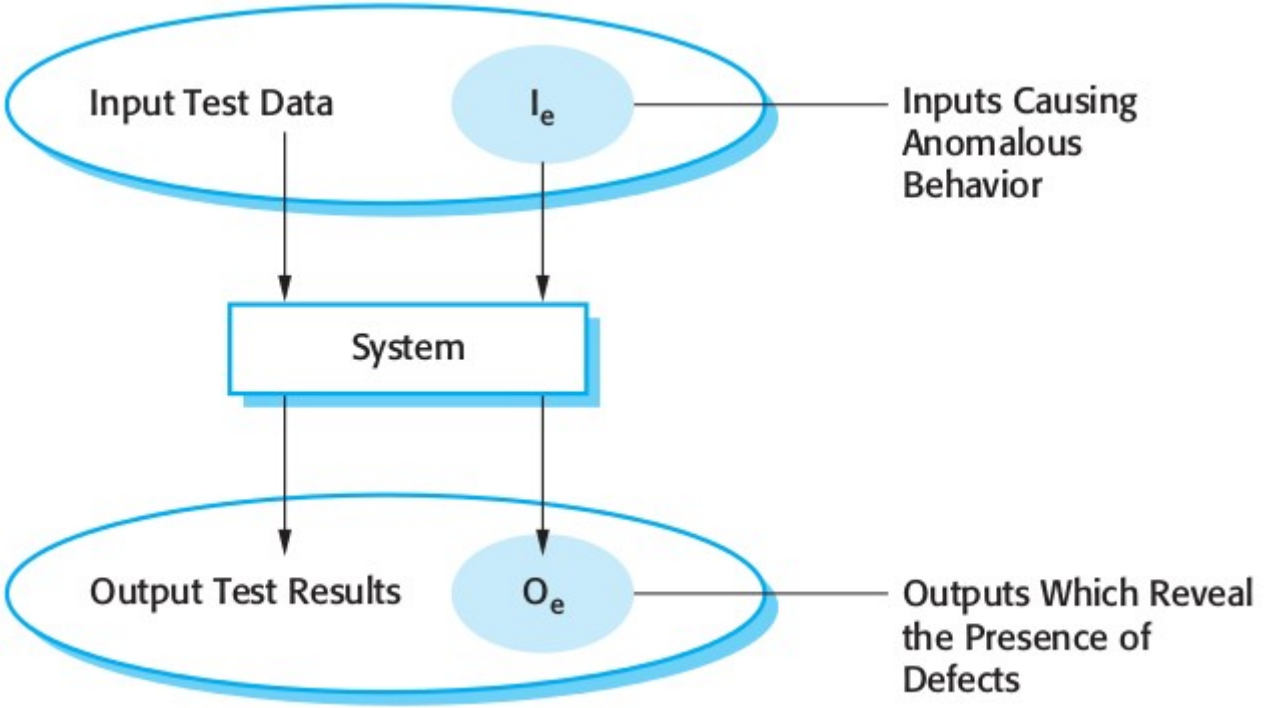
Geçerlilik ve Hata Testi

- İlk hedef bizi geçerlilik testine götürür
  - Sistemin beklenen kullanımını yansıtan bir dizi test karşısında sistemin doğru çalışması beklenir
- İkinci hedef bizi hata testine götürür
  - Testler hataları ortaya çıkarmak için tasarlanır. Hata testindeki testler kasıtlı olarak anlaşılmaz olabilir, sistemin normal kullanımını yansıtmak zorunda değildir

Test Süreci Hedefleri

- Geçerlilik testleri
  - Yazılım geliştiriciye ve müşteriye yazılımın gereksinimleri karşıladığını göstermek
  - Başarılı bir test sistemin beklendiği gibi çalıştığını gösterir
- Hata testleri
  - Yazılımın davranışının yanlış olduğu veya program tanımına uymadığı durumları tespit etmek
  - Başarılı bir test sistemin yanlış çalışmasına sebep olan testtir, böylece sistemin kusurları ortaya çıkar

Program Testinin Girdi-Çıktı Modeli



Gerçekleme ve Doğrulama

- Gerçekleme:
  - "Ürünü doğru üretiyor muyuz?”
  - Yazılım, tanımına uygun olarak üretilmelidir
- Doğrulama:
  - "Doğru ürünü mü üretiyoruz?”
  - Yazılım, kullanıcının ihtiyacı neyse onu yapmalıdır

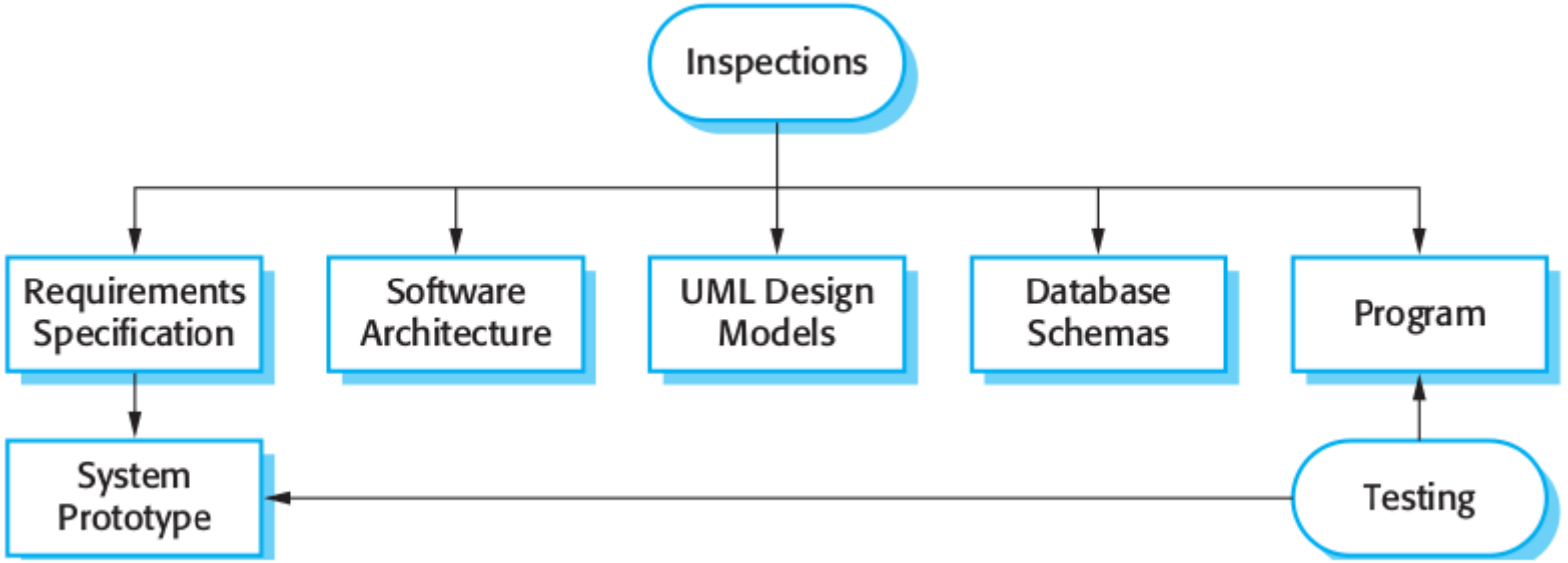
V&V (Gerçekleme ve Doğrulama) Güvenilirliği

- V & V'nin amacı sistemin amacına uygun olduğuna dair güveni oluşturmaktır
- Sistemin amacı, kullanıcı beklentileri ve pazara bağlıdır
- Yazılımın amacı
  - Güven düzeyi yazılımın bir kurum için ne kadar önemli olduğu ile ilgilidir
- Kullanıcı beklentileri
  - Kullanıcılar belirli uygulama türlerine karşı düşük beklenti içinde olabilirler
- Pazar
  - Bir ürünü erken pazara çıkarmak programın hatalarını bulmaktan daha önemli olabilir

İnceleme ve Test

- Yazılımı inceleme Hataları keşfetmek için sistemin analizini yapma ile ilgilidir (statik gerçekleme)
  - Kod analizi ve araç-tabanlı belgeleme ile desteklenebilir
- Yazılım Testi Ürün davranışını gözlemleme ile ilgilidir (dinamik gerçekleme)
  - Sistem test verisi ile çalıştırılır ve davranışı gözlemlenir

İnceleme ve Test



Yazılım İnceleme

- Hataların ve anormalliklerin belirlenmesi için gereksinimlerin, analizin, tasarımın, kaynak kodun ve hatta testlerin insanlar tarafından incelenmesini kapsar
- İnceleme yapmak için sistemin çalışıyor olması gerekmez; bu yüzden gerçekleştirime geçmeden önce de yapılabilir
- Programdaki hataların ortaya çıkarılmasında etkin bir yöntem olarak bilinir

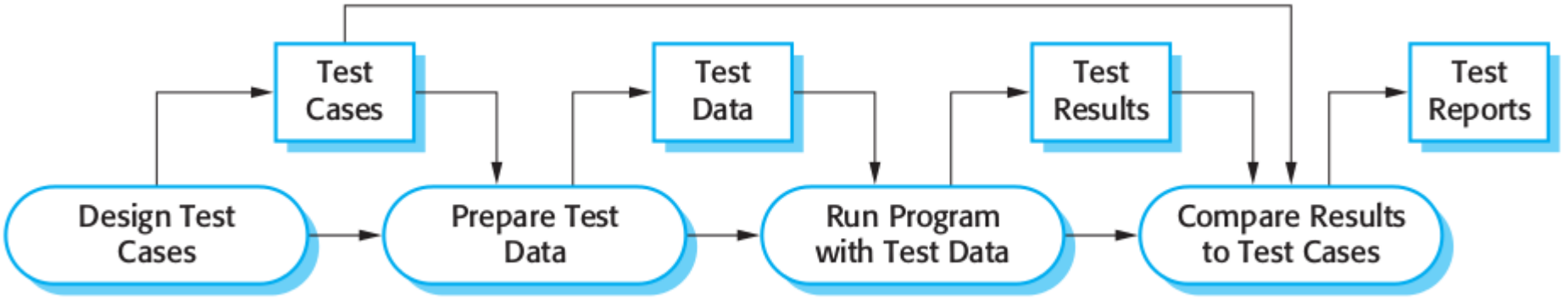
İncelemenin avantajları

- Testler esnasında, hatalar başka hataları gizleyebilir. İnceleme durağan bir süreç olduğundan hatalar arasındaki ilişkilerle uğraşmak zorunda kalmayız
- Sistemin tamamlanmamış sürümleri herhangi bir ek maliyet gerektirmeden incelenebilir. Eğer bir program henüz tamamlanmamışsa, uygun olan kısımlarını test edebilmek için özel testler uygulanabilir
- Programın hatalarının yanında, programın yazım standartları, taşınabilirlik gibi kalite özelliklerine uyup uymadığını da inceleme ile görebiliriz

İnceleme ve Test

- İnceleme ve test, birbirini tamamlarlar; birbirinin zıttı olan gerçekleme teknikleri değillerdir
- V & V sürecinde ikisi de kullanılmalıdır
- İnceleme ile yazılımın tanımına uygunluğu kontrol edilebilir ancak müşterinin gerçek taleplerine uygunluğu kontrol edilemez
- Performans, kullanılabilirlik gibi fonksiyonel olmayan sistem gereksinimleri inceleme ile kontrol edilemez

Yazılım Test Sürecinin Bir Modeli



Test Adımları

- Geliştirme testleri - sistem kodlanırken kusurları ve bugları bulmak için yapılan testler
- Sürüm testleri – Kullanıcılara sunulmadan önce ayrı bir test takımının sistemin sürümünün tamamı üzerinde yaptığı testler
- Kullanıcı testleri – Kullanıcılar veya potansiyel kullanıcıların sistemi kendi ortamlarında test etmeleri

Geliştirme Testleri

- Sistemi geliştiren takım tarafından yapılan geliştirme testleri
  - Birim testi – Program birimlerinin veya nesne sınıflarının ayrı ayrı test edilmesidir. Nesnelerin ve metotların işlevselliklerinin test edilmesine odaklanmalıdır
  - Bileşen testi – Bağımsız birimler birleşerek bileşenleri oluştururlar. Bileşen testi bileşen arayüzlerinin test edilmesine odaklanmalıdır
  - Sistem testi – Bileşenler birleşerek sistemi oluştururlar ve sistem bir bütün olarak test edilir. Sistem testleri bileşenler arasındaki etkileşimlerin test edilmesine odaklanmalıdır

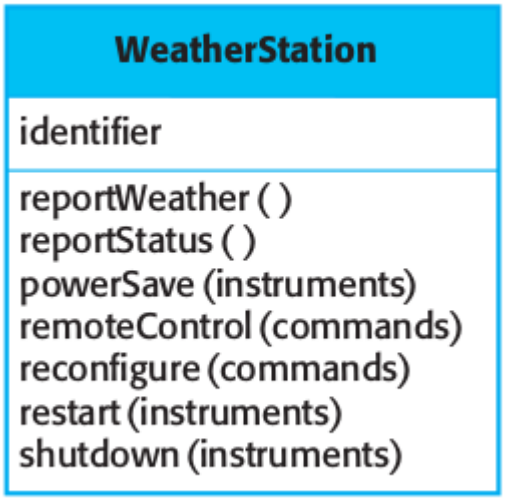
Birim Testleri

- Birim testi tekil birimlerin sistemden izole edilerek test edilmesi sürecidir
- Bir hata testi sürecidir
- Birimler şunlar olabilir:
  - Bir nesnedeki fonksiyon veya metotlar
  - Birçok özellik ve metottan oluşan nesne sınıfları
  - Miras veya genelleme durumunda nesneler arayüzler veya süpersınıflar ile birlikte

Nesne Sınıfı Testleri

- Bir sınıfın tüm içeriğini test etmek
  - Bir nesne ile ilgili tüm operasyonları test etme
  - Tüm nesne özelliklerini atama ve sorgulama
  - Nesnenin her durumunu kontrol etme
- Miras birim testleri tasarlamayı zorlaştırır

Meteoroloji İstasyonu Nesne Arayüzü



Meteoroloji İstasyonu Testleri

- Bunlar için test tanımlanmalı: reportWeather, calibrate, test, startup ve shutdown
- Bir durum modeli kullanarak, durumlar arası geçişi ve bu geçişe sebep olan olay sırasını belirleriz
- Örneğin:
  - Shutdown -> Running-> Shutdown
  - Configuring-> Running-> Testing -> Transmitting -> Running
  - Running-> Collecting-> Running-> Summarizing -> Transmitting -> Running

Otomatik Testler

- Eğer mümkünse, birim testleri otomatikleşmiş olmalıdır böylece elle müdahale olmadan testler çalışır ve kontrol edilir
- Otomatik birim testinde, testlerinizi yazmak ve çalıştırmak için JUnit gibi bir test çerçevesi kullanırız
- Birim testlerinde kullanılan çerçeveler sizin kendinize özgü testler oluşturmakta kullanabileceğiniz genel test sınıfları sunar

Otomatik Test Bileşenleri

- Kurulum (Setup) - sitemi test durumu ile ilklediğimiz yani girdileri ve beklenen çıktıları verdiğimiz yer
- Çağrı (Call) – test edilecek nesneyi veya metodu çağırdığımız yer
- İddia (Assertion) – çağrı sonunda oluşan çıktıları beklenen çıktılarla karşılaştırdığımız yer. Eğer iddia sonunda true oluşursa test başarılıdır, eğer false oluşursa test başarısızdır

Birim Testlerinin Etkinliği

- Testler, beklendiği gibi kullanıldığında, test edilen bileşenin yapması beklenen işi yaptığını göstermelidir
- Eğer bileşende hatalar varsa testlerle ortaya çıkarılmalıdır
- Burada iki tip birim testi karşımıza çıkıyor:
  - Programın normal operasyonunu gerçekleştirirken bileşenin beklendiği gibi çalıştığını göstermeli
  - Anormal girdilerin düzgünce işlendiğini ve programın çökmesine sebep olmadığını göstermeli (test deneyimine dayanır).

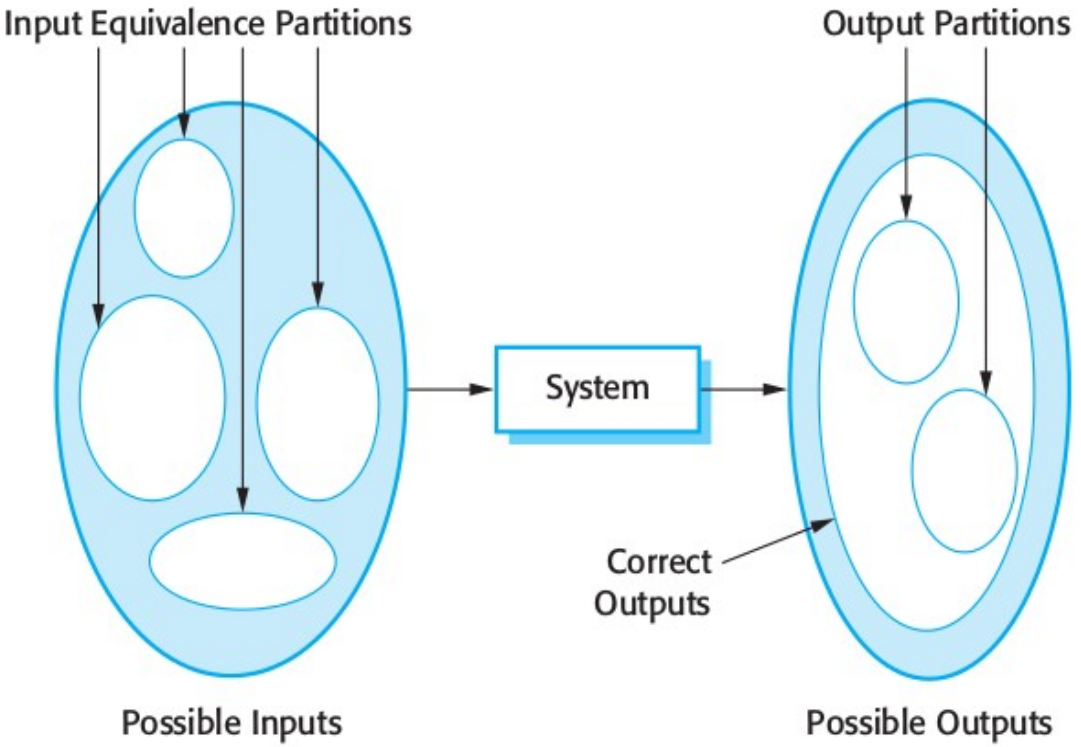
Test Stratejileri

- Parça testi – ortak özellikleri olan girdileri gruplara ayırıp aynı yolla işleme
  - Bu grupların herbiri için hangi testlerin yapılacağına karar verilmelidir
- Kılavuz tabanlı test – test durumlarını seçmek için kılavuz kullanılır
  - Bu kılavuzlar geçmiş deneyimlere ve programcıların genelde yaptığı geliştirme hatalarına dayanırlar

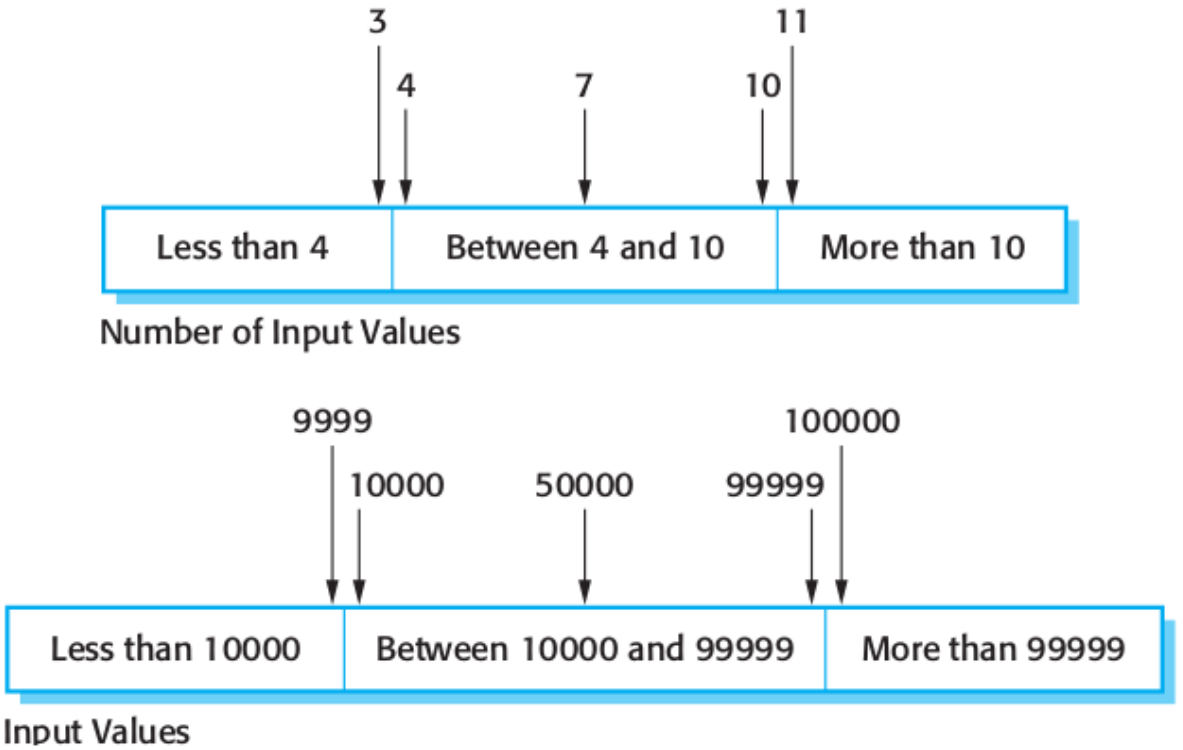
Parça Testleri

- Program herbir sınıf üyesine aynı davranışı gösterdiği için bu sınıflardan her birine denklik sınıfı denir
- Herbir parçadan test durumları seçilmelidir

Denklik Parçaları



Denklik Parçaları



Test Kılavuzu (diziler)

- Tek bir eleman içeren dizilerle yazılımı test et
- Farklı testler için farklı boyutlarda diziler kullan
- Testlerde ilk, son ve orta elemana erişilmesini sağla
- Sıfır uzunluğundaki dizilerle test yap

Genel Test Kılavuzu

- Sistemi tüm hata mesajlarını üretmeye zorlayacak girdiler seç
- Taşmaya sebep olabilecek girdiler oluştur
- Bu tip girdilerle programı defalarca çalıştır
- Geçersiz çıktılar üretmeye zorla
- Çok büyük veya çok küçük sonuç üretecek hesaplamalara zorla

Önemli Noktalar

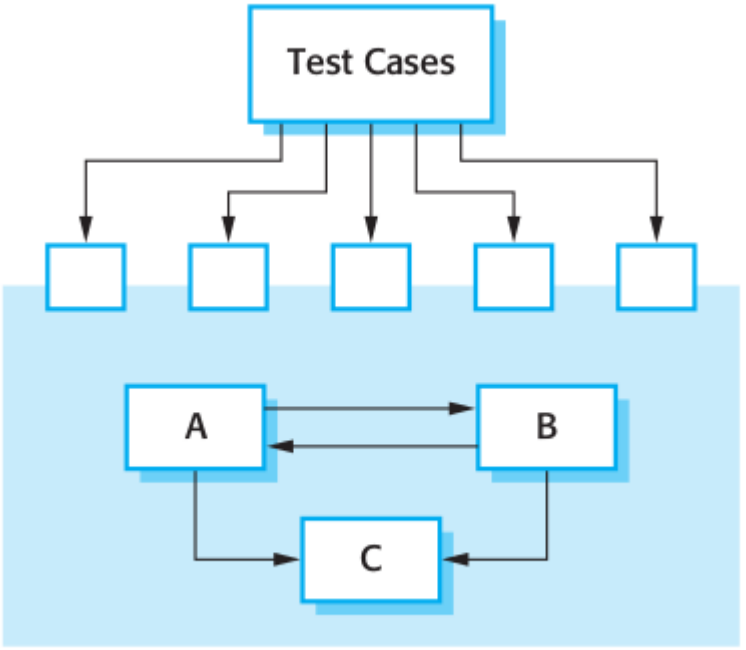
- Testler sadece görünür hataları tespit edebilir. Testler programda hiç hata kalmadığını garanti etmez
- Geliştirme testleri, geliştirme takımının sorumluluğundadır. Farklı bir takım, müşteriye teslim edilmeden önce sistem testlerini yapmalıdır
- Geliştirme testleri nesneleri ve metotları tek başına test ettiğimiz birim testlerini, birbiriyle ilişkili nesne gruplarını test ettiğimiz bileşen testlerini ve sistemin bir kısmını veya tamamını test ettiğimiz sistem testlerini kapsar

Yazılımın Sınanması

Bileşen Testi

- Yazılım bileşenleri genellikle birbirleri ile etkileşimde olan birçok nesneden oluşurlar
  - Örneğin, hava istasyonu sisteminde yeniden yapılandırma bileşeni (reconfiguration) yeniden yapılandırmanın herbir yönü ile ilgilenen nesneleri kapsar
- Bu nesnelerin işlevselliğine tanımlanmış olan bileşen arayüzü aracılığıyla ulaşılır
- Dolayısıyla bileşenlerin testi arayüzün tanımlandığı şekilde davranış gösterdiğini göstermeye odaklanmalıdır
  - Bileşen içindeki birimlerin birim testlerinin tamamlanmış olduğu varsayılabilir

Arayüz Testleri



Arayüz Testleri

- Hedefle arayüzler sebebiyle oluşan hataları ve arayüzlerin geçersiz kullanımlarını saptamaktır
- Arayüz tipleri
  - Parametre arayüzleri – Bir metottan diğerine geçirilen veri
  - Paylaşımlı bellek arayüzleri – Prosedürler veya fonksiyonlar arasında paylaşılan bellek bloğu
  - Prosedürel arayüzler – Başka alt sistemlerden çağrılacak bir grup prosedürü kapsülleyen bir alt sistem
  - Mesaj geçme arayüzleri – Diğer alt sistemlerden servis talep eden alt sistemler

Arayüz Hataları

- Arayüzün yanlış kullanımı
  - Bir bileşen diğerini çağırırken yanlış kullanım yapması; yanlış sırada parametre geçme
- Arayüzün yanlış anlaşılması
  - Çağıran bileşen çağrılan bileşenin davranışı hakkında yanlış varsayımda bulunur; ikili aramayı sıralanmamış dizide yapmak
- Zamanlama hataları
  - Çağıran ve çağrılan bileşenler farklı hızda çalıştıklarından güncel olmayan veriye erişilir; gerçek zamanlı sistemlerde mesaj geçişi veya ortak bellek kullanımında oluşur

Arayüz Test Kılavuzu

- Parametrelerin sınır değerleri ile prosedürleri çağır
- İşaretçi tipi parametreleri null işaretçilerle çağır
- Bileşenin hata oluşturacağı testler tasarla
- Mesaj geçilen sistemlerde stres testi kullan
- Belleğin paylaşımlı kullanıldığı sistemlerde, bileşenlerin farklı sırada aktif oladuğu testler tasarla

Sistem Testleri

- Geliştirme sürecinde yapılan sistem testleri sistemin bir sürümünü oluşturmak için bileşenlerin entegrasyonunu ve entegre edilmiş sistemin testini kapsar
- Sistem testinde odaklanılması gereken bileşenler arasındaki etkileşimlerin test edilmesidir
- Sistem testleri bileşenlerin uyumlu olduğunu, doğru etkileşimde olduğunu ve bileşenlerin arayüzler aracılığıyla veriyi doğru olarak ve doğru zamanda ilettiğini onaylar
- Sistem testi sistemin aniden ortaya çıkan davranışlarını test eder

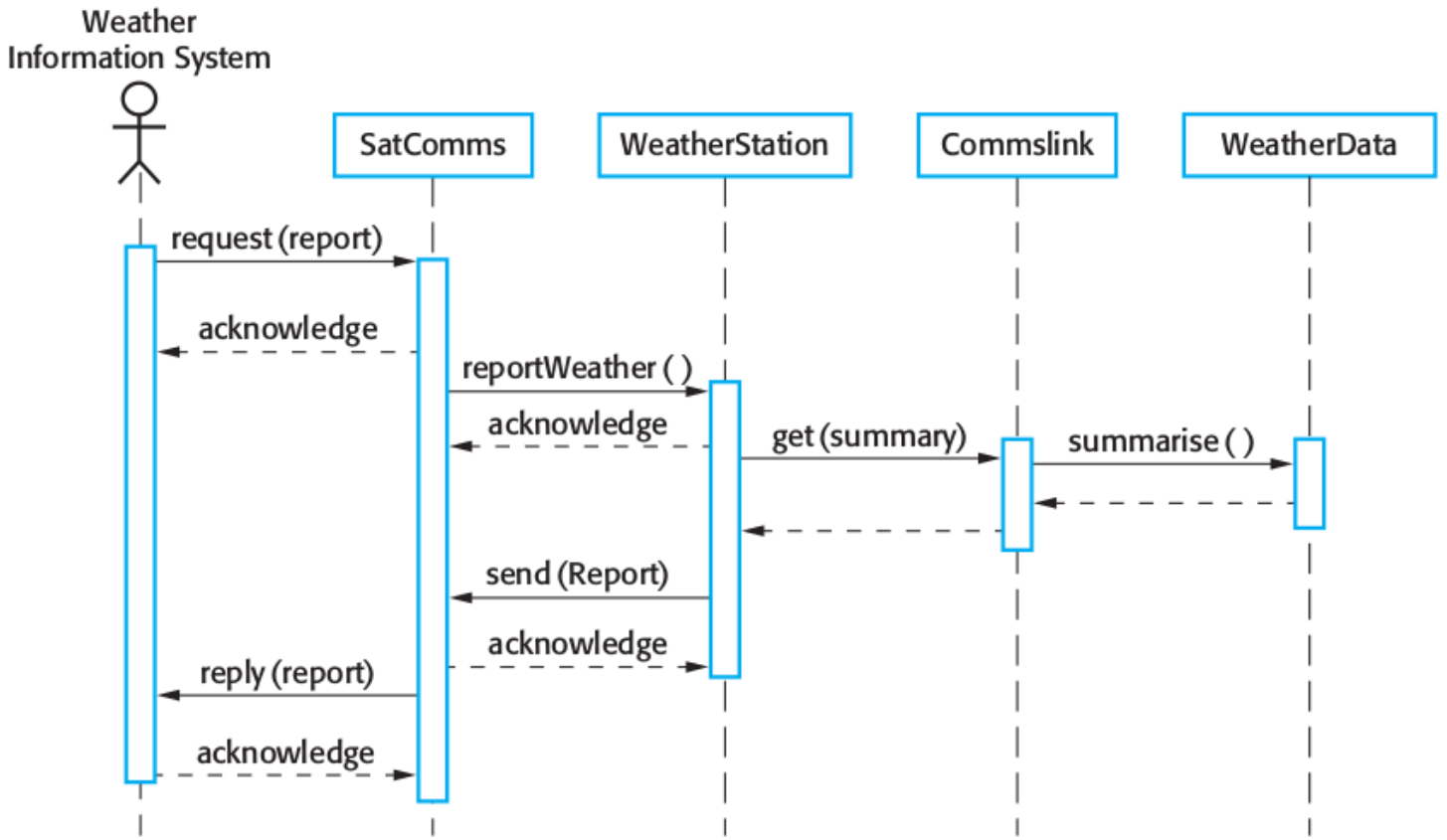
Sistem ve Bileşen Testleri

- Sistem testi esnasında, ayrı olarak geliştirilmiş yeniden kullanılabilir bileşenler veya satın alınmış sistemler yeni geliştirilen sistemle birlştirilebilir. Daha sonra tüm sistem test edilir
- Farklı takım üyeleri veya farklı takımlar tarafından geliştirilmiş bileşenler bu aşamada birleştirilir. Sistem testi tekil değil ortaklaşa bir süreçtir
  - Bazı şirketlerde sistem testleri tasarımcılardan veya geliştiricilerden kimseyi içermeyen ayrı bir takım tarafından gerçekleştirilir

Kullanım Durumu Testleri

- Kullanım durumları sistem testi için temel olarak kullanılabilecek sistem etkileşimlerini tanımlarlar
- Her kullanım durumu birçok sistem bileşeninin etkileşimini kapsar dolayısıyla kullanım durumunu test etmek bu etkileşimleri gerçekleşmeye zorlamak anlamına gelir
- Kullanım durumları ile ilgili sıralama diyagramları, test edilen bileşenleri ve aralarındaki etkileşimi gösterirler

Collectweather sıralama diyagramı



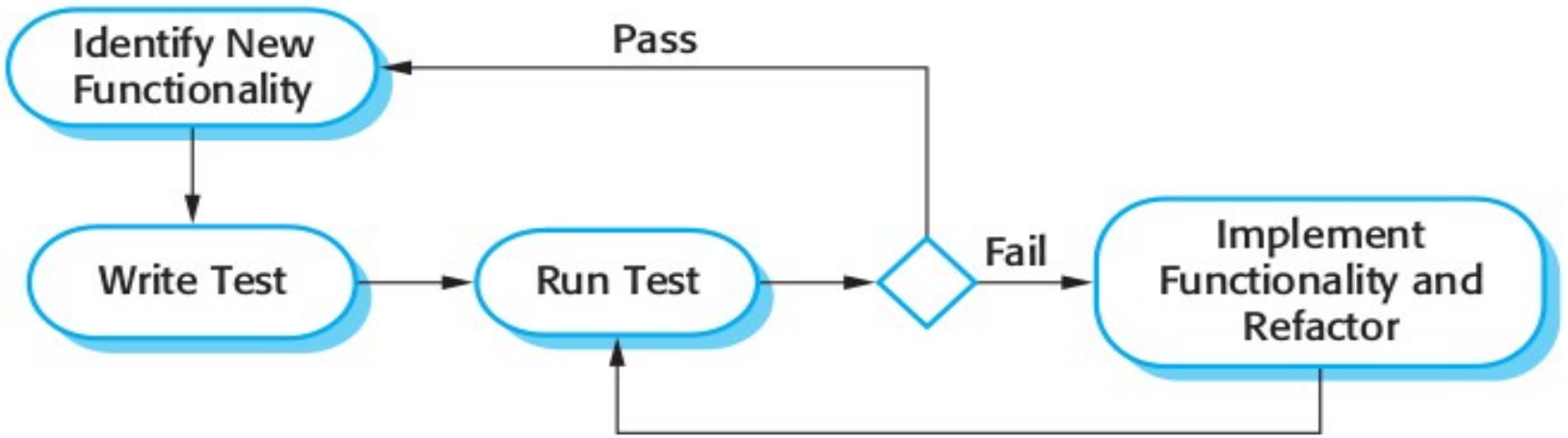
Test İlkeleri

- Çok kapsamlı ve ayrıntılı sistem testi mümkün olmadığından gerekli sistem testlerini kapsayacak test ilkeleri geliştirilebilir
- Test ilkelerine örnekler:
  - Menüler aracılığıyla ulaşılan tüm sistem fonksiyonları test edilmelidir
  - Aynı menü üzerinden ulaşılan fonksiyonların kombinasyonları test edilmelidir
  - Kullanıcı girişinin olduğu biryerde tüm fonksiyonlar hatalı ve doğru girdi ile test edilmelidir

Test Güdümlü Geliştirme

- Test güdümlü geliştirme kodlama ile testlerin birlikte yürütüldüğü bir program geliştirme yaklaşımıdır
- Testler kodlardan önce yazılır ve testleri geçmek geliştirmenin kritik faktörüdür
- Kod o artırımın yanında testi ile birlikte artırımlı olarak geliştirilir. Yazılan kod testleri geçmeden bir sonraki artırıma geçilmez
- TGG, XP gibi çevik metodların bir parçası olarak ortaya çıkmıştır. Fakat aynı zamanda plan güdümlü geliştirme sürecinde de kullanılabilir

Test Güdümlü Geliştirme



Test Güdümlü Geliştirme Süreci İşleri

- Fonksiyonelliğin artırımının tanımı yaparak başlar. Bu genellikle küçük ve birkaç satırda yazılabilecek birşeydir
- Bu fonksiyonellik için bir test oluştur ve bunu otomatik test olarak gerçekle
- Testi diğer bütün testlerle birlikte çalıştır. Henüz gerekli kodu yazmadığın için test hata verir
- Gerekli kodu geliştirip testleri yeniden çalıştır
- Testler başarı ile çalıştırıldıktan sonra diğer fonksiyonellikler için aynı şekilde devam et

Test Güdümlü Yazılımın Yararları

- Kodu kapsama
  - Yazılan her kod parçası bir testle ilişkilendirildiğinden tüm kodun en azından bir testten geçmesi gerekir
- Regression testi
  - Yaptığımız yeni değişimlerin kodda yeni hatalar oluşturmadığından emin olabiliriz
- Basitleştirilmiş debug
  - Bir test hata verdiğinde hatanın nerede oluştuğu hemen tespit edilir ve yeni yazılan kod gözden geçirilir
- Sistem dokümantasyonu
  - Testler kodun yapması gereken işi tanımlayan bir nevi dokümandır

Regression Testleri

- Regression testleri yeni yazılan kodun sistemin çalışmasını bozmadığını kontrol etmek için yapılır
- Testler elle yapılıyorsa regression testi maliyetli olabilir ancak otomatik testlerle çok kolaydır. Programa yapılan her değişiklik için tüm testler yeniden çalıştırılmalıdır
- Değişiklikler onaylanmadan önce testleri geçmiş olmalıdır

Sürüm Testleri

- Sürüm testleri yazılımın geliştirme takımı dışındaki kişilerin kullanımı için yapılır
- Buradaki amaç üreticiyi programın kullanım için yeterince iyi olduğuna ikna etmektir
- Dolayısıyla sürüm testi sistemin tanımlanmış fonksiyonellikleri, performansı ve normal kullanımda hata vermediğini göstermek zorundadır
- Sürüm testleri genellikle sistem tanımından oluşturulan kara-kutu testlerdir

Sürüm Testi ve Sistem Testi

- Sürüm testi sistem testinin bir türüdür
- Önemli farklar:
  - Sürüm testinde, geliştirme takımına dahil olmayan ayrı bir takım testleri yapmalıdır
  - Geliştirme takımının yaptığı sistem testi, sistemdeki hataları bulma amaçlı olmalıdır (hata testi). Sürüm testinin amacı sistemin gereksinimleri karşıladığını ve kullanım için yeterince iyi olduğunu göstermektir (geçerlilik testi).

Gereksinim Tabanlı Test

- Gereksinimlerin her biri için testler oluşturulur
- Psikiyatrik hasta takip sistemi gereksinimleri:
  - Eğer bir hastanın herhangi bir ilaca alerjisi varsa bu ilaç reçetelendiğinde sistem kullanıcısına uyarı mesajı verilmelidir
  - Reçeteleme yapan kişi bu uyarıyı dikkate almazsa neden dikkate almadığına dair bir sebep belirtmek zorundadır

Gereksinim Testleri

- Alerjisi olmayan bir hasta kaydı yap. Alerji yaptığı görülen bir ilacı bu hastaya reçetele. Uyarı mesajının gelmediğini gör
- Alerjisi olan bir hasta kaydı yap. Bu hastanın alerjisi olduğu ilacı ona reçete et ve sistemin uyarı verdiğini gör
- İki veya daha fazla ilaca alerjisi olan bir hasta girişi yap
- İki ilacı ayrı ayrı bu hastaya reçete et ve sistemin ikisi için de doğru şekilde uyarı mesajı verip vermediğini gör
- Hastaya iki alerjik ilacı reçete et ve iki uyarı mesajının da doğru olarak geldiğini gör
- Uyarı mesajı veren bir ilaç reçete et ve bu uyarıyı gözardı et. Sistemin senden açıklama beklediğini gör

Senaryo İle Test Edilen Özellikler

- Sisteme girişlerde yetki kontrolü
- Bir dizüstü bilgisayara belirli bir hastanın kaydını indirmek veya bilgisayardan kayıt yüklemek
- Ev ziyareti zamanlama
- Mobil bir cihazda hasta kayıtlarının şifrelenmesi ve şifre çözülmesi
- Kayıt çekme ve düzenleme
- İlaçların yan etkilerini sağlayan ilaç veritabanıyla bağlantılar
- Çağrı yönlendirme sistemi

Psikiyatrik Takip Sistemi İçin Kullanım Senaryosu

Kate is a nurse who specializes in mental health care. One of her responsibilities is to visit patients at home to check that their treatment is effective and that they are not suffering from medication side -effects. On a day for home visits, Kate logs into the MHC-PMS and uses it to print her schedule of home visits for that day, along with summary information about the patients to be visited. She requests that the records for these patients be downloaded to her laptop. She is prompted for her key phrase to encrypt the records on the laptop.

One of the patients that she visits is Jim, who is being treated with medication for depression. Jim feels that the medication is helping him but believes that it has the side -effect of keeping him awake at night. Kate looks up Jim’s record and is prompted for her key phrase to decrypt the record. She checks the drug prescribed and queries its side effects. Sleeplessness is a known side effect so she notes the problem in Jim’s record and suggests that he visits the clinic to have his medication changed. He agrees so Kate enters a prompt to call him when she gets back to the clinic to make an appointment with a physician. She ends the consultation and the system re-encrypts Jim’s record. After, finishing her consultations, Kate returns to the clinic and uploads the records of patients visited to the database. The system generates a call list for Kate of those patients who she has to contact for follow-up information and make clinic appointments.

Psikiyatrik Takip Sistemi İçin Kullanım Senaryosu

Kate psikiyatrik hasta bakımı konusunda uzman bir hemşiredir. Görevlerinden biri de hastaları evlerinde ziyaret edip tedavilerinin etkinliğini ve ilaçların yan etkilerini kontrol etmektir. Ev ziyareti gününde Kate psikiyatrik hasta takip sistemine giriş yapar ve o güne ait hasta ziyaret listesi ve hastalar hakkında özet bilgilerin çıktısını alır. Bu hastalarla ilgili bilgilerin dizüstü bilgisayarına indirilmesini talep eder. Sistem dizüstünde kayıtların şifrelenmesi için sistem anahtar cümle ister.

Ziyaret ettiği hastalardan biri depresyon tedavisi gören Jim'dir. Jim, ilaçların kendine iyi geldiğini fakat yan etki olarak geceleri uyuyamadığını söyler. Kate, Jim'in kaydına bakar ve deşifre edilmesi için anahtar cümlesini girer. Reçete edilmiş ilacı ve yan etkilerini kontrol eder. Uykusuzluk bu ilacın yaygın bir yan etkisidir ve Jim'in problemini kaydına not alır ve Jim'e kliniğe uğrayıp ilacını değiştirtmesini önerir. Jim onaylar ve Kate kliniğe geri döndüğünde bir doktor randevusu ayarlayabilmek için bir çağrı yönlendirme kaydı girer. Hasta ziyaretini sonlandırır ve sistem Jim'in kaydını yeniden şifreler. Tüm ziyaretlerini tamamladıktan sonra Kate kliniğe döner ve ziyaret ettiği hastaların kayıtlarını veritabanına yükler. Sistem Kate'e klinik randevuları ayarlaması için bir arama listesi sunar.

Performans Testleri

- Sürüm testlerinin bir kısmı performans ve güvenlik gibi sistem özelliklerini test etme işini de kapsayabilir
- Testler sistemin kullanım profillerini yansıtmalıdır
- Performans testleri genellikle sistem yükünü sistemin performansının kabul edilemeyeceği noktaya vardıracak kadar artıran testlerdir
- Stres testi bir tür performans testidir ve sisteme kasten aşırı yükleme yapılarak sistemin davranışı test edilir

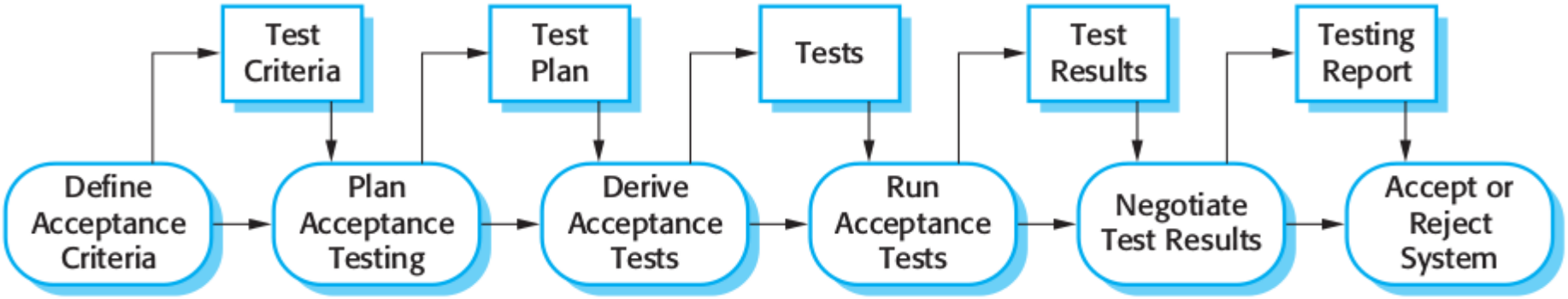
Kullanıcı Testleri

- Kullanıcı veya müşteri testleri, kullanıcıların girdi ve tavsiye sağladığı test süreci adımıdır
- Detaylı sistem ve sürüm testleri yapılmış olsa bile kullanıcı testleri zorunludur
  - Bunun sebebi, kullanıcının çalıştığı ortamın sistemin performansı, güvenilirliği, kullanışlılığı ve sağlamlığı üzerindeki etkisinin büyük olmasıdır. Bu herhangi bir test düzeneği ile taklit edilemez

Kullanıcı Testlerinin Türleri

- Alpha testi
  - Geliştiriciler ve kullanıcılar geliştirici tarafında birlikte test yaparlar
- Beta testi
  - Kullanıcılara, sistem geliştiricilerle birlikte test etmeleri ve hataları ortaya çıkarmaları için bir sürüm hazırlanır
- Kabul testi
  - Müşteriler sistemi geliştiricilerden teslim almak için sistemin kabul edilmeye hazır olup olmadığına ve müşteri tarafında kullanılmaya başlanıp başlanmayacağına karar vermek için test ederler. Özellikle müşteriye özel sistemler

Kabul Testi Süreçleri



Kabul Testi Adımları

- Kabul kriteri belirleme
- Kabul testi planlama
- Kabul testleri türetme
- Kabul testlerini çalıştırma
- Test sonuçlarını tartışma
- Sistemin kabulü/reddi

Çevik Yöntemler ve Kabul Testi

- Çevik yöntemlerde müşteri geliştirme takımının bir parçası olarak çalışır ve sistemin kabulü için alınan kararlardan sorumludur
- Testler müşteri tarafından tanımlanır ve değişiklik yapılınca diğer testlerle birlikte otomatik olarak çalıştırılır
- Ayrı bir kabul testi süreci yoktur
- Buradaki temel problem geliştirici ile birlikte çalışan kullanıcının tüm paydaşların isteklerini temsil edip edemeyeceğidir

Önemli Noktalar

- Yazılımı test ederken, sistemi bozmak amacıyla deneyimleri ve kılavuzları kullanarak test tiplerini belirlemeliyiz
- Mümkünse otomatik testler yazmalıyız. Bu testler sistemde yapılan her değişiklik için yeniden çalıştırılmalıdır
- Önce-test geliştirme, kodlardan önce testlerin yazıldığı bir yaklaşımdır
- Senaryo testi uydurulmuş tipik bir kullanım senaryosu ile test durumlarının türetilmesidir
- Kabul testleri yazılımın gerçek ortamda kullanılmaya başlanması için yeterince iyi olup olmadığının kontrolüdür