Sertifikalı Test Uzmanı

İleri Seviye Ders Programı Test Analisti

Versiyon 2012

International Software Testing Qualifications Board

İleri Seviye Test Analisti Çalışma Grubu: Judy McKay (Başkan), Mike Smith, Erik Van Veenendaal; 2010-2012.

Revizyon Geçmişi

Versiyon	Tarih	Notlar
ISEB v1.1	04EYL01	ISEB Pratisyen Ders Programı
ISTQB 1.2E	EYL03	ISTQB İleri Seviye Ders Programı - EOQ-SG
V2007	12EKİ07	Sertifikalı Test Uzmanı İleri Seviye Ders Programı versiyon 2007
D100626	26HAZ10	2009'da kabul edilen değişikliklerin yapılması, farklı modüller için bölümlerin ayrılması
D101227	27ARA10	Cümle anlamına herhangi bir etkisi olmayan format değişikliklerinin ve düzeltmelerin kabulü.
D2011	23EKİ11	Ders programında ayrım, Öğrenme Hedefleri (ÖH) üzerinde yeniden çalışılması ve ÖH'lerinin uyumlu hale getirilmesi için metinde yapılan değişiklikler. Çalışma Çıktılarının (ÇÇ) eklenmesi.
Alfa 2012	09MAR12	Ekim versiyonundan sonra ülke kurullarından alınan tüm yorumların eklenmesi.
Beta 2012	07NİS12	Alfa versiyonundan sonra ülke kurullarından alınan tüm yorumların eklenmesi.
Beta 2012	07NİS12	Beta Versiyonunun genel kurulda gündeme alınması
Beta 2012	08HAZ12	Düzenlenmiş versiyonunun bir kopyasının ülke kurullarına gönderilmesi
Beta 2012	27HAZ12	Uzman Seviye ve Terimler Sözlüğü çalışma gruplarının yorumlarının eklenmesi
RC 2012	15AĞU12	Yayına aday versiyonunun yayınlanması - nihai ülke kurulu yorumları dahildir
GA 2012	19EKİ12	Genel Kurul yayını için nihai düzenlemelerin ve temize çekme işleminin gerçekleştirilmesi

Önsöz

Yazılım Test ve Kalite Derneği (www.turkishtestingboard.org) tarafından 2014 yılında Türkçeleştirme çalışması tamamlanıp yayınlanan ISTQB Temel Seviye Ders Programına aldığımız çok olumlu tepkiler ve artık Türkiye'de yazılım test sektörünün belli bir olgunluğa erişmiş olması bizleri ISTQB İleri Seviye ders programlarını Türkçeleştirmek için oldukça cesaretlendirdi. Bu cesaretin bir sonucu olarak dernek gönüllülerinin karşılık beklemeden yaptıkları itinalı çabalarla şu anda okumakta olduğunuz çalışmaya ulaşmış olduk. Bu çalışma sırasında daha önce Türkçeleştirme çalışmaları yapılmış olan ISTQB Temel Seviye Ders Programı, ISTQB Yazılım Testi Terimler Sözlüğü baz alındı, çevirinin yazılım test sektöründe kullanılan Türkçe'ye uygun olmasına dikkat edildi. Bu özene rağmen dilin yaşayan bir varlık olduğu, sürekli değiştiği, İngilizcedeki bazı kelimelerin Türkçemizde tam karşılığının olmadığı ve yazılım test sektöründe terimlerin halen standartlaşmadığı gözardı edilmemelidir. Bu kısıtlardan dolayı çevirinin yaşanan gelişmeler ışığında her zaman güncel olabilmesi için yeni gelişmeleri ve önerilerinizi info@turkishtestingboard.org e-posta adresine gönderebilirsiniz.

Çevirinin hızla büyümekte olan Türkiye yazılım test sektörüne faydalı olması dileğiyle.

Yazılım Test ve Kalite Derneği
Mart, 2015

Teşekkür

ISTQB (International Software Testing Qualifications Board – www.istqb.org) yazılım testi terimler sözlüğünün Türkçeleştirme çalışmasına katkıda bulunan Yazılım Test ve Kalite Derneği terimler sözlüğü çalışma grubu üyelerine burada tekrar teşekkür etmek isteriz. Terimler sözlüğü çalışma grubu üyeleri (alfabetik sıraya göre):

- Adem Çağlar
- Ali Dağdemir
- Aydın Akkaya
- Barış Sarıalioğlu
- Burak Yolaçan
- Burcu Nurcan
- Emrah Yayıcı
- Fatma Molu
- Halil Yaman Manargalı
- Hsan Ulusoy
- İlhami Ulusoy
- Kadir Herkiloğlu
- Koray Yitmen
- Mehmet Nuri Gülöksüz
- Merve İçöz
- Onur Sertel
- Ömer Hamdi Kaya
- Pınar Cinali
- Semih İnce
- Selin Hekimoğlu
- Sera Seren
- Üncile Algül
- Volkan Arısoy

Hakkımızda

Yazılım Test ve Kalite Derneği – Turkish Testing Board (TTB) – www.turkishtestingboard.org

Yazılım Test ve Kalite Derneği kar amacı gütmeyen ve Türkiye'deki bilişim profesyonellerinin yazılım testi alanında ISTQB standartlarında eğitilmesi ve sertifikalanmasını hedefleyen bir dernektir. Temel faaliyetleri:

- Türkiye bilişim sektörünün uluslararası pazarlarda rekabet edebilmesi için sektörün yazılım testi ve kalitesi konusunda bilgilendirilmesi,
- Türkçeleştirme çalışmalarıyla uluslararası bilgi birikiminin Türkiye bilişim sektörüne kazandırılması,
- Konferanslar düzenlenmesi,
- Sektör raporlarının hazırlanmasıdır.

TestIstanbul Konferansları – www.testistanbul.org

Yazılım Test ve Kalite Derneği tarafından 2010 yılından itibaren her yıl düzenlenen TestIstanbul Konferansları, yazılım testi ve kalitesi alanında Doğu Avrupa, Ortadoğu ve Kuzey Afrika bölgesinin en büyük etkinliklerinden biri olup yerli ve yabancı bine yakın profesyoneli bir araya getirmektedir.

Test Panelleri

TestFinance Paneli

Bankacılık ve finans sektörünün üst düzey yöneticilerinin katılımıyla düzenli aralıklarla dernek tarafından gerçekleştirilen finans sektöründe yazılım testinin öneminin ve geleceğinin tartışıldığı paneller serisidir.

TestInsurance Paneli

Sigortacılık sektörünün üst düzey yöneticilerinin katılımıyla düzenli aralıklarla dernek tarafından gerçekleştirilen sigortacılık sektöründe yazılım testinin öneminin ve geleceğinin tartışıldığı paneller serisidir.

TestTelco Paneli

Telekom sektörünün üst düzey yöneticilerinin katılımıyla düzenli aralıklarla dernek tarafından gerçekleştirilen telekom sektöründe yazılım testinin öneminin ve geleceğinin tartışıldığı paneller serisidir.

TestDefense Paneli

Savunma sanayi sektörünün üst düzey yöneticilerinin katılımıyla düzenli aralıklarla dernek tarafından gerçekleştirilen savunma sanayi sektöründe yazılım testinin öneminin ve geleceğinin tartışıldığı paneller serisidir.

TestAnkara Paneli

Ankara ve çevre bölgede çalışan ve yazılım geliştirme yaşam döngüsünde yer alan tüm paydaşların (iş birimleri, test mühendisleri, yazılımcılar, iş analistleri, proje yöneticileri, veritabanı yöneticileri ve tasarımcıların) katılımının hedeflendiği yazılım testinin öneminin ve geleceğinin tartışıldığı paneller serisidir.

Turkey Software Quality Report – Türkiye Yazılım Kalite Raporu

Yazılım Test ve Kalite Derneği tarafından 2011 yılından itibaren yüzlerce bilişim profesyoneli ve akademisyeninin katılımıyla düzenlenen anket sonuçlarının değerlendirilmesiyle hazırlanan, Türkiye bilişim sektörüne yön verir nitelikte çıkarımların olduğu rapordur. İngilizce yayınlanan rapor tüm ISTQB üye dernekleri aracılığıyla 100'den fazla ülkedeki bilişim profesyoneline ulaşmaktadır.

ISTQB Worldwide Software Testing Report – ISTQB Dünya Yazılım Test Raporu

International Software Testing Qualifications Board tarafından 100'den fazla ülkeden on binlerce bilişim profesyoneli ve akademisyeninin katılımıyla düzenlenen anket sonuçlarının değerlendirilmesiyle hazırlanan, dünya bilişim sektörüne yön verir nitelikte çıkarımların olduğu rapordur.

International Software Testing Qualifications Board (ISTQB) – www.istqb.org

Merkezi Belçika'da bulunan uluslararası, kar amacı gütmeyen bir dernek olan ISTQB, yazılım test sektörünün gelişimi, standartlaşması için müfredatlar oluşturup bu müfredatlar doğrultusunda sertifika sınavları düzenlemektedir. Haziran 2013 itibariyle 100'den fazla ülkede 354,000'den fazla profesyonel ISTQB sertifikası almıştır. ISTQB, ülkelerde bağımsız dernekler şeklinde temsil edilmektedir. Türkiye'de ISTQB'nin temsilciliğini 2006 yılından beri Yazılım Test ve Kalite Derneği üstlenmektedir.

Yazılım Test ve Kalite Derneği | www.turkishtestingboard.org | info@turkistesingboard.org
Tel: +90 212 276 06 48 | Faks: +90 212 276 06 47

İçindekiler

Teşekkür

Bu doküman, International Software Testing Qualifications Board'un İleri Seviye Çalışma Grubunun alt çalışma grubu olan test analisti çalışma grubunun çekirdek ekibi tarafından hazırlanmıştır: Judy McKay (Başkan), Mike Smith, Erik van Veenendaal.

Çekirdek ekip, gözden geçirme ekibine ve ulusal kurullara önerilerinden ve geribildirimlerinden dolayı teşekkür eder.

İleri Seviye Çalışma Grubunun üyeleri (soyadına göre alfabetik sırayla):

Graham Bath, Rex Black, Maria Clara Choucair, Debra Friedenberg, Bernard Homès (Başkan Yardımcısı), Paul Jorgensen, Judy McKay, Jamie Mitchell, Thomas Mueller, Klaus Olsen, Kenji Onishi, Meile Posthuma, Eric Riou du Cosquer, Jan Sabak, Hans Schaefer, Mike Smith (Başkan), Geoff Thompson, Erik van Veenendaal, Tsuyoshi Yumoto.

Aşağıdaki kişiler, bu ders programının gözden geçirilmesine, yorumlanmasına ve oylanmasına katkıda bulunmuşlardır:

Graham Bath, Arne Becher, Rex Black, Piet de Roo, Frans Dijkman, Mats Grindal, Kobi Halperin, Bernard Homès, Maria Jönsson, Junfei Ma, Eli Margolin, Rik Marselis, Don Mills, Gary Mogyorodi, Stefan Mohacsi, Reto Mueller, Thomas Mueller, Ingvar Nordstrom, Tal Pe'er, Raluca Madalina Popescu, Stuart Reid, Jan Sabak, Hans Schaefer, Marco Sogliani, Yaron Tsubery, Hans Weiberg, Paul Weymouth, Chris van Bael, Jurian van der Laar, Stephanie van Dijk, Erik van Veenendaal, Wenqiang Zheng, Debi Zylbermann.

Bu doküman, ISTQB® Genel Kurulu tarafından 19 Ekim 2012 tarihinde resmi olarak yayınlanmıştır.

0. Ders Programına Giriş

0.1 Amaç

Bu ders programının amacı ileri seviye test analisti uluslararası yazılım test uzmanı niteliklerine yönelik bir çerçeve oluşturmaktır. ISTQB®, bu ders programını aşağıdaki kişilere aşağıdaki amaçlar doğrultusunda kullanılmak üzere sunmaktadır:

1. Yerel dillerine tercüme ettirmeleri ve eğitim sağlayıcıları akredite etmek için ulusal kurullara, ulusal kurullar, bu ders programını o ülkede kullanılan konuşma dili ihtiyaçlarını gözönüne alarak uyarlayabilir ve yerel yayınlarda kullanılmak üzere referanslarda değişiklik yapabilir.
2. Ders programlarına yönelik öğrenim hedeflerine göre uyarlanan sınav sorularını yerel dillerinde oluşturabilmeleri için sınav kurullarına,
3. Kurs materyallerini hazırlamaları ve uygun eğitim yöntemlerini belirlemeleri için eğitim sağlayıcılara,
4. Sınava hazırlanmaları için yazılım test uzmanı sertifika adaylarına (eğitimin bir parçası olarak ya da bireysel düzeyde),
5. Yazılım testi konusundaki uzmanlıklarını geliştirmeleri ve kitaplar ve makaleler için temel teşkil etmesi amacıyla uluslararası yazılım ve sistem mühendisliği topluluğuna.

ISTQB®, yazılı izin talep etmeleri ve izin almaları koşuluyla diğer kurum ve kişilerin bu ders programını farklı amaçlar doğrultusunda kullanmasına müsaade edebilir.

0.2 Genel Bakış

İleri Seviye, üç ayrı ders programından oluşmaktadır:

- Test Yöneticisi
- Test Analisti
- Teknik Test Analisti

İleri Seviyeye Genel Bakış dokümanı aşağıdaki başlıkları içermektedir:

- Ders programlarının her birine yönelik çalışma çıktıları
- Ders programlarının her birinin özeti
- Ders programları arasındaki ilişkiler
- Bilişsel seviyelere (K-seviyeleri) ilişkin açıklamalar
- Ekler

0.3 Değerlendirilebilir Öğrenim Hedefleri

Öğrenim Hedefleri, çalışma çıktılarını destekler ve İleri Seviye Test Analisti Sertifikasyonunun alınabilmesi için gerekli olan sınava hazırlanılması amacıyla kullanılır. Genel anlamda bakıldığında bu ders programı, K1 seviyesindedir. Bu, adayın bir terimi veya kavramı tanıyacağı, anımsayacağı ve hatırlayacağı anlamına gelir. K2, K3 ve K4 seviyelerindeki öğrenim hedefleri, ilgili bölümün başında gösterilmiştir.

1. Test Süreci

Anahtar Sözcükler

somut test senaryosu, çıkış kriteri, üst seviye test senaryosu, alt seviye test senaryosu, mantıksal test senaryosu, test kontrolü, test tasarımı, test yürütme, test uyarlama, test planlama

Test Süreci için Öğrenim Hedefleri

1.2 Yazılım Geliştirme Yaşam Döngüsünde Test

TA-1.2.1 (K2) Farklı yazılım geliştirme yaşam döngüsü modelleriyle çalışırken test analistinin zamanlamasının ve katılım seviyesinin nasıl ve neden değişiklik gösterdiğini açıklayın

1.3 Test Gözetimi, Planlama ve Kontrol

TA-1.3.1 (K2) Testin planlanması ve kontrolüne yardımcı olmak için test analisti tarafından atılan adımları özetleyin

1.4 Test Analizi

TA-1.4.1 (K4) Proje açıklaması ve yaşam döngüsü modeliyle birlikte verilen bir senaryonun, analiz ve tasarım aşamalarında test analistinin görevlerini belirleyin

1.5 Test Tasarımı

TA-1.5.1 (K2) Test koşullarının neden paydaşlar tarafından anlaşılması gerektiğini açıklayın TA-1.5.2 (K4) Verilen iş senaryosuna yönelik en uygun alt (somut) ve üst seviye (mantıksal) test senaryolarını belirleyin **1.6 Testin Uyarlanması**

TA-1.6.1 (K2) Test analizi ve test tasarımına yönelik tipik çıkış kriterlerini açıklayın ve bu kriterlerin sağlanmasının testin uyarlanması için harcanan iş gücünü nasıl etkilediğini belirleyin

1.7 Testin Yürütülmesi

TA-1.7.1 (K3) Belirli bir senaryoda, testlerin yürütülmesi sırasında atılması gereken adımları ve göz önünde bulundurulması gereken hususları belirleyin

1.8 Çıkış Kriterini Değerlendirme ve Raporlama

TA-1.8.1 (K2) Test senaryosunun yürütülmesi sırasında durum bilgilerinin doğruluğunun neden önemli olduğunu açıklayın

1.9 Test Kapanışı İşlemleri

TA-1.9.1 (K2) Test kapanışı işlemleri sırasında test analisti tarafından sunulması gereken çalışma çıktılarına dair örnekler verin

1.1 Giriş

ISTQB® Temel Seviye ders programında, temel test süreci aşağıdaki faaliyetleri kapsayacak şekilde tanımlanmıştır:

- Planlama, gözetim ve kontrol
- Analiz ve tasarım
- Uyarlama ve yürütme
- Çıkış kriterini değerlendirme ve raporlama
- Test kapanışı işlemleri

Söz konusu faaliyetlerden bazıları, İleri Seviyede, süreçleri arındırmak ve optimize etmek, yazılım geliştirme yaşam döngüsüne daha uygun bir şekilde yerleştirmek ve etkin test gözetimi ve kontrolünü kolaylaştırmak iç in birbirinden bağımsız bir şekilde ele alınmıştır:

- Planlama, gözetim ve kontrol
- Analiz
- Tasarım
- Uyarlama
- Yürütme
- Çıkış kriterini değerlendirme ve raporlama
- Test kapanışı işlemleri

Bu faaliyetler, birbirini takip edecek şekilde ya da paralel olarak uygulanabilir. Örneğin, tasarım, uyarlama ile paralel gerçekleştirilebilir (örn. keşif testi). Doğru testlerin ve test senaryolarının belirlenmesi, bunların tasarlanması ve yürütülmesi, test analistinin odaklanması gereken başlıca alanlardır. Her ne kadar test sürecindeki diğer adımların anlaşılması önemli olsa da test analistinin çalışmalarının çoğunluğu, genellikle test projesinin analiz, tasarım, uyarlama ve yürütme aşamaları sırasında gerçekleşmektedir.

Tecrübeli test uzmanları, bu ders programında tanımlanmış olan farklı test unsurlarını kendi organizasyonlarına, ekiplerine ya da görevlerine uygularken çeşitli zorluklarla karşı karşıya kalmaktadır. Test uzmanları test edilmekte olan sistemin türünün yanında, yazılım geliştirme yaşam döngüsünün türünü de göz önünde bulundurmalıdır.

1.2 Yazılım Geliştirme Yaşam Döngüsünde Test

Yazılım geliştirme yaşam döngüsü test stratejisinin bir parçası olarak ele alınmalıdır. Test analistinin katılım sağlayacağı yerler, farklı yaşam döngülerinde değişiklik göstermektedir. Test analistinin katılımı ve test analistinden beklentiler çok büyük farklılık gösterebilir. Test süreçleri birbirinden bağımsız şekilde gerçekleştirilmediğinden test analisti, aşağıdaki gibi diğer ilgili organizasyonel alanlara bilgi sunulabilecek noktaların bilincinde olmalıdır:

- İş analisti - gereksinimlerin gözden geçirilmesi
- Proje yönetimi - proje planı girdileri
- Yapılandırma ve değişiklik yönetimi - sürüm doğrulama testi, versiyon kontrolü
- Yazılım geliştirme – teste hazır olan kod parçacığı ve hazır edilme zamanı
- Yazılım bakımı - hata yönetimi, dönüş süresi (diğer bir deyişle, hatanın tespit edilmesinden çözülmesine kadar geçen süre)
- Teknik destek - geçici çözümlere yönelik doğru dokümanlar
- Teknik dokümanların hazırlanması (Örn. veritabanı tasarımları) - bu dokümanlar için faydalanılan kaynakların yanı sıra dokümanların teknik olarak gözden geçirilmesi

Test faaliyetleri sıralı, döngüsel ya da artırımlı yapıya sahip olabilmektedir. Test faaliyetlerinin seçilen yazılım geliştirme yaşam döngüsü modeline uygun olması gerekmektedir. Örneğin, sıralı V modelinde sistem test seviyesinde uygulanan ISTQB® temel test süreci, aşağıdaki şekilde olabilir:

- Sistem testinin planlaması, proje planlamasıyla aynı anda gerçekleşir ve test kontrolü, sistem testi yürütme ve kapanış işlemleri tamamlanana kadar devam eder.
- Sistem testinin analiz ve tasarımı, gereksinimlerin analizi, üst (mimari) ve alt (birim)seviye tasarımların yapılmasıyla paralel gerçekleştirilir.
- Sistem test ortamı (örn. test ortamları, test donanımı) uyarlaması, sistem tasarımı sırasında başlayabilir. Ancak bu hazırlığın büyük bir kısmı, kodlama ve birim testleriyle ve sistem test yürütmesinden birkaç gün öncesine kadar uzayabilen sistem test uyarlama faaliyetleriyle aynı anda gerçekleştirilecektir.
- Sistem testinin yürütülmesi, sistem test giriş kriterlerinin tümü karşılandığında (ya da devredışı bırakıldığında) başlar. Bu, tipik olarak en azından birim testinin ve genellikle birim entegrasyon testinin tamamlandığı anlamına gelir. Sistem testinin yürütülmesi, sistem test çıkış kriterleri karşılanana kadar devam eder.
- Sistem testinin çıkış kriterlerinin değerlendirilmesi ve sistem test sonuçlarının raporlanması, sistem testinin yürütülmesi boyunca devam eder ve projenin sona erme süreci yaklaştıkça daha sık ve öncelikle gerçekleştirilir.
- Sistem testinin kapanışı faaliyetleri, sistem test çıkış kriterleri karşılandıktan ve sistem test yürütmesinin tamamlandığı beyan edildikten sonra gerçekleştirilir ancak kimi zaman, kullanıcı kabul testi bitene ve tüm proje faaliyetleri son bulana kadar geciktirilebilir.

Döngüsel ve artırımlı modellerde, görevler bahsi geçen sıralamayla gerçekleşmeyebilir ve bazı görevler hariç tutulabilir. Örneğin, döngüsel modellerde döngülerin her biri için indirgenmiş bir standart test süreci dizisi kullanılabilir. Analiz ve tasarım, uyarlama, yürütme, değerlendirme ve raporlama, döngülerin her biri için gerçekleştirilirken planlama, projenin başlangıcında ve kapanış raporlaması ise projenin sonunda gerçekleştirilir. Çevik yazılım geliştirme projelerinde yaygın olarak daha gayri resmi bir süreç takip edilmekte ve proje kapsamında değişikliklerin kolaylıkla ortaya çıkması için daha yakın bir çalışma ilişkisi kurulmaktadır. Çevik metodolojiler, "hafif" bir süreç olduğundan günlük "ayakta" toplantılar gibi daha hızlı bir iletişim yöntemini uyguladığından test dokümantasyonu daha az kapsamlıdır (toplantılar yaklaşık 10-15 dakika sürdüğünden "ayakta toplantı" olarak adlandırılır dolayısıyla kimsenin oturmasına gerek kalmaz ve herkes aktif katılım sağlar).

Diğer tüm yaşam döngüsü modelleri dışında çevik projeler, test analistinin en kısa zamanda müdahil olmasını gerektirir. Test analisti, projenin başlangıcından itibaren sürece dahil olmalı ve yazılımcılar mimari ve tasarım üzerinde çalışırken onlarla işbirliği içinde hareket etmelidir. Gözden geçirmeler resmi bir sürece oturtulmayabilir ancak yazılımın geliştirilme süreci boyunca devam etmelidir. Test analistlerinin proje boyunca katılımları beklenmektedir. Bu yoğunluk nedeniyle çevik proje ekiplerinin üyeleri genellikle tek bir projeye atanırlar ve projenin tüm aşamalarına katılım sağlarlar.

Döngüsel/artırımlı modeller, geliştirilen yazılımda sürekli değişikliklerin olması beklenen çevik yaklaşımdan, kendi içinde döngüsel/artırımlı bir yapıya sahip V modele kadar (kimi zaman gömülü döngüsel olarak adlandırılır) değişiklik gösterir. Gömülü döngüsel modellerin söz konusu olduğu durumlarda test analistinin standart planlama ve tasarım aşamalarına katılması beklenir ancak yazılım geliştirildikçe, test edildikçe, değiştirildikçe ve uygulamaya geçirildikçe daha interaktif bir rol oynamalıdır.

Hangi yazılım geliştirme yaşam döngüsü kullanılırsa kullanılsın test analistinin, şahsi katılımına ve katılımının zamanlamasına dair beklentileri anlamalıdır. Yukarıda bahsedilen V modeli kapsamındaki döngüsel model gibi çok sayıda hibrid model bulunmaktadır. Test analisti, katılım sağlaması gereken anı belirlemek üzere hazır bir modelin tanımına bağlı kalmak yerine en etkin rolü ve çalışma şeklini belirlemelidir.

1.3 Test Planlama, Gözetim ve Kontrol

Bu bölüm, planlama, gözetim ve kontrol test süreçlerine odaklanmaktadır.

1.3.1 Test Planlama

Çoğu zaman test planlama, test faaliyetlerinin başlangıcında gerçekleşir ve test stratejisinde tanımlanan misyonu ve hedefleri yerine getirmek üzere gerekli tüm faaliyet ve kaynakların tanımlanmasını ve planlanmasını kapsar. Test planlaması sırasında test yöneticisi ile birlikte çalışan test analistinin aşağıdakileri göz önünde bulundurması ve bunlara uygun planlama yapması önemlidir:

- Test planlarının fonksiyonel testle sınırlı olmamasını sağlayın. Tüm test çeşitleri, test planında göz önünde bulundurulmalı ve bu doğrultuda plan hazırlanmalıdır. Örneğin fonksiyonel teste ek olarak, test analisti kullanılabilirlik testinden sorumlu olabilir. Bu test çeşidi, test planı dokümanında da ele alınmalıdır.
- Test tahminlemelerini test yöneticisiyle birlikte gözden geçirin ve test ortamının sağlanması ve tedarik edilmesi için yeterli zamanın ayrıldığından emin olun.
- Yapılandırma testine yönelik plan yapın. Farklı çeşitlerde işlemciler, işletim sistemleri, sanal makineler, tarayıcılar ve çeşitli çevresel ekipmanlar çok sayıda olası yapılandırmada bir araya getirilecekse bu kombinasyonları yeterli ölçüde kapsayacak test tekniklerini uygulamaya yönelik plan yapın.
- Dokümanları test etmeye yönelik plan yapın. Yazılımlar kadar dokümantasyonlarda kullanıcılar için temin edilir. Dokümanların etkin olabilmeleri için doğru ve güncel olmaları gerekir. Test analisti, dokümanları test etmek için zaman ayırmalıdır ve ekran görüntülerinde ya da videolarda kullanılacak verileri hazırlamaya yardımcı olmaları için teknik personelle birlikte çalışmak durumunda kalabilir.
- Kurulum prosedürlerini test etmeye yönelik plan yapın. Yedekleme ve geri yükleme prosedürlerinin yanı sıra kurulum prosedürleri de uygun şekilde test edilmelidir. Bu prosedürler, yazılımdan daha önemli olabilir; yazılım kurulamıyorsa kullanılamayacaktır. Test analisti başlangıç testini genellikle, nihai kurulum süreçleri uygulamaya girmeden önce yapılandırılmış bir sistem üzerinde gerçekleştirdiğinden bunun planlanması sırasında zorluklarla karşılaşabilir.
- Testi, yazılım yaşam döngüsüyle aynı doğrultuda olacak şekilde planlayın. Yapılacak işlerin birbirini takip eder şekilde gerçekleştirilmesi çoğu zaman mümkün olmamaktadır. Çoğu işin, (en azından kısmen) aynı anda gerçekleştirilmesi gerekmektedir. Test analisti, seçilen yaşam döngüsünden ve kendisinin tasarıma, geliştirmeye ve uyarlamaya ne kadar katılım göstereceğine dair paydaşların beklentilerden haberdar olmalıdır. Bu aynı zamanda onaylama ve regresyon testi için zaman ayrılmasını da kapsar.
- Farklı disiplinlerden kişilerin biraraya geldiği ekiplerde riskleri tanımlamak ve analiz etmek için daha fazla zaman ayrılması gerektiğini unutmayın. Her ne kadar genellikle risk yönetimi çalıştaylarının düzenlenmesinden sorumlu olmasa da, test analistinin bu faaliyetlere aktif katılım sağlaması beklenmektedir.

Test esası, test koşulları ve test senaryoları arasında karmaşık ilişkiler olabilir. Dolayısıyla söz konusu çalışma ürünleri arasında da çoklu ilişkiler bulunabilir. Test planlamasının ve kontrolünün etkin bir şekilde gerçekleştirilebilmesi için bunların anlaşılması gerekir. Test analisti genellikle, bu ilişkileri belirleyebilecek ve bağımlılıkları mümkün olduğunca ortadan kaldırmak üzere çalışacak en uygun kişidir.

1.3.2 Test Gözetimi ve Kontrol

Her ne kadar test gözetimi ve kontrol test yöneticisinin görevleri arasında yer alsa da test analisti, kontrolün daha sağlıklı yapılabilmesi için gerekli ölçümlerle sürece katkıda bulunur.

Yazılım geliştirme yaşam döngüsü süresince büyük miktarda sayısal veri toplanmalıdır (Örn. tamamlanan planlama faaliyetlerinin yüzdesi, erişilmiş kapsam yüzdesi, başarılı/başarısız olan test senaryosu sayısı). Senaryoların her birinde bir temel çizgi (diğer bir deyişle, referans standart) belirlenmelidir ve ardından ilerleme, bu temel çizgiyle karşılaştırılarak takip edilmelidir. Her ne kadar test yöneticisi, özet niteliğindeki metrik bilgilerin derlenmesinden ve raporlanmasından sorumlu olacak olsa da test analisti, metriklerin her birine yönelik bilgi toplar. Tamamlanan test senaryolarının, yazılan hata raporlarının ve ulaşılan kilometre taşlarının her biri, toplam proje metriklerinde bir araya getirilecektir. Metriklerin gerçek durumu yansıtabilmeleri için çeşitli izleme araçlarına girilen verilerin mümkün olduğunca doğru olması çok önemlidir.

Doğru metrikler, yöneticilerin bir projeyi yönetebilmelerini (izleyebilmelerini) ve gerekli olduğu durumlarda değişiklik yapabilmelerini (kontrol edebilmelerini) mümkün kılar. Yazılımın bir alanında rapor edilen hata sayısının çok yüksek olması, bu alanda ek testlerin yapılmasına ilişkin gerekliliğe işaret edebilir. Gereksinimlerin ve risklerin kapsanma bilgisi (izlenebilirlik), kalan işleri öncelik sırasına koymak ve kaynakları tahsis etmek için kullanılabilir. Kök neden bilgisi, süreç iyileştirmesine yönelik alanları belirlemek için kullanılır. Kaydedilen bilgiler doğru ise proje kontrol edilebilir ve doğru durum bilgileri, paydaşlara rapor edilebilir. Gelecekteki projeler, planlama sırasında önceki projelerden elde edilen bilgilerin göz önünde bulundurulması durumunda daha etkin bir şekilde planlanabilir. Doğru verilerin sayısız kullanım alanı vardır. Verinin doğru, zamanında ve objektif olmasını sağlamak, test analistinin yükümlülüğündedir.

1.4 Test Analizi

Test planlaması sırasında test projesinin kapsamı tanımlanır. Test analisti, aşağıdakileri yapabilmek için bu kapsamdan faydalanır:

- Test esasını analiz etmek
- Test koşullarını tanımlamak

Test analistinin, test analizinde etkin bir şekilde ilerleme gösterebilmesi için aşağıdaki giriş kriterlerinin karşılanması gerekir:

- Test esası olarak baz alınacak test nesnesini açıklayan bir dokümanın olması
- Bu dokümanın gözden geçirme aşamasından geçmiş ve gözden geçirmenin ardından ihtiyaç duyulduğunda güncellenmiş olması
- Kalan test çalışmalarını tamamlamak üzere kullanılabilecek bütçe ve test planının bulunması

Test koşulları genel olarak test esasının ve test hedeflerinin analiziyle tanımlanır. Dokümantasyonun eski ya da olmadığı bazı durumlarda test koşulları, ilgili paydaşlarla konuşularak da tanımlanabilir (örn. çalışma atölyelerinde ya da planlama aşamasında). Bunun ardından söz konusu koşullar, test tasarım teknikleri kullanılarak test edilecek öğenin belirlenmesi için kullanılır.

Her ne kadar test koşulları test edilmekte olan öğeye özel olsa da test analistinin göz önünde bulundurması gereken bazı standart hususlar bulunmaktadır.

- Genellikle, test koşullarını farklı seviyelerde ayrıntı içerecek şekilde tanımlamak önerilir. Başlangıçta, "x ekranının fonksiyonalitesi" gibi testin genel hedeflerini tanımlamak için üst seviye koşullar belirlenir. Bunu takiben, daha özel test senaryolarına esas teşkil etmesi açısından daha detaylı koşullar tanımlanır. Örn. "X ekranı, olması gereken uzunluktan bir hane kısa olan hesap numarasını reddediyor". Test koşullarını belirlemek için bu tür hiyerarşik bir yaklaşımın kullanılması, kapsamın üst seviye unsurlar için yeterli olmasını sağlamaya yardımcı olabilir.

- Ürün riskleri tanımlandıysa ürün risklerinin her birini ele almak için gerekli olan test koşulları tanımlanmalı ve söz konusu risk unsuruna kadar izlenmelidir.

Sonuç olarak bu süreç boyunca test analisti yapacağı testlerin tasarımları hakkında bilgisini artırmalıdır.

1.5 Test Tasarımı

Test tasarımı süreci aşağıdaki faaliyetleri içerir:

- Alt seviye (somut) ya da üst seviye (mantıksal) test senaryolarının en uygun olduğu test alanlarını belirlemek
- Gerekli test kapsamı için test senaryosu tasarım tekniklerini belirlemek
- Tanımlanan test koşullarını karşılayan test senaryoları oluşturmak

Risk analizi ve test planlama sırasında tanımlanan önceliklendirme kriterleri, analiz ve tasarım aşamalarından uyarlama ve yürütme aşamalarına kadar tüm süreç boyunca uygulanmalıdır.

Tasarlanmakta olan test çeşitlerine bağlı olarak test tasarımının giriş kriterlerinden biri, tasarım çalışmaları sırasında kullanılacak olan test otomasyon araçlarının elverişliliği olabilir.

Testler tasarlanırken aşağıdakilerin unutulmaması gerekir:

- Bazı test öğeleri, detaylı testlerin tanımlanması yerine sadece test koşullarının tanımlanmasıyla daha iyi bir şekilde ele alınabilir. Bu durumda test koşulları, detaylı testlerde bir kılavuz olarak kullanılmak üzere tanımlanmalıdır.
- Başarı/başarısızlık kriterleri açık bir şekilde tanımlanmalıdır.
- Testler sadece testi hazırlayan kişi tarafından değil, diğer test uzmanları tarafından da anlaşılabilecek bir şekilde tasarlanmalıdır. Testi hazırlayan kişinin, testi yürütecek kişi olmadığı hallerde, diğer test uzmanlarının, testin önemini ve test hedeflerini anlayabilmeleri için daha önce tanımlanan testleri okumaları ve anlamaları gerekecektir.
- Testler, testleri gözden geçirecek olan yazılımcılar ve testleri onaylaması gereken denetçiler gibi diğer paydaşlar açısından da anlaşılabilir olmalıdır.
- Testler, sadece kullanıcı tarafından görülebilen arayüzdeki etkileşimleri değil, yazılımın aktörlerle etkileşimlerini de (örn. son kullanıcılar, diğer sistemler) kapsayacak şekilde tasarlanmalıdır. Süreçler arası iletişimler de yazılımla etkileşim içine girer ve hata içerebilir; bu nedenle, test analistinin testleri, söz konusu riskleri ortadan kaldıracak şekilde tasarlaması gerekmektedir.
- Testler, çeşitli nesneleri arasındaki ara yüzleri test edecek şekilde tasarlanmalıdır.

1.5.1 Somut ve Mantıksal Test Senaryoları

Test analistinin görevlerinden bir tanesi, belirli bir duruma en uygun test senaryolarını belirlemektir. Somut test senaryoları, test uzmanının test senaryosunu yürütmesi (veri gereksinimleri dahil olmak üzere) ve sonuçlarını onaylaması için gerekli tüm bilgi ve prosedürleri sunar. Somut test senaryoları, gereksinimlerin açık bir şekilde tanımlandığı, testi gerçekleştirecek kişilerin daha az deneyimli olduğu ve testlerin, denetçiler gibi başkaları tarafından onaylanmasının gerekli olduğu durumlarda oldukça kullanışlıdır. Somut test senaryoları, kusursuz bir şekilde tekrarlanabilir (diğer bir deyişle, başka bir test uzmanı da aynı sonuca ulaşacaktır) olduğu gibi test sırasında test uzmanının becerilerini sınırlama eğilimi gösterebilir ve ciddi miktarda bakım çalışmasının yapılmasını zorunlu kılabilir.

Mantıksal test senaryoları, test edilmesi gereken unsura yönelik kılavuz işlevi görür ve test analistinin, testi yürütürken takip etmekte olduğu prosedürü veya mevcut verileri değiştirmesine olanak tanır. Mantıksal test senaryoları, somut test senaryolarına kıyasla daha iyi bir kapsam sunabilir. Bu aynı zamanda tekrarlanabilirlik açısından kayıp yaşanacağı anlamına gelir. Mantıksal test senaryoları, gereksinimlerin açık bir şekilde tanımlanmadığı, testi yürütecek uzmanın hem test hem de ürün konusunda deneyimli olduğu ve dokümantasyona ihtiyaç duyulmadığı (örn. herhangi bir denetim gerçekleştirilmeyecekse) durumlarda kullanımı açısından idealdir. Mantıksal test senaryoları, gereksinimlerin henüz açık bir şekilde tanımlanmadığı anal iz sürecinin başlangıcında tanımlanabilir. Söz konusu test senaryoları, gereksinimlerin daha tanımlı ve stabil bir hal aldığı durumlarda somut test senaryolarını geliştirmek için kullanılabilir. Bu durumda test senaryosu, birbirini takip edecek şekilde olu şturulur. Yürütülecek test senaryosunun somut test senaryoları olması kaydıyla mantıksaldan somuta doğru geçiş yaşanır.

1.5.2 Test Senaryolarının Oluşturulması

Test senaryoları, test koşullarının belirlenen test tasarım teknikleri (Bkz. Bölüm 3) kullanılarak adım adım ele alınmasıyla tasarlanır.
Test senaryoları, kullanılmakta olan test stratejisinin ifade ettiği üzere tekrarlanabilir, onaylanabilir ve test esasına kadar izlenebilir (örn. gereksinimler) olmalıdır.

Test senaryosu tasarımı, aşağıdakilerin tanımlanmasını gerektirir:

- Hedef
- Projenin ya da test ortamı gereksinimleri, teslimat planları, sistem durumu gibi ön koşullar
- Test verisi gereksinimleri (test senaryosunun yürütülebilmesi için sistemde bulunması gereken verilerin yanı sıra test senaryosuna yönelik giriş bilgileri)
- Beklenen sonuçlar
- Etkilenen veriler, sistemin durumu, sonraki süreçlerin tetikleyicileri gibi son koşullar

Test senaryosu ayrıntıları, geliştirme maliyetini ve tekrarlanabilirliği etkileyebileceğinden test senaryoları gerçek anlamda oluşturulmadan önce belirlenmelidir. Test senaryosunda daha az detayın yer alması, test analistine test senaryosunu yürütürken daha fazla esneklik sağlar ve olası ilgi çekici alanları inceleme olanağı sunar. Diğer yandan ayrıntının az olması, tekrarlanabilirlik yüzdesini de düşürmektedir.

Genellikle, bir testin beklenen sonuçlarının tanımlanması sırasında çeşitli zorluklarla karşılaşılmaktadır. Beklenen sonuçların manuel olarak hesaplanması yorucu ve insan hatalarına açık olacaktır; mümkün olması halinde otomatik bir test sonucu hesaplama mekanizmasının bulunması ya da oluşturulması tercih edilir. Beklenen sonuçların tanımlanması sırasında test uzmanları, sadece ekrandaki sonuçlarla ilgilenmemeli, verileri ve çevresel durumları da göz önünde bulundurmalıdır. Test esasının açık bir şekilde tanımlanması durumunda teorik olarak doğru sonuçların tanımlanması daha kolay olacaktır. Ancak test esasları genellikle belirsiz, çelişkili, önemli alanları kapsam dışında bırakan ya da eksiklikler içeren yapıdadır. Böyle durumlarda test analisti, ilgili konu hakkında uzman olmalı ya da ilgili uzmanlara erişimi olmalıdır. Buna ek olarak test esasının açıkça tanımlandığı durumlarda dahi girdi ve çıktılar arasındaki karmaşık ilişkiler, beklenen sonuçların tanımlanmasını zorlaştırabilir; test sonuçlarını hesaplayabilen bir mekanizmanın bulunması şarttır. Test senaryosunun, sonuçların doğruluğunu belirlemek için herhangi bir mekanizma kullanılmadan yürütülmesi genellikle çok az fayda sağlar ve sıklıkla, sistemde sahte arıza raporlarının oluşmasına neden olur.

Yukarıda tanımlanan faaliyetler, her ne kadar test esasına göre farklılık gösterecek olsa da tüm test seviyelerinde uygulanabilir. Örneğin, kullanıcı kabul testleri, temel olarak gereksinimlere, kullanım senaryolarına ve tanımlı iş süreçlerine dayalı olsa da birim testleri, esasen alt seviye tasarıma, kullanıcı hikayelerine ve kodun kendisine dayalı olabilir. Her ne kadar testin hedefi farklılık gösterse de söz konusu faaliyetlerin tüm test seviyelerinde gerçekleştiği unutulmamalıdır. Örneğin, birim seviyesindeki fonksiyonel test, belirli bir bileşenin ilgili bileşene yönelik ayrıntılı tasarımında belirtilen fonksiyonaliteyi sunduğundan emin olmak açısından tasarlanır. Entegrasyon seviyesindeki fonksiyonel test ise bileşenlerin birbiriyle etkileşim içinde çalıştığını ve etkileşimleri aracılığıyla fonksiyonaliteyi sağladıklarını onaylar. Sistem seviyesinde ise uçtan uca fonksiyonalite testin hedeflerinden biri olmalıdır. Testler analiz edilirken veya tasarlanırken testin amacının yanı sıra testin hedef seviyesinin de hatırlanması önemlidir. Bu yaklaşım ihtiyaç duyulabilecek araçların yanı sıra gerekli ayrıntı seviyesini belirlemeye de yardımcı olur (örn. sürücüler, birim test seviyesindeki taklit uygulamalar).

Test koşullarının ve test senaryolarının geliştirilmesi sırasında tipik olarak bazı dokümanlar oluşturulur ve bunlar da test çalışmasının ürünlerine dönüşür. Pratikte test çalışması ürünlerinin belgelendirildiği kapsam bile belirgin ölçüde değişiklik göstermektedir. Bu, aşağıdakilerden herhangi birinden etkilenebilir:

- Proje riskleri
- Dokümantasyonun projeye katacağı değer
- İzlenmesi gereken standartlar ve/veya karşılanması gereken yönetmelikler
- Kullanılan yaşam döngüsü modeli (örn. Çevik yaklaşım "sadece yeterli sayıda" belge hazırlanmasını hedefler)
- Test analizi ve tasarımı vasıtasıyla test esasının izlenebilirliğine yönelik gereksinim

Testin kapsamına bağlı olarak test analizi ve tasarımı, test nesnelerinin kalite karakteristiklerini ele alabilir. ISO 25000 standardı [ISO25000] (ISO 9126'nın yerine geçer) bu konuya yönelik faydalı bilgiler verir. Sistemler test edilirken ek karakteristikler geçerli olabilir.

Test analizi ve test tasarımı süreçleri, gözden geçirme ve statik analizlerle harmanlanarak geliştirilebilir. Aslında test analizi ve tasarımı bir çeşit statik testtir, çünkü hatalar bu süreçler boyunca incelenen dokümanlarda bulunabilir. Gereksinimlere dayalı test analizi ve test tasarımı, gereksinim gözden geçirme toplantılarına hazırlanmak açısından mükemmel bir yoldur. Testlerin oluşturulması amacıyla gereksinimlerin okunması, gereksinimin anlaşılmasını ve gereksinimin karşılanıp karşılanmadığını değerlendirmek için bir yol belirlenmesini gerektirir. Bu faaliyet genellikle açık olmayan, test edilemez ya da kabul kriterleri tanımlanmamış gereksinimlerin belirlenmesini sağlar. Benzer şekilde test senaryoları, risk analizleri ve test planları da gözden geçirmeye tabi olabilir.

Çevik yaklaşımların kullanıldığı bazı projelerde dokümante edilmiş çok az sayıda gereksinim bulunabilir. Bunlar kimi zaman "kullanıcı hikayeleri" formatında olabilir. Kullanıcı hikayesi, kabul kriterlerini içermelidir. Yazılımın kabul kriterini yerine getirdiğini gösterebiliyorsa, diğer tamamlanan fonksiyonalitelerle entegrasyona hazır olduğu ya da fonksiyonalitesini göstermek üzere hali hazırda entegre edilmiş olduğu düşünülür.

Test tasarımı sırasında ayrıntılı test ortamı gereksinimleri tanımlanabilir ancak pratikte bunlar, test uyarlamasına kadar nihai şeklini almayabilir. Test ortamının test nesnelerinden ve test yazılımından çok daha fazlasını kapsadığı unutulmamalıdır. Örneğin, test ortamı odaları, ekipmanları, personeli, yazılımı, araçları, çevresel donanımları, iletişim ekipmanlarını, kullanıcı yetkilerini ve testleri gerçekleştirmek için gerekli tüm diğer öğeleri kapsayabilir.

Test analizinin ve test tasarımının çıkış kriterleri, proje parametrelerine bağlı olarak değişiklik gösterebilir. Kriterlerin ölçülebilir olması ve sonraki aşamalar açısından gerekli bilgi ve hazırlığın sağlanması çok önemlidir.

1.6 Testin Uyarlanması

Test uyarlaması, test tasarımının yerine getirilmesi anlamına gelir. Bu, otomatik testlerin oluşturulmasını, testlerin yürütülmeye hazırlanmasını (hem manuel hem de otomatik olarak), test verilerinin ve test senaryosunun yürütülmeye başlayabilmesi için test ortamlarının nihai hale getirilmesini ve kaynak tahsisini içeren test yürütme çizelgesinin hazırlanmasını kapsar. Bu aynı zamanda göz önünde bulundurulan test seviyesine yönelik doğrudan ve dolaylı giriş kriterleriyle yapılacak kıyaslamayı ve sürecin önceki aşamalarındaki çıkış kriterlerinin karşılandığından emin olmak için gerekli faaliyetleri içerir. Test çıkış kriterlerinin atlanmış olması durumunda uyarlama sırasında gecikmelerin, kalite problemlerinin ve beklenmedik iş yükünün oluşması mümkündür. Test uyarlaması çalışmalarına başlamadan önce tüm çıkış kriterlerinin karşılanması çok önemlidir.

Yürütme sırasına karar verirken göz önünde bulundurulması gereken çok sayıda unsur olabilir. Bazı durumlarda test senaryolarının test grupları kapsamında düzenlenmesi gerekli olabilir (diğer bir deyişle, test senaryosu grupları). Bu, ilgili test senaryolarının bir arada yürütülebilmesi için testin düzenlenmesine yardımcı olabilir. Risk bazlı test stratejisi kullanılıyorsa risk öncelik sırası, test senaryolarına yönelik yürütme sırasını belirleyebilir. Doğru paydaşlara, ekipmanlara, verilere ve test edilecek fonksiyonalitelere erişim gibi sırayı belirleyecek başka etmenler de bulunabilir. Kodun bölümler halinde yayınlanması ve test çalışmalarının, yazılımın teste hazır olduğu sıralamayla koordine edilmesi sıklıkla izlenen bir yoldur. Özellikle artımlı yaşam döngüsü modellerinde, yazılımın test edilebilir bir sıralamayla teste hazırlandığından emin olmak açısından test analistinin yazılım ekibiyle koordineli çalışması önemlidir. Test uyarlaması sırasında test analisti, testin belirli bir sıralamayla gerçekleştirilmesini gerektirebilecek kısıtlamaları dikkatle inceleyerek manuel ve otomatik testlerin yapılacağı sıralamayı nihai haline getirmeli ve onaylamalıdır. Bağımlılıklar belgelendirilmeli ve kontrol edilmelidir.

Ayrıntı seviyesi ve test uyarlaması sırasında tamamlanan işlerin karmaşıklığı, test senaryolarının ve test koşullarının ayrıntı seviyesinden etkilenebilir. Bazı durumlarda yönetmelikler geçerli olabilir ve testler, Amerika Birleşik Devletleri Federal Havacılık İdaresi'nin DO -178B/ED 12B [RTCA DO-178B/ED-12B] standartları gibi yürürlükteki standartlara uyumlu olduğuna ilişkin kanıt göstermelidir.

Yukarıda da belirtildiği gibi test için test verilerine ihtiyaç duyulmaktadır ve bazı durumlarda veri kümeleri çok büyük olabilir. Uyarlama sırasında test analistleri, veritabanlarına ya da diğer depolama alanlarına yüklemek üzere girdileri ve çevre verilerini oluşturur. Test analistleri, manuel testlerin yanı sıra veri güdümlü otomasyon testlerinde kullanılmak üzere gerekli verileri de oluşturur.

Test uyarlaması, test ortamıyla birlikte değerlendirilir. Bu aşama sırasında ortam tamamen ayarlanmalı ve test yürütmesinden önce onaylanmalıdır. "Amaca uygun" bir test ortamının oluşturulması büyük önem taşımaktadır. Diğer bir deyişle, test ortamı, kontrollü test sırasında ortaya çıkan hataların vurgulanmasını mümkün kılmalı, hata oluşmadığında normal bir şekilde çalışmalı ve g erekli olması halinde daha üst test seviyelerine yönelik üretim ya da son kullanıcı ortamlarını kopyalayabilmelidir. Beklenmedik değişikliklere, test sonuçlarına ve diğer etmenlere bağlı olarak test yürütme sırasında test ortamında değişiklikler yapılması gerekebilir. Ortam değişiklikleri yürütme sırasında yapılırsa söz konusu değişikliklerin, hali hazırda tamamlanmış testler üzerindeki etkisinin değerlendirilmesi önemlidir.

Test uyarlaması sırasında test uzmanları, test ortamının oluşturulmasından ve bakımından sorumlu kişilerin herkes tarafından biliniyor ve erişilebilir olmasını sağlamalı ve tüm test yazılımlarının ve test destek araçlarının ve ilgili süreçlerin kullanıma hazır olduğundan emin olmalıdır. Bu, yapılandırma yönetimini, hata yönetimini ve test kaydı tutma ve yönetimini içerir. Buna ek olarak test analistleri, çıkış kriteri değerlendirmesine ve test sonuçlarını raporlamaya yönelik bilgileri toplamak için kullanılan prosedürleri onaylamalıdır.

Test uyarlamasında dengeli bir yaklaşımın kullanılması akıllıca olacaktır. Örneğin, risk bazlı analitik test stratejileri sıklıkla dinamik test stratejileriyle harmanlanmaktadır. Bu durumda bazı testler keşif testleri şeklinde bir senaryoya bağlı olmadan koşturulacaktır.

Önceden hazırlanmamış testler, geçici ya da hedefsiz olmamalıdır zira bu durum süre ve kapsam açısından belirsizlikler yaratacaktır. Yıllar içinde test uzmanları, saldırılar, hata tahminleme [Myers79] ve keşif testi gibi tecrübeye dayalı çok sayıda teknik geliştirmiştir. Bu tekniklerde de test analizi, test tasarımı ve test uyarlaması yapılmaya devam edecektir ancak bunlar özellikle test yürütme sırasında eş zamanlı olarak yapılacaktır.

Bu tür dinamik test stratejileri takip edilirken testlerin her birinden elde edilen sonuçlar, sonraki testlerin analiz, tasarım ve uyarlama aşamalarını etkileyecektir. Söz konusu bu stratejiler hataların tespiti konusunda yardımcı ve sıklıkla etkin olsa da bunların bazı dezavantajları bulunmaktadır. Söz konusu teknikler, test analistinin uzman olmasını gerektirmektedir, test süresinin öngörülmesi ve kapsamın takip edilmesi zor olabilir ve iyi bir dokümantasyon ya da araç desteği olmazsa tekrarlanabilirlik özelliği kaybedebilir.

1.7 Testin Yürütülmesi

Test yürütme, test nesnesi teslim edildikten ve test yürütmeye yönelik giriş kriteri sağlandıktan (ya da devredışı bırakıldıktan) sonra başlar. Testler, test uyarlaması sırasında belirlenen planlamaya göre yürütülmelidir ancak test analistinin, test sırasında gözlemlenen diğer ilgi çekici test senaryolarını ve davranışlarını kapsam dahiline alması için yeterli süreye sahip olması gerekmektedir (söz konusu sapmalar sırasında tespit edilen arızalar tanımlanmalı ve varsa arızanın yeniden oluşturulması için gerekli betiğe dayalı test senaryosundaki sapmalar da belirtilmelidir). Sistemli bir şekilde tanımlanmış test senaryolarıyla tecrübeye dayalı olarak oluşturulmuş test senaryolarının entegrasyonu, test kapsamındaki boşlukların kapanması sağlar.

Test yürütme faaliyetinin odak noktasında, gerçekleşen sonuçların beklenen sonuçlarla karşılaştırması yer almaktadır. Test analistleri, söz konusu görevleri vurgulamalı ve bunlara odaklanmalıdır; aksi takdirde, testin tasarımına ve uyarlanmasına yönelik tüm çalışmalar, arızalar tespit edilemediğinde (yanlış negatif sonuç) ya da doğru davranışlar yanlış sınıflandırıldığında (yanlış pozitif sonuç) boşa gidecektir. Beklenen ve gerçekleşen sonuçlar eşleşmiyorsa bir olay meydana gelmiş demektir. Olaylar, nedeni belirlemek (test nesnesindeki bir hatadan kaynaklanabilir ya da kaynaklanmayabilir) ve olayın çözümüne yardımcı olmak açısından veri toplamak (hata yönetimi konusunda ayrıntılı bilgi için Bölüm 6'yı inceleyin) amacıyla dikkatli bir şekilde incelenmelidir.

Bir arıza tespit edildiği zaman testin doğruluğundan emin olmak için test dokümanları (test gereksinimleri, test senaryosu, vb) dikkatli bir şekilde incelenmelidir. Bir test dokümanı çeşitli nedenlerden ötürü hatalı olabilir. Hatalı olması durumunda düzeltilmeli ve test yeniden yapılmalıdır. Test esasındaki ve test nesnesindeki değişiklikler, test birden fazla defa başarılı bir şekilde tamamlandıktan sonra bile bir test senaryosunun hatalı olmasına neden olabileceğinden test uzmanları, gözlemlenen sonuçların yanlış bir teste dayanması ihtimalini göz önünde bulundurmalıdır.

Test yürütme sırasında test sonuçları düzgün bir şekilde kayıt altına alınmalıdır. Gerçekleştirilen ancak sonuçları kayıt altına alınmamış testlerin, doğru sonucu belirlemek üzere tekrarlanması gerekebilir ancak bu durum, verimsizliğe ve gecikmelere neden olacaktır. (Uygun şekilde tutulan kayıtların, keşif testi gibi test tekniklerindeki kapsam ve tekrarlanabilirlik sorunlarını bertaraf edebileceğini lütfen unutmayın.) Test nesnesi, test yazılımı ve test ortamları devamlı gelişme gösterebileceğinden tutulan kayıtlar, yapılandırmaların yanı sıra versiyonları da kayıt altına almalıdır. Test kaydı tutma, testlerin yürütülmesine ilişkin ilgili kayıtları içererek kronolojik bir kayıt dosyasının elde edilmesini sağlar.

Hem her bir testin hem de faaliyetlerin ve olayların sonuçları kaydedilmelidir. Testlerin her biri kendisine özel şekilde tanımlanmalı ve test yürütülürken testin durumu kayıt altına alınmalıdır. Test yürütmeyi etkileyen tüm olaylar kayıt altına alınmalıdır. Test kapsamını ölçmek için yeterli bilgi kaydedilmeli ve test sırasında yaşanan gecikmelerin ve kesintilerin nedenleri belgelendirilmelidir. Buna ek olarak, test kontrolünü, test ilerleme raporunu, çıkış kriterinin ölçümünü ve test süreci ilerleyişini desteklemek üzere gerekli bilgiler kayıt altına alınmalıdır.

Kayıt altına alma yöntemleri, teste ve stratejiye göre değişiklik gösterir. Örneğin, otomatik birim testi gerçekleştiriliyorsa otomatik testler, kayıt altına alınan bilgilerin çoğunu üretmelidir. Manuel test gerçekleştiriliyorsa test analisti testin yürütülmesine ilişkin bilgileri test yönetim aracına kaydetmelidir. Testin yürütülmesinde olduğu gibi bazı durumlarda kayıt altına alınan testin yürütülmesine ilişkin bilgilerin miktarı, yönetmeliklere ya da denetimlere ilişkin gereksinimlerden etkilenir.

Bazı durumlarda kullanıcılar ve müşteriler testin yürütülmesine katılabilir. Bu, sistemde güven yaratılması için kullanılabilecek yollardan biridir ve bu sayede, testin daha az sayıda hatayla karşılaşacağı varsayılır. Bu tür bir varsayım ilk safhalardaki test seviyelerinde genellikle geçersizdir ama kullanıcı kabul testi sırasında geçerlilik kazanabilir.

Aşağıda, test yürütme sırasında göz önünde bulundurulması gereken bazı özel başlıklara değinilmiştir:

- "İlgisiz" tuhaflıkları tespit edin ve inceleyin. İlgisiz gibi görünen gözlemler veya sonuçlar, genellikle temelde yatan hataların göstergeleridir.
- Yazılımın yapmaması gerekenleri yapmadığından emin olun. Yazılımın yapması gerekenleri yapıp yapmadığını kontrol etmek testin normal odak noktalarından biridir ancak test analisti, yazılımın yapmaması gereken bir davranışta bulunmadığından da emin olmalıdır.

- En iyi test senaryosu grubunu oluşturun ve zaman içinde büyümesini ve değişmesini bekleyin. Kod gelişecektir ve söz konusu yeni fonksiyonaliteleri kapsam dahiline almanın yanı sıra yazılımın diğer alanlarındaki regresyonları kontrol etmek için ek testlerin yapılması gerekecektir. Testteki boşluklar, genellikle yürütme sırasında keşfedilir. Bir test senaryosu grubunun oluşturulması kesintisiz bir süreçtir.
- Bir sonraki test çalışmaları için notlar alın. Yazılım kullanıcıya sunulduğunda ya da piyasaya dağıtıldığında test sona ermez. Yazılımın yeni bir versiyonunun ya da sürümünün üretilmesi olasılığı yüksektir dolayısıyla bilgiler depolanmalı ve bir sonraki test çalışmalarından sorumlu test uzmanlarına aktarılmalıdır.
- Tüm manuel testleri yeniden yapmayı beklemeyin. Tüm manuel testlerin yeniden yapılmasını beklemek gerçekçi olmayacaktır. Bir problem olduğundan şüpheleniliyorsa test analisti söz konusu durumu incelemeli ve test senaryolarının bir sonraki yürütülmesinde yakalanacağını varsaymak yerine not etmelidir.
- Ek test senaryoları için verileri hata takip aracına yükleyin. Önceden hazırlanmamış ya da keşif testi sırasında tespit edilen hatalara ilişkin test senaryosu oluşturmayı göz önünde bulundurun ve bunları regresyon test senaryosu grubuna ekleyin.
- Regresyon testinden önce hataları bulun. Genellikle regresyon testi için ayrılan zaman kısıtlıdır ve regresyon testi sırasında hataların bulunması programda gecikmelere neden olabilir. Regresyon testleri genellikle hataların büyük bir oranını bulmaz; bunun nedeni, söz konusu testlerin hali hazırda yapılmış olması (örn. aynı yazılımın bir önceki versiyonu için) ve hataların önceki testlerde tespit edilmiş olmasıdır. Bu, regresyon testlerinin devredışı bırakılması gerektiği anlamına gelmemelidir; sadece regresyon testlerinin yeni hataları tespit etme becerisi diğer testlerinkinden düşüktür.

1.8 Çıkış Kriterini Değerlendirme ve Raporlama

Test süreci perspektifinden bakıldığında test gözetimi, raporlama gereksinimlerini destekleyecek uygun bilgilerin toplanmasına bağlıdır. Bu testin tamamlanmasına doğru ilerlemenin ölçümünü de kapsar. Planlama stratejilerinde çıkış kriteri tanımlandığında "zorunlu" ve "gerekli" kriterlerinin ayrıştırılması gerekebilir. Örneğin kriterler şu maddeleri içerebilir: "Yazılımda 1. Öncelikli ya da 2. Öncelikli hataların bulunmaması zorunludur" ve "Tüm test senaryolarında %95 başarı oranı elde edilmesi gerekir". Bu durumda "zorunlu" olan kriterdeki maddenin yerine getirilememesi çıkış kriterinin başarısız olmasına neden olurken "gerekli" statüsünde olan "Tüm test senaryolarında %95 başarı oranı elde edilmesi" kriterinde %93'lük başarı oranı yakalanmış olması projenin bir sonraki seviyeye ilerlemesine izin verebilir. Çıkış kriteri, nesnel bir şekilde değerlendirilebilmesi için açık ve net bir şekilde tanımlanmalıdır.

Test analisti, çıkış kriterinin karşılanmasına yönelik ilerlemeyi değerlendirmesi için test yöneticisi tarafından kullanılacak bilgileri temin etmekle ve verilerin doğruluğundan emin olmakla yükümlüdür. Örneğin test yönetimi sisteminin, test senaryosunun tamamlanmasına ilişkin aşağıdaki durum kodlarını sunması halinde:

- Başarılı
- Başarısız
- İstisnayla başarılı

Test analisti, bunların anlamları konusunda detaylı bilgi sahibi olmalı ve durumları kesintisiz bir şekilde güncellemelidir. Yazılımda "İstisnayla başarılı" bir hatanın bulunduğu ancak bunun sistemin fonksiyonalitesini etkilemediği anlamına mı geldiği? Kullanıcının aklının karışmasına neden olabilecek bir kullanılabilirlik hatasının gerçekleşmesi halinde ne olacağı? gibi. Başarı oranının bir "zorunlu" çıkış kriteri olması durumunda test senaryosunun "istisnayla başarılı" yerine "başarısız" olarak değerlendirilmesi hayati önem taşıyan bir etmendir. Başarısızlığın nedeninin bir hata olmamasına rağmen "Başarısız" olarak etiketlenen test senaryolarına dikkat edilmelidir. (örn. test ortamı uygun olmayan bir şekilde yapılandırılmış olabilir). Takip edilen metrikler ya da durum değerlerinin kullanımı konusunda herhangi bir karışıklık olursa test analisti, test yöneticisi ile durumu değerlendirmelidir; bu şekilde, bilgiler proje boyunca doğru ve kesintisiz bir şekilde takip edilebilir.

Test analistinden, test döngüleri sırasında durum raporu vermesi ve testin sonunda hazırlanan nihai rapora katkıda bulunması talep edilebilir. Bu, hata ve test yönetimi sistemlerinden metriklerin toplanmasının yanı sıra genel kapsamın ve ilerlemenin değerlendirilmesini gerektirebilir. Test analisti, raporlama araçlarını kullanabilmeli ve test yöneticisinin ihtiyacı olan bilgileri çıkartabilmesi için talep edilen bilgileri sağlayabilmelidir.

1.9 Test Kapanışı İşlemleri

Test yürütmesinin tamamlandığına karar verildiğinde test projesinin önemli çıktıları tespit edilmeli ve ilgili kişiye aktarılmalı ya da arşivlenmelidir. Bunlar birlikte düşünüldüğünde test kapanışı faaliyetleri olarak adlandırılır. Test analistinin bu çıktıları ihtiyaç duyan paydaşlara ulaştırmak için sürece katılması beklenmektedir. Örneğin, ertelenen ya da kabul edilen bilinen hatalar, sistemi kullanacak ya da sistemin kullanımını destekleyecek kişilere iletilmelidir. Testler ve test ortamları, bakım testinden sorumlu olan kişilere verilmelidir. Diğer bir çıktı ise regresyon test kümesi (otomatik ya da manuel) olabilir. Test çalışması ürünleri hakkındaki bilgi, ilgili bağlantılarla birlikte açık bir şekilde belgelendirilmelidir ve kişilere uygun erişim hakları verilmelidir.

Test analisti, "güçlü yanları" güçlendirmek ve "zayıf yanları" ortadan kaldırmak ya da en azından kontrol altına almak için edinilen tecrübelerin (test projesi kapsamında ve yazılımın geliştirme yaşam döngüsü sırasında elde edilen) belgelendirilebileceği ve planların yapılabileceği geçmişe dönük toplantılara da katılmalıdır. Test analisti, söz konusu toplantılar açısından iyi bir bilgi kaynağıdır ve süreç iyileştirme konusunda geçerli veriler toplanacaksa söz konusu toplantılara katılmalıdır. Toplantıya sadece test yöneticisinin katılacak olması durumunda test analisti, ilgili bilgileri test yöneticisine ileterek projenin doğru bir resminin sunulmasını sağlamalıdır.

Sonuçların, kayıtların, raporların ve diğer dokümanların ve çıktıların yapılandırma yönetiminde arşivlenmesi de gerçekleştirilmelidir. Bu görev genellikle test analisti tarafından yerine getirilmektedir ve gelecekteki projelerde söz konusu bilgilerin kullanılacak olması halinde önemli bir kapanış faaliyetidir.

Test yöneticisi arşivlenmesi gereken bilgilerin neler olduğuna karar verirken test analisti, projenin gelecekte yeniden başlatılacak olması ihtimaline karşın ihtiyaç duyulacak bilgiler konusunda da düşünmelidir. Projenin sonunda söz konusu bilgiler üzerinde düşünmek, proje daha sonra ya da başka bir ekiple yeniden başlatıldığında aylar sürecek efordan kurtulunmasını sağlayacaktır.

2. Test Yönetimi: Test Analistinin Sorumlulukları – 90 Dakika

Anahtar Sözcükler

ürün riski, risk analizi, risk tanımlama, risk seviyesi, risk yönetimi, riski azaltma, risk-bazlı test, test gözetimi, test stratejisi

Test Yönetimi için Öğrenme Hedefleri: Test Analistinin Sorumlulukları

2.2 Test Gözetimi ve Kontrolü

TA-2.2.1 (K2) Projenin uygun şekilde izlenebilmesi ve kontrol edilebilmesi için test sırasında takip edilmesi gereken bilgi türlerini açıklayın

2.3 Dağıtılmış, Dış Kaynaklarla ve Test Ekibine Dahil Edilen Kaynaklarla Yapılan Test

TA-2.3.1 (K2) 24 saat açık bir test ortamında çalışırken takip edilebilecek iyi iletişim uygulamalarına dair örnekler verin

2.4 Risk Bazlı Testlerde Test Analistinin Görevleri

TA-2.4.1 (K3) Risk tanımlama sürecine katılın, risk değerlendirmesini gerçekleştirin ve uygun risk azaltma yöntemleri konusunda önerilerde bulunun

2.1 Giriş

Test yönetimi sırasında her ne kadar test analistinin test yöneticisiyle etkileşim içine girdiği ve test yöneticisine veri sunduğu çok sayıda alan olsa da bu bölümde test analistinin eforunun büyük bir kısmını harcadığı alanlara odaklanılmaktadır. Test yöneticisinin ihtiyaç duyduğu verileri test analistinden istemesi beklenmektedir.

2.2 Test Gözetimi ve Kontrolü

Testin ilerleyişinin takip edildiği başlıca beş boyut bulunmaktadır:

- Ürün (kalite) riskleri
- Hatalar
- Testler
- Kapsam
- Güven

Ürün riskleri, hatalar, testler ve kapsam, test analisti tarafından ölçülebilir ve rapor edilebilir. Güven ise genellikle öznel olarak rapor edilir. Bu metrikleri desteklemek için gerek duyulan bilgilerin toplanması, test analistinin günlük çalışmalarından biridir. Yanlış bilgiler, yanlış eğilim bilgilerine ve yanlış sonuçlara yol açabileceğinden söz konusu verilerin doğruluğunun büyük önem taşıdığı unutulmamalıdır. Bazı durumlarda yanlış bilgiler, yanlış yönetim kararlarının verilmesine yol açacak ve test ekibinin güvenilirliğini zedeleyecektir.

Test Analisti risk bazlı test yaklaşımını kullanırken aşağıdaki etmenleri takip etmelidir:

- Test vasıtasıyla azaltılan riskler nelerdir
- Test vasıtasıyla azalmayacak riskler

Risk azaltımının takibi, genellikle testin tamamlanma durumunu da takip eden bir araç kullanılarak takip edilir (örn. test yönetim araçlar). Bu nedenle tanımlanan riskler, yürütülmesi ve başarılı olması durumunda riskleri azaltacak olan test senaryolarıyla eşlenmiş olan test koşullarıyla eşlenmelidir. Bu vasıtayla, test senaryoları güncellenirken risk azaltma bilgileri de otomatik olarak güncellenir. Bu işlem hem manuel hem de otomasyona geçirilmiş testler için yapılabilir.

Hata takibi, genellikle hata takip aracı kullanılarak gerçekleştirilir. Hatalar kayıt altına alınırken hataların her birine ilişkin sınıflandırma bilgileri de kaydedilir. Bu bilgiler yazılımın kalitesini ve testin ilerleyişini gösteren eğilimleri ve grafikleri üretmek amacıyla kullanılır. Sınıflandırma bilgileri, 'Hata Yönetimi' bölümünde daha ayrıntılı bir şekilde ele alınmıştır. Yazılım geliştirme yaşam döngüsü, hatalara ilişkin kayıt altına alınan bilgilerin miktarını ve bu bilgileri kaydetmek için kullanılan yöntemleri etkileyebilir.

Test yürütülürken test senaryosu durum bilgileri kayıt altına alınmalıdır. Bu işlem genellikle test yönetim aracı ile gerçekleştirilir ancak gerekli olması halinde manuel olarak da gerçekleştirilebilir. Test senaryolarına ilişkin tutulan bilgiler şunları içerebilir:

- Test senaryosunun oluşturulma durumu (örn., tasarlandı, gözden geçirildi)
- Test senaryosunun yürütülme durumu (örn., başarılı, başarısız, bloke, atlandı)
- Test senaryosu yürütme bilgisi (örn. tarih ve saat, test uzmanının adı, kullanılan veriler)
- Test senaryosu yürütme çıktıları (örn. ekran görüntüleri, ilgili kayıtlar)

Test senaryoları, tanımlanmış risk öğelerinde olduğu gibi, test etmekte oldukları gereksinimlerle eşlenmelidir. Test senaryosu A, gereksinim A ile eşlendiyse ve söz konusu test senaryosu, ilgili gereksinimle eşlenen tek senaryoysa test senaryosu A yürütüldüğünde ve başarılı olduğunda gereksinim A'nın da yerine getirildiği düşünülecektir.
Bu yaklaşım yeterli veya yetersiz olabilir. Çoğu durumda bir gereksinimi ayrıntılı bir şekilde test etmek için daha fazla sayıda test senaryosuna ihtiyaç duyulur ancak zaman kısıtlaması nedeniyle söz konusu testlerin ancak bir alt kümesi oluşturulur. Örneğin bir gereksinimin ayrıntılı bir şekilde test edilmesi için 20 test senaryosuna gerek duyulmasına rağmen sadece 10 test senaryosu oluşturulmuş ve koşturulmuş olabilir. Bu durumda aslında %50'lik kapsam elde edilmişken %100 elde edilmiş olarak kabul edilecektir. Gereksinimlerin ve kapsanma miktarının doğru biçimde takip edilmesi bir güven ölçütü olarak kullanılabilir.

Kaydedilecek bilgi miktarı (ve ayrıntı seviyesi), yazılım geliştirme yaşam döngüsü modeli de dahil olmak üzere çok sayıda etmene bağlıdır. Örneğin çevik yazılım geliştirme projelerinde, daha fazla yüz yüze iletişime ve ekibin yakın ilişki içinde bulunmasına bağlı olarak daha az durum bilgisi kaydedilecektir.

2.3 Dağıtılmış, Dış Kaynaklarla ve Test Ekibine Dahil Edilen Kaynaklarla Yapılan Test

Çoğu durumda testlerin tümü, tek bir yerde bulunan tek bir test ekibi tarafından gerçekleştirilmez. Testlerin birden fazla yerde gerçekleştirilmesi halinde söz konusu testlerin "dağıtılmış" olduğu söylenebilir. Testler tek bir yerde gerçekleştiriliyorsa merkezi olarak adlandırılabilir. Testler, bir ya da daha fazla yerde şirket çalışanları dışında kişiler tarafından tamamen şirket test ekiplerinden bağımsız bir şekilde gerçekleştiriliyorsa söz konusu test çalışmala rının dış kaynak kullanımıyla yapıldığı söylenebilir. Testler, şirket proje ekibiyle aynı yerde bulunan farklı kişiler tarafından gerçekleştirilirse söz konusu testlerin test ekibine dahil edilen kaynaklarla yapıldığı ifade edilebilir.
Test ekibinin bir kısmının birden fazla yere ve hatta birden fazla şirkete dağıldığı projeler üzerinde çalışırken test analistinin, etkin iletişim ve bilgi aktarımına özel ilgi göstermesi gerekir. Bazı kurumlar, "24 saat test" modeliyle çalışmaktadır. Bu modelde, belirli bir saat dilimindeki ekip, testin 24 saat boyunca devam edebilmesi için testi başka bir saat dilimindeki ekibe devreder. Bu işlem, işleri devralacak olan test analistinin özel bir planlama yapmasını gerektirir. İyi planlama, sorumlulukların anlaşılması ve doğru bilgilere erişebilmek açısından önemlidir.
Sözlü iletişimin mümkün olmadığı durumlarda yazılı iletişim yeterli olmalıdır. Bu, e-postaların, durum raporlarının, test yönetimi ve hata takip araçlarının etkin bir şekilde kullanılmasını gerektirir. Test yöne timi aracı testlerin kişilere atanmasını mümkün kılıyorsa aynı zamanda bir planlama aracı ve insanlar arasında işlerin aktarımı açısından kullanımı kolay bir yöntem görevini de görebilir. Doğru şekilde kayıt altına alınan hatalar, gerekli olduğu hallerde takip edilmeleri amacıyla diğer ekip üyelerine aktarılabilir.

2.4 Risk Bazlı Testlerde Test Analistinin Görevleri

2.4.1 Genel Bakış

Test yöneticisi, risk-bazlı test stratejilerinin oluşturulması ve yönetilmesinden genel anlamda sorumludur. Test yöneticisi, risk bazlı yaklaşımın doğru bir şekilde uyarlandığından emin olmak için test analistinden sürece katılmasını isteyebilir. Test analisti, aşağıdaki risk-bazlı test görevlerinde aktif olarak rol almalıdır:

- Risk tanımlama
- Risk değerlendirme
- Risk azaltma

Söz konusu görevler, ortaya çıkan riskleri, değişen öncelikleri ele almak ve risk durumunu düzenli aralıklarla değerlendirmek ve rapor etmek için projenin yaşam döngüsü boyunca tekrarlamalı olarak gerçekleştirilmelidir.

Test analistleri, test yöneticisi tarafından hazırlanan risk bazlı test çerçevesi dahilinde çalışmalıdır. Emniyete, işletmeye, ekonomik endişelere ve siyasi etmenlere ilişkin riskler gibi projenin arz ettiği işletme alanındaki riskler konusundaki bilgi birikimiyle de projeye katkıda bulunmalıdır.

2.4.2 Risk Tanımlama

Risk tanımlama sürecine paydaşlardan en geniş katılım sağlandığı zaman çok sayıda risk tespit edilecektir. Test analistleri, genellikle test edilmekte olan sistemle ilgili iyi bir bilgi birikimine sahip olduğundan kullanıcılarla ve paydaşlarla görüşmeler yapmak için en uygun kişilerdir. Bu kapsamda, bağımsız değerlendirmeler yapar, risk şablonlarının kullanımını ve değerlendirilmesini sağlar, risk atölyeleri düzenler, olası ve mevcut kullanıcılarla beyin fırtınası yapar, test kontrol listelerini tanımlar ve benzer sistem veya projelere yönelik deneyimlerinden yararlanır. Test analisti, test sırasında ele alınması gereken iş riski alanlarını belirlemek üzere özellikle kullanıcılarla ve diğer alan uzmanlarıyla yakın iletişim içinde çalışmalıdır. Test analisti, risklerin kullanıcılar ve paydaşlar üzerindeki olası risklerini tanımlamada da yardımcı olabilir. Projede tanımlanabilecek örnek risklerden bazıları şunlardır:

- Yazılımın fonksiyonalitesindeki doğrulukla ilgili sorunlar, örn. yanlış hesaplamalar
- Kullanılabilirlik sorunları, örn. yetersiz sayıda klavye kısayolu
- Öğrenilebilirlik sorunları, örn. önemli karar noktalarında kullanıcı yönlendirmelerinin yetersizliği

Belirli kalite karakteristiklerinin test edilmesine yönelik göz önünde bulundurulması gereken konular bu ders programının 4. Bölümünde ele alınmıştır.

2.4.3 Risk Değerlendirme

Risk tanımlama, mümkün olduğunca çok sayıda ilgili riskin belirlenmesi süreciyken risk değerlendirmesi, tanımlanan riskler üzerine yapılan çalışmalarla ilgilidir. Risklerin her birinin sınıflandırılması ve risklerin her birinin gerçekleşme olasılığının ve bunun etkilerinin belirlenmesini amaçlar.

Risk seviyesinin belirlenmesi için risk öğelerinin her birinin ortaya çıkma olasılığının ve bunların olası etkilerinin değerlendirilmesi gerekir. Ortaya çıkma olasılığı sistem canlıdayken potansiyel bir problemin ortaya çıkması olasılığı olarak yorumlanır. Diğer bir deyişle teknik risklerden kaynaklanır. Teknik test analisti, risk öğelerinin her birine ilişkin olası teknik risk seviyesinin tespit edilmesine ve anlaşılmasına katkı sağlarken test analisti, problemin ortaya çıkması durumunda iş ve müşteri üzerindeki olası etkilerinin anlaşılmasına katkıda bulunmaktadır.

Ortaya çıkmasının ardından görülen etki, sıklıkla kullanıcılar, müşteriler veya diğer paydaşlar üzerinde görülen etkinin ciddiyeti olarak yorumlanır. Diğer bir deyişle iş ile ilgili veya operasyonel risklerden kaynaklanır. Test Analisti, risk öğelerinin her birinin etkisinin hangi alanlarda görüleceğinin ve kullanıcıya olan etkisinin ne olacağının belirlenmesine yönelik çalışmalara katkıda bulunmalıdır. İşletmeyle ilgili riskleri etkileyen etmenlerden bazıları şunlardır:

- Etkilenen özelliğin kullanım sıklığı
- İş kaybı
- Olası finansal, ekolojik ya da sosyal kayıplar veya yükümlülükler

- Hukuki veya cezai yasal yaptırımlar
- Emniyetle gereksinimler
- Cezalar, lisans kayıpları
- Makul geçici çözüm yollarının bulunmaması
- Etkilenen özelliğin görülebilirliği
- Hatanın itibar kaybına neden olup olmayacağı
- Müşteri kaybı

Test analistinin risklere ilişkin mevcut verileri ve test yöneticisi tarafından ortaya koyulan çerçeveyi dikkate alarak işletme riski seviyelerini belirlemesi gerekir. Bunlar, terimlerle (örn. düşük, orta, yüksek) veya sayılarla sınıflandırılabilir. Risk seviyesini tanımlı bir ölçek üzerinde nesnel olarak ölçmenin mümkün olmadığı durumlarda belirlenen seviye nicel bir ölçüm verisi olmayacaktır. Riskin ortaya çıkma ihtimalini ve maliyetini/sonuçlarını isabetli bir biçimde ölçmek çok zor olduğunda riskin seviyesi genellikle nitel olarak belirlenir.

Risk seviyelerinin tanımlanmasında sayıların kullanılması riskin seviyesinin nicel olduğu anlamına gelmez. Örneğin test yöneticisi, işletme riskinin 1 ila 10 arasında değişen bir değerle sınıflandırılması gerektiğini belirtebilir. Buradaki en yüksek değer olan 1, işletme açısından arz ettiği risk ve etkileri en yüksek olan değer olacaktır. Olasılık (teknik riskin değerlendirilmesi) ve etki (işletme riskinin değerlendirilmesi) belirlendikten sonra söz konusu değerler, risk öğelerinin her birine yönelik genel risk puanını belirlemek üzere toplanabilir. Bunun ardından genel puan risk azaltma faaliyetlerini öncelik sırasına koymak için kullanılır. PRISMA® [vanVeenendaal12] gibi risk bazlı test modellerinden bazıları olasılık ve etki seviyelerini bir araya getirmeyip ayrı ayrı ele alarak teknik risklerin ve işletme risklerinin ayrı ayrı ele alınmasına olanak verir.

2.4.4 Risk Azaltma

Test analistleri proje kapsamında aşağıdaki faaliyetleri yerine getirmelidir:

- Test öğelerinin başarılı olup olmadıklarını net bir şekilde gösteren iyi tasarlanmış test senaryolarını kullanarak ve gereksinimler, tasarımlar ve kullanıcı dokümantasyonu gibi proje çıktılarının gözden geçirilmesine katılarak ürün riskini düşürmelidir
- Test stratejisinde ve test planında tanımlanan uygun risk azaltma faaliyetlerini yerine getirmelidirler.
- Proje ilerledikçe toplanan ek bilgilere dayanarak mevcut riskleri yeniden değerlendirmeli, gerektiği yerlerde olasılık ve etki seviyelerini güncellemelidir.
- Test sırasında elde edilen veriler ışığında yeni riskleri ön görmelidirler.

Test yapmak, ürün (kalite) risklerini azaltma yöntemlerden biridir. Hataları bulan test uzmanları, hataları yazılım canlıya alınmadan önce ele almak için hatalar ve fırsatlar konusunda bilinç yaratarak riski azaltır. Test uzmanlarının herhangi bir hata bulamaması durumunda test, belirli koşullar altında (diğer bir deyişle, test edilen koşullar altında) sistemin doğru şekilde çalıştığını varsayarak riski azaltır. Test analistleri test verisinin doğru toplanmasını, gerçekçi kullanım senaryolarının belirlenip koşulmasını ve kullanılabilirliği artırmaya yönelik çalışmaların gerçekleştirilmesini sağlayarak olası risk azaltma yöntemlerinin belirlenmesine yardımcı olurlar.

2.4.4.1 Testlerin Önceliklendirilmesi

Risk seviyesi, testlerin önceliklendirilmesinde de kullanılır. Örneğin, test analisti, bir muhasebe sisteminin testleri sırasında işlem doğruluğu alanında yüksek risk bulunduğunu tespit edebilir. Sonuç olarak test uzmanı, riski azaltmak amacıyla yürütülebilecek ve doğruluğu onaylanabilecek güçlü bir veri kümesi elde etmek için diğer iş alanı uzmanlarıyla birlikte çalışabilir. Benzer şekilde test analisti, yeni bir ürünün en belirgin risklerinden birinin kullanılabilirlik ile ilgili sorunlar olduğunu tespit edebilir. Bu durumda sorunların kullancı kabul testlerinden tespit edilmesini beklemek yerine entegrasyon testi seviyesinde kullanılabilirlik testlerine öncelik verebilir. Test planının aksamaması için bu önceliklendirmeye mümkün olduğunca erken karar verilmelidir.

Bazı durumlarda en yüksek risk taşıyan testlerin tümü düşük risk taşıyanlardan önce koşulur ve testler kati bir sıralamayı takip eder (bu yöntem genellikle "önce derinlik" olarak adlandırılır). Diğer durumlardaysa koşulacak testlerin belirlenmesinde örnekleme bazlı bir yaklaşım kullanılır. Bu yaklaşımda testler aynı seviyede hiçbir riski açıkta bırakmayacak şekilde, risk seviyesini bir önceliklendirme aracı olarak kullanarak seçilir (bu yönt em genellikle "önce genişlik" olarak adlandırılır).

Risk bazlı testin, "önce derinlik" ya da "önce genişlik" mantığına dayanarak gerçekleştirilmesi önemli olmaksızın teste ayrılan zamanın, tüm testler gerçekleştirilmeden son bulması mümkündür. Risk bazlı test, bu noktada test uzmanlarının yöneticilere geriye kalan testlerin ne kadar risk taşıdığına ilişkin bilgi verebilmesini sağlar. Yöneticiler bu bilgiyi kullanarak test süresinin uzatılmasına ya da riskin kullanıcılara, müşterilere, yardım masasına/teknik destek birimine aktarılmasına karar verebilir.

2.4.4.2 Testin Sonraki Test Döngülerinde Yeniden Düzenlenmesi

Risk değerlendirmesi, testlerin başlamasından önce bir kez gerçekleştirilen bir aktiviteden ziyade bir süreçtir. Planlanan test döngüleri aşağıdaki etmenler ele alınarak yeniden bir risk analizine tabi olacaktır:

- Ürün risklerinde ortaya çıkan yeni veya bariz bir risk
- Test sırasında tespit edilen tutarsız ya da hatalara açık alanlar
- Çözülen hatalardan kaynaklanabilecek riskler
- Test sırasında bulunan tipik hatalar
- Yeterince test edilmemiş alanlar (düşük test kapsamı)

Test için ek süre tahsis edilirse risk kapsamasının daha düşük risk içeren alanlara doğru genişletilmesi mümkün olabilir.

3. Test Teknikleri – 825 dak.

Anahtar Sözcükler

sınır değer analizi (BVA), neden-sonuç grafiği, kontrol listesine dayalı test etme, sınıflandırma ağacı yöntemi, kombinasyonlu test, karar tablosu testi, hata sınıflandırması, hata bazlı teknik, alan analizi, hata tahminleme, denklik paylarına ayırma, tecrübeye dayalı teknik, keşif testi, dikey dizi, dikey dizi testi, ikili test, gereksinim bazlı test, spesifikasyon bazlı teknik, durum geçişi testi, test başlatma belgesi, kullanım senaryosu testi, kullanıcı hikayesi testi

Test Teknikleri için Öğrenme Hedefleri

3.2 Spesifikasyon Bazlı Teknikler

TA-3.2.1 (K2) Neden-sonuç grafiklerinin kullanımını açıklayın

TA-3.2.2 (K3) Tanımlanan kapsam seviyesine ulaşmak için denklik paylarına ayırma test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.3 (K3) Tanımlanan kapsam seviyesine ulaşmak için sınır değer analizi test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.4 (K3) Tanımlanan kapsam seviyesine ulaşmak için karar tablosu test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.5 (K3) Tanımlanan kapsam seviyesine ulaşmak için durum geçişi test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.6 (K3) Tanımlanan kapsam seviyesine ulaşmak için ikili test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.7 (K3) Tanımlanan kapsam seviyesine ulaşmak için sınıflandırma ağacı test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.8 (K3) Tanımlanan kapsam seviyesine ulaşmak için kullanım senaryosu test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.9 (K2) Kullanıcı hikayelerinin, çevik yazılım geliştirme projelerindeki testlere nasıl rehberlik ettiğini açıklayın TA-3.2.10 (K3) Tanımlanan kapsam seviyesine ulaşmak için alan analizi test tasarım tekniğini kullanarak belirli bir spesifikasyon öğesinden yararlanarak test senaryoları yazın

TA-3.2.11 (K4) Bulunacak olası hata türlerini belirlemek ve uygun spesifikasyon bazlı teknikleri belirlemek için bir sistemi veya gereksinim spesifikasyonlarını analiz edin

3.3 Hata Bazlı Teknikler

TA-3.3.1 (K2) Hata bazlı test tekniklerinin uygulamalarını açıklayın ve bunların kullanım alanlarıyla spesifikasyon bazlı tekniklerin kullanım alanları arasındaki farkları belirtin

TA-3.3.2 (K4) Belirli bir hata sınıflandırmasının belirli bir durumdaki uygulanabilirliğini iyi sınıflandırma ölçütlerini kullanarak analiz edin.

3.4 Tecrübeye Dayalı Teknikler

TA-3.4.1 (K2) Tecrübeye dayalı tekniklerin ilkelerini, spesifikasyon bazlı ve hata bazlı tekniklerle kıyaslayarak avantajlarını ve dezavantajlarını açıklayın.

TA-3.4.2 (K3) Belirli bir senaryoda keşif testlerini ve sonuçların nasıl rapor edilebileceğini açıklayın TA-3.4.3 (K4) Belirli bir projede hedeflere ulaşmak için spesifikasyon bazlı, hata bazlı ya da tecrübeye dayalı tekniklerin, hedeflere ulaşmak için nasıl kullanılmaları gerektiğini belirleyin

3.1 Giriş

Bu bölümde göz önünde bulundurulan test tasarım teknikleri, aşağıdaki kategorilere ayrılmıştır:

- Spesifikasyon bazlı (ya da davranış bazlı ya da kara kutu)
- Hata bazlı
- Tecrübeye dayalı

Söz konusu teknikler, birbirlerini tamamlar niteliktedir ve gerçekleştirilmekte olduğu test seviyesi önemli olmaksızın herhangi bir test faaliyetinde kullanılabilir.

Tekniklere ilişkin üç kategorinin de hem fonksiyonel hem de fonksiyonel olmayan kalite karakteristiklerini test etmekte kullanılabileceği unutulmamalıdır. Fonksiyonel olmayan karakteristiklere ilişkin test bir sonraki bölümde ele alınmıştır.

Bu bölümlerde ele alınan test tasarım teknikleri, esasen optimum test verilerinin belirlenmesine (örn. denklik payları) ya da test sıralamalarının elde edilmesine (örn. durum modelleri) odaklanabilir. Bütüncül test senaryolarının oluşturulması için tekniklerin bir araya getirilmesi de yaygın olarak kullanılan bir yöntemdir.

3.2 Spesifikasyon Bazlı Teknikler

Spesifikasyon bazlı teknikler, yazılımın iç çalışma mantığına herhangi bir referansta bulunmadan test esasının analizine dayanılarak test senaryolarının elde edildiği test tekniklerdir.

Spesifikasyon bazlı tekniklerin yaygın özellikleri şöyledir:

- Durum geçişi şemaları ve karar tabloları gibi modeller, test tekniği doğrultusunda test tasarımı sırasında oluşturulur.
- Test koşulları, söz konusu modellerden sistematik bir şekilde elde edilir.

Bazı teknikler, test tasarımı ve test yürütme faaliyetlerinin ilerleyişini ölçmek için kullanılan kapsam kriterlerinin belirlenmesinde de kullanılmaktadır. Kapsam kriterinin tamamen yerine getirilmiş olması yapılacak başka bir test kalmadığı anlamına gelmemektedir. Bunun yerine, bu kapsam kriteri kullanılarak test kapsamında artış sağlanamayacağı anlamına gelir.

Spesifikasyon bazlı teknikler, genellikle sistem gereksinimleri dokümanlarına dayanır. Gereksinim spesifikasyonları sistemin nasıl çalışacağını tanımladığından özellikle fonksiyonalite alanındaki testlerin gereksinimlerden elde edilmesini sağlar. Bazı durumlarda dokümante edilmiş herhangi bir gereksinim bulunmayabilir ancak daha önce sistem üzerinde yapılmış projelerden dolaylı gereksinimler elde edilebilir.

Çok sayıda spesifikasyon bazlı test tekniği bulunmaktadır. Söz konusu teknikler, farklı yazılım ve senaryoları hedef almaktadır. Aşağıdaki bölümler, tekniklerin her birinin uygulanabilirliğini, test analistinin karşılaşabileceği bazı kısıtlamaları ve zorlukları, test kapsamının ölçüldüğü yöntemi ve hedef alınan hata türlerini göstermektedir.

3.2.1 Denklik Paylarına Ayırma

Denklik paylarına ayırma (Equivalence Partitioning), girdilerin, çıktıların, dahili değerlerin ve zamana bağlı değerlerin ele alınışını etkin bir şekilde test ederken gerekli test senaryolarının sayısını azaltmak için kullanılır. Paylara ayırma, sistem tarafından aynı şekilde ele alınan değer kümelerinin yani denklik sınıflarının (sıklıkla denklik payı olarak da adlandırılır) bulunmasını hedefler. Bir paydan temsilen bir değerin seçilmesiyle aynı payda yer alan tüm öğelerin kapsam dahiline alındığı varsayılır.

Uygulanabilirlik

Bu teknik, testin herhangi bir aşamasında uygulanabilir. Test edilecek değer kümesindeki tüm öğelerin aynı şekilde ele alınması uygun olduğunda ve yazılım tarafından kullanılan değer kümeleri birbirleriyle etkileşim kurmadığı durumlarda kullanılabilir. Değer kümelerinin seçimi, yazılım gereksinimlerine göre geçerli ve geçersiz sayılan payların her ikisine de uygulanabilir.

Bu teknik, payların sınırlarındaki değerleri kapsamak amacıyla test değerlerini genişleten sınır değer analiziyle birlikte kullanıldığında en güçlü haline ulaşır. Temel fonksiyonalitenin çalışıp çalışmadığını hızla belirlediği için yeni bir yazılımın ya da yeni bir sürümün alım testi (duman testi) sırasında sıklıkla kullanılan bir tekniktir.

Sınırlamalar/Zorluklar

Varsayımın yanlış olması ve paydaki değerlerin yazılım tarafından tam olarak aynı şekilde ele alınmaması durumunda bu test, hataların atlanmasına neden olabilir. Buna ek olarak payların dikkatle seçilmesi çok önemlidir. Örneğin, pozitif ve negatif sayıları kabul eden bir girdi alanı iki geçerli pay halinde test edilmelidir. Farklı ele alınma ihtimalleri nedeniyle bunlardan biri pozitif sayılar için test edilirken diğeri negatif sayılar için test edilmelidir. Sıfır değerine izin verilmesine ya da verilmemesine bağlı olarak bu da ayrı bir pay oluşturabilir. Test analistinin, değerlerin en iyi şekilde paylara ayrılması için temelde yatan işlemi anlaması çok önemlidir.

Kapsam

Kapsam, test edilen payların sayısının tanımlanan pay sayısına bölünmesiyle elde edilir. Tek bir paydan birden fazla değerin kullanılması, kapsam yüzdesini artırmaz.

Hata Türleri

Bu teknik sayesinde fonksiyonel hatalar bulur.

3.2.2 Sınır Değer Analizi

Sınır Değer Analizi (Boundary Value Analysis), denklik paylarının sınırlarında bulunan değerleri test etmek için kullanılır. İki sınır değer analizi yaklaşımı bulunmaktadır: iki değerli ve üç değerli testler. İki değerli testlerde sınır değeri (sınır üzerindeki) ve sınırdan hemen sonraki değer (mümkün olan en küçük artırımla) kullanılır. Örneğin pay, 1 ila 10 arasındaki değerleri 0.5 artırımla kapsıyorsa üst sınıra yönelik iki test değeri 10 ve 10.5 olacaktır. Alt sınıra yönelik test değerleri ise 1 ve 0.5 olacaktır. Sınırlar, tanımlanan denklik payındaki maksimum ve minimum değerler tarafından tanımlanır.

Üç değerli sınır testinde ise sınırdan önceki, sınırdaki ve sınırdan sonraki değerler kullanılır. Bir önceki örnekte üst sınır testi değerleri 9.5, 10 ve 10.5 olacaktır. Alt sınıra yönelik test değerleri ise 1.5, 1 ve 0.5 olacaktır. İki ya da üç sınır değeri kullanma kararı, test edilmekte olan öğeyle ilgili risklere göre değişir. Üç değerli sınır yaklaşımı, daha yüksek risk arz eden öğeler için kullanılır.

Uygulanabilirlik

Bu teknik, testin her aşamasında uygulanabilir ve denklik payları çıkarıldıktan sonra kullanıma alınabilir. Sınır üzerinde ve sınırın dışında olma kararını verebilmek için denklik paylarına ihtiyaç duyulur. Örneğin, bir sayı aralığı bir paydır. Bazı denklik paylarında sınır değer kavramı nlamını yitirmektedir. Örneğin dikdörtgen nesnelerden oluşan bir payın sınır değeri bulunmamaktadır. Sınır değer analizi, sayısal aralıklara ek olarak aşağıdakilere de uygulanabilir:

- Sayısal olmayan değişkenlerin sayısal özellikleri (örn. uzunluk)
- Döngüler
- Depolanan veri yapıları
- Fiziki nesneler (bellek dahil olmak üzere)
- Zamana bağlı faaliyetler

Sınırlamalar/Zorluklar

Bu tekniğin doğruluğu denklik paylarının doğru tanımlanmasına bağlı olduğundan aynı sınırlamalara ve zorluklara tabidir. Test analisti, test edilecek değerleri doğru bir şekilde belirlemek için geçerli ve geçersiz değerlerdeki artımların bilincinde olmalıdır. Talep edilen paylar, sınır değer analizinde kullanılabilir ancak geçerli girdi aralıklarıyla sınırlı değildir. Örneğin, bir veri alanı tarafından desteklenen basamak sayısı test edilirken izin verilen maksimum basamak sayısını içeren bir pay (sınır) ve maksimum değerin bir sayı üzerinde başlayan başka bir pay (sınırın dışında) bulunmaktadır.

Kapsam

Kapsam, test edilen sınır değerlerinin tanımlanan toplam sınır değerlerinin sayısına bölünmesiyle belirlenir (iki değerli ya da üç değerli yöntemlerden biri kullanılarak). Bu hesaplama sınır testi için kapsam yüzdesini belirleyecektir.

Hata Türleri

Sınır değer analizi, sınırlarda hata olup olmadığını veya sınırda herhangi bir eksiklik olup olmadığını güvenilir bir şekilde tespit eder ve ekstra sınırların bulunduğu durumları da algılayabilir. Bu teknik, olması gerekenden daha küçük ve daha büyük olan hatalar başta olmak üzere sınır değerlerinin kullanılması sırasında ortaya çıkabilecek hataları tespit eder (örn. sınırın yerinde bulunmaması). Buna ek olarak yük sınırlarının toleransı gibi fonksiyonel olmayan hataların bulunması için de kullanılabilir (örn. sistem aynı anda 10.000 kullanıcıyı desteklemektedir).

3.2.3 Karar Tabloları

Karar tabloları, koşullar arasındaki kombinasyonları test etmek için kullanılır. Karar tabloları, ilgili koşul kombinasyonlarının tümünü ve test edilmekte olan sistem tarafından işlenen tüm olası kombinasyonların tümünü onaylamak için kullanılabilecek bir yöntemdir. Karar tablosu testinin amacı, tüm koşul, ilişki ve kısıtlamaların kombinasyonlarının test edilmesini sağlamaktır. Tüm olası kombinasyonları test etmeye çalışırken karar tabloları çok büyüyebilir. Kombinasyonların sayısını tüm olası öğelerden sadece "ilgi çekici" öğelere indirgemek için testi hızlandırabilecek yöntemlerden biridir ve indirgenmiş karar tablosu testi olarak adlandırılır. Bu teknik kullanıldığı zaman test farklılık gösteren çıktıları oluşturan koşullara indirgenmektedir. Koşul kombinasyonlarının mümkün olmadığı gereksiz test veya testler kaldırılır. Karar tablosunu bütünüyle kullanmak ya da indirgenmiş karar tablolarını kullanmak konusundaki karar genellikle risklere dayanarak verilir. [Copeland03]

Uygulanabilirlik

Bu teknik yaygın bir şekilde entegrasyon, sistem ve kabul testi seviyelerinde uygulanır. Koda bağlı olarak, bir karar mantığı kümesinin sorumlusunun bir bileşen olması durumunda birim testinde de kullanılabilir. Gereksinimlerin akış tablosu ya da iş kuralı tabloları halinde sunulması durumunda bu tekniğin kullanımı avantaj sunmaktadır. Karar tabloları da bir gereksinim tanımlama tekniğidir ve bazı gereksinimler bu formatta temin edilebilir. Gereksinimlerin tablo ya da akış şeması halinde temin edilemediği durumlarda koşul kombinasyonları genellikle anlatı formatındadır. Karar tablolarını hazırlarken tanımlanan koşul kombinasyonlarının yanı sıra mevcut olan ancak açık bir şekilde ifade edilmemiş olanların da göz önünde bulundurulması önemlidir. Geçerli bir karar tablosu hazırlamak için test uzmanının, tüm koşul kombinasyonlarına yönelik beklenen tüm çıktıları gereksinimden ya da bilinen bir test sonucundan elde etmesi gerekir. Etkileşim içindeki tüm koşullar göz önünde bulundurulduğunda karar tablosu iyi bir test aracı olarak kullanılabilir.

Sınırlamalar/Zorluklar

Etkileşim içindeki tüm koşulların bulunması, özellikle gereksinimlerin açık bir şekilde tan ımlanmadığı ya da mevcut olmadığı durumlarda zor olabilir. Koşul kümeleri hazırlandıktan sonra beklenen sonucun bilinmediği kombinasyonların ortaya çıkması sıklıkla karşılaşılan bir durumdur.

Kapsam

Karar tablosunda minimum test kapsamı, sütunların her birine yönelik bir test senaryosunun oluşturulmasıyla elde edilebilir. Burada, herhangi bir bileşik koşul olmadığı ve tüm olası koşul kombinasyonlarının sütuna kaydedildiği varsayılır. Karar tablosu kullanılarak testlere yönelik karar verirken, test edilmesi gereken sınır koşullarını da göz önünde bulundurmak çok önemlidir. Söz konusu sınır koşulları, yazılımı doğru bir şekilde test etmek için gerekli test senaryosu sayısını artırabilir. Sınır değer analizi ve denklik paylarına ayırma, karar tablosu tekniğini tamamlayıcı niteliktedir.

Hata Türleri

Beklenmedik sonuçlara yol açan koşul kombinasyonlarına dayanarak yapılan yanlış işlemler, tipik hatalar arasında yer almaktadır. Karar tablolarının oluşturulması sırasında analiz dokümanlarında hatalar bulunabilir. En sık karşılaşılan hata türü, eksiklikler (belirli bir durumda olması gerekenler hakkında herhangi bir bilgi bulunmaması) ve çelişkilerdir. Test, ele alınmayan ya da doğru bir şekilde ele alınmayan koşul kombinasyonlarındaki sorunları da bulabilir.

3.2.4 Neden-Sonuç Grafiği

Neden-sonuç grafikleri, kullanıcı hikayeleri veya akış tabloları gibi bir programın fonksiyonel mantığını (diğer bir deyişle, kurallarını) tanımlayan herhangi bir kaynaktan oluşturulabilir. Programın mantıksal yapısına grafiksel bir genel bakış kazanmak açısından faydalıdır ve karar tablolarının yaratılmasına temel teşkil ederler. Kararların neden-sonuç grafiklerinde ve/veya karar tablolarında ele alınması test kapsamının sistematik bir şekilde gerçekleştirilmesini sağlar.

Uygulanabilirlik

Neden-sonuç grafikleri, karar tablolarıyla aynı durumlarda ve aynı test seviyelerinde uygulanır. Bir neden-sonuç grafiği, özellikle sonuçlara yol açan (nedensellik) durum kombinasyonlarını, sonuçları hariç tutan koşul kombinasyonlarını (değil), bir sonuç oluşturmak için doğru olması gereken birden fazla koşulu (ve) ve belirli bir sonucu oluşturmak için doğru olabilecek alternatif koşulları (ya da) gösterir. Söz konusu ilişkilerin, öyküsel bir anlatıdan ziyade, bir neden-sonuç grafiğinde gösterilmesi daha kolaydır.

Sınırlamalar/Zorluklar

Diğer test tasarım teknikleriyle kıyaslandığında neden-sonuç grafiklerinin öğrenilmesi daha fazla zaman ve çaba gerektirir. Buna ek olarak araç desteği gerektirir. Neden-sonuç grafikleri hazırlayan ve okuyan tarafından anlaşılması gereken belirli notasyonlara sahiptir.

Kapsam

Minimum kapsamın elde edilmesi için kombinasyon koşulları dahil olmak üzere etki hattının olası nedenlerinden her biri test edilmelidir. Neden-sonuç grafikleri, veri üzerindeki kısıtlamaları ve mantık akışındaki kısıtlamaları tanımlayacak bir yöntemdir.

Hata Türleri

Söz konusu grafikler, karar tabloları tarafından da bulunan kombinasyonlu hataları tespit eder. Buna ek olarak grafiklerin oluşturulması, test esasında yeterli detay seviyesinin elde edilmesini sağlar; bu doğrultuda, test esasının kalitesinin ve ayrıntı seviyesinin artırılmasına yardımcı olur ve test uzmanının eksik gereksinimleri tanımlamasını sağlar.

3.2.5 Durum Geçişi Testi

Durum geçişi tekniği, geçerli ve geçersiz geçişler vasıtasıyla yazılımın tanımlı durumlara girme ve tanımlı durumlardan çıkma becerisini test etmek için kullanılır. Olaylar, yazılımın bir durumdan diğerine geçişine ve çeşitli faaliyetlerde bulunmasına yol açar. Olaylar, geçiş yolu seçimini etkileyen koşullardan (kimi zaman koruyucu koşullar ya da geçiş koşulları olarak adlandırılır) etkilenebilir. Geçerli kullanıcı adı/parola kombinasyonuna sahip bir oturum açma olayı, geçersiz parolaya sahip bir oturum açma olayından farklı bir geçiş gösterecektir.

Durum geçişleri, grafik formatında durumlar arasındaki geçerli geçişlerin tümünü gösteren durum geçişi şeması şeklinde ya da hem geçerli hem de geçersiz tüm olası geçişleri gösteren durum tablosu şeklinde takip edilebilir.

Uygulanabilirlik

Durum geçişi testi, tanımlı durumları ve söz konusu durumlar arasında geçişe neden olacak olayları (örn. değişen ekranlar) olan tüm yazılımlara uygulanabilir. Durum geçişi testi, farklı test seviyelerine yönelik kullanılabilir. Gömülü yazılım, web ve işleme dayalı yazılımlar, bu test tekniği açısından en ideal adaylardır. Kontrol sistemleri, diğer bir deyişle trafik ışığı kontrol birimleri de bu test tekniği açısından ideal adaylardır.

Sınırlamalar/Zorluklar

Durumların belirlenmesi, durum tablosunun ya da şemasının tanımlanması sürecin en z orlu kısmıdır. Yazılımda bir kullanıcı arayüzü bulunuyorsa kullanıcıya yönelik görüntülenen çok sayıda ekran, yaygın olarak durumları tanımlamak için kullanılır. Gömülü yazılımlarda durumlar, donanımın karşılaşacağı durumlara bağlı olabilir.

Durumların kendilerine ek olarak durum geçişi testinin temel birimi, 0-anahtarı olarak bilinen bağımsız geçiştir. Tüm geçişlerin test edilmesi, bazı durum geçişi hatalarının tespit edilmesini sağlayacaktır ancak işlemler ardışık test edildiğinde tespit edilen hata sayısı artabilir. Birbirini takip eden iki geçişin ardışık test edilmesi 1-anahtarı olarak adlandırılırken, birbirini takip eden üç geçişin test edilmesi 2-anahtarı olarak adlandırılır. (Söz konusu anahtarlar kimi zaman N-1 anahtarları olarak adlandırılır. Burada N, kat edilen geçiş sayısını temsil etmektedir.) Örneğin tek bir geçiş (0-anahtarı) olacaktır. [Bath08]

Kapsam

Diğer test tekniklerinde de olduğu gibi test kapsamı seviyeleri arasında bir hiyerarşi bulunur. Kabul edilebilir minimum kapsam, her durumun ve geçişin sağlanması şeklindedir. %100 geçiş kapsamı (%100 0 -anahtarı kapsamı ya da %100 mantıksal dal kapsamı olarak da bilinir), sistem tasarımını ya da durum geçişi modelinin (şemasının ya da tablosunun) kusurlu olmaması şartıyla her durumun ziyaret edi ldiğine ve her geçişin denendiğine ilişkin bir garanti niteliğindedir. Durumlar ve geçişler arasındaki ilişkilere bağlı olarak diğer geçişleri tek bir defa yürütmek üzere bazı geçişlerin bir kereden fazla sınanması gerekebilir.

"n-anahtarı kapsamı" terimi, kapsam dahilindeki geçişlerin sayısına ilişkindir. Örneğin, %100 1-anahtarı kapsamının elde edilmesi için birbirini takip eden iki geçişin ardışık olarak en az bir defa test edilmiş olmasıdır. Bu test, %100 0-anahtarı kapsamının kaçıracağı arıza türlerinden bazılarını da tetikleyecektir.

"Dairesel kapsam", ardışık geçişlerin döngü oluşturduğu durumlarda uygulanır. Herhangi bir duruma ilişkin döngülerin, test edilen aynı duruma dönmesiyle %100 dairesel kapsam elde edilir. Bu, döngüde yer alan tüm durumlar için test edilmelidir.

Bu yaklaşımların tümünde daha yüksek seviyeli bir kapsam elde etmek için tüm geçersiz geçişler kapsam dahiline alınmalıdır.

Kapsam gereksinimleri ve durum geçişi testine yönelik kümeler, geçersiz geçişlerin bulunup bulunmadığını tanımlamalıdır.

Hata Türleri

Yanlış işlem, yanlış ya da desteklenmeyen geçişler, çıkışı bulunmayan durumlar ve mevcut olmayan durumlar tipik hatalar arasında yer almaktadır. Durum makinesi modelinin oluşturulması sırasında spesifikasyon dokümanında hatalar tespit edilebilir. En sık karşılaşılan hata türü, eksiklikler (belirli bir durumda olması gerekenler hakkında herhangi bir bilgi bulunmaması) ve çelişkilerdir.

3.2.6 Kombinasyonlu Test Teknikleri

Kombinasyonlu test her biri çok sayıda değer içeren çeşitli parametrelere sahip yazılımlar test edilirken kullanılır. Parametreler bağımsız olmalı ve herhangi bir etmene dair herhangi bir seçeneğin, diğer etmenin herhangi bir seçeneğiyle bir arada kullanılabilmesi açısından uyumlu olmalıdır. Sınıflandırma ağaçları, belirli seçeneklerin uyumsuz olması durumunda bazı kombinasyonların hariç tutulmasına izin verebilir. Bu, bir araya getirilen etmenlerin birbirini etkilemeyeceği anlamına gelmez; birbirlerini etkileyebilirler ancak bu, kabul edilebilir şekilde olmalıdır.

Kombinasyonlu test, tanımlanan kapsam seviyesine ulaşmak açısından söz konusu kombinasyonlardan oluşan uygun bir alt kümenin tanımlanması için kullanılabilecek bir yöntemdir. Kombinasyonlara eklenecek öğelerin sayısı, tekil öğeler, ikililer, üçlüler ve diğerleri dahil olmak üzere test analisti tarafından seçilebilir [Copeland03]. Test analistine bu görev kapsamında yardımcı olacak çok sayıda araç bulunmaktadır (örnekler için bkz. www.pairwise.org). Söz konusu araçlar, parametrelerin ve değerlerinin listelenmesini (ikili test ve dikey dizi testi) ya da grafik formatında (sınıflandırma ağacı) gösterilmesini gerektirir [Grochtmann94]. İkili test, kombinasyonda yer alan değişken çiftlerini test etmek için kullanılır. Dikey diziler önceden tanımlanmıştır ve test edildikleri zaman kapsam seviyesine ulaşacak kombinasyon kümelerini oluşturmak suretiyle test analistinin dizideki değişkenlere yönelik test edilebilecek öğeleri çıkarmasını sağlayan matematiksel doğruluk taşıyan tablolardır [Koomen06]. Sınıflandırma ağacı araçları, test analistinin test edilecek kombinasyonların boyutunu tanımlamasını sağlar (diğer bir deyişle, iki değerli, üç değerli, vb kombinasyonlar).

Uygulanabilirlik

Çok sayıda kombinasyonun bulunmasıyla ilgili problem, teste ilişkin en az iki farklı durumda kendisini gösterir. Bazı test senaryoları, çok sayıda girdi alanı içeren örneklerde olduğu gibi çok sayıda olası değeri içeren çok sayıda parametre içermektedir. Bu durumda parametre değerlerinden oluşan kombinasyon, test senaryolarına yönelik girdi verilerini oluşturur. Buna ek olarak bazı sistemler, birden fazla boyutta yapılandırılabilir ve bu durum, potansiyel olarak büyük yapılandırma alanlarının oluşmasına yol açabilir. Her iki durumda da kombinasyonlu test, boyutları açısından esnek olan bir kombinasyonlar alt kümesi tanımlamak üzere kullanılabilir.

Çok sayıda değer içeren parametrelerde denklik sınıfının paylara ayrılması veya bazı diğer yöntemler, elde edilen kombinasyonların sayısını düşürmek için kombinasyonlu testin uygulanmasından önce, ilk olarak parametrelerin her birine yönelik değerlerin sayısını düşürmek için parametrelere teker teker uygulanabilir.

Söz konusu teknikler, genellikle entegrasyon, sistem ve sistem entegrasyon testi seviyelerinde uygulanır.

Sınırlamalar/Zorluklar

Bu tekniklerdeki en büyük kısıt, birkaç testin sonuçlarının tüm testleri temsil ettiği ve söz konusu birkaç testin beklenen kullanımı temsil ettiği varsayımıdır. Belirli değişkenler arasında beklenmedik bir etkileşim olması durumunda ilgili kombinasyonun test edilmemesi halinde söz konusu etkileşim tespit edilemeyecektir. Söz konusu tekniklerin, teknik bilgi birikimine sahip olmayan kitlelere açıklanması oldukça zordur zira bahsi geçen kişiler, testlerin mantığa dayalı indirgenmesini anlayamayabilir.

Parametrelerin ve ilgili değerlerinin tanımlanması kimi zaman zor bir işlemdir. Belirli bir kapsam seviyesinin elde edilmesi için minimum kombinasyon kümesinin bulunması da manuel olarak oldukça güçtür. Minimum kombinasyon kümelerini bulmak için genellikle araçlar kullanılır. Bazı araçlar, kimi kombinasyonların nihai kombinasyon seçimine eklenmesi ya da seçimden

çıkarılması özelliklerini destekleyebilir. Bu özellik, alan bilgisine ya da ürün kullanım bilgisine dayanarak etmenleri vurgulamak y da etmenleri üzerindeki vurguyu kaldırmak için test analisti tarafından kullanılabilir.

Kapsam

Çeşitli kapsam seviyeleri bulunmaktadır. En düşük kapsam, 1-yönlü ya da tekil kapsamdır. Seçilen kombinasyonların en azından birindeki her parametrenin değerinin bulunması gerekir. Bir sonraki kapsam seviyesi, 2-yönlü ya da ikili kapsamdır. En az bir kombinasyondaki iki parametrenin ikili değerinin bulunması gerekir. Test n-yönlü kapsam şeklinde genişletilebilir. Burada, seçilen kombinasyon kümelerine n parametreden oluşan kümelerin alt kombinasyon değerlerinin girilmesi gerekir. N değeri ne kadar yüksekse %100 kapsama ulaşmak için o kadar fazla kombinasyona ihtiyaç duyulur. Söz konusu tekniklerle sağlanan minimum kapsam, araç tarafından oluşturulan her kombinasyona karşılık bir test senaryosunun olması şeklindedir.

Hata Türleri

Bu tür bir testle tespit edilen en yaygın hatalar, çok sayıda parametrenin değerlerinin bir araya getirilmesi sırasında ortaya çıkan hatalardır.

3.2.7 Kullanım Senaryosu Testi

Kullanım senaryosu testi, sistem kullanımını taklit eden işlem ve senaryo bazlı testler sunar. Kullanım senaryoları, sistemle aktörler arasındaki etkileşimler baz alınarak tanımlanır. Aktörler, kullanıcı ya da harici sistem olabilir.

Uygulanabilirlik

Kullanım senaryosu testi, sistem ve kullanıcı kabul testi seviyelerinde uygulanır. Entegrasyonun seviyesine bağlı olarak entegrasyon testinde ve hatta bileşenin davranışına bağlı olarak birim testinde de kullanılabilir. Kullanım senaryoları, sistemin kullanımına dair gerçekçi bir portre çizdiklerinden performans testinin temelini teşkil eder. Kullanım senaryolarında açıklanan senaryolar, sistem üzerinde gerçekçi bir yük oluşturmak için sanal kullanıcılara atanabilir.

Sınırlamalar/Zorluklar

Geçerli olmaları için kullanım senaryolarının gerçekçi kullanıcı işlemlerini yansıtması gerekir. Bu bilgiler, bir kullanıcıdan ya da kullanıcı temsilcisinden elde edilmelidir. Kullanım senaryosunun gerçek bir kullanıcının faaliyetlerini doğru bir şekilde yansıtmaması durumunda kullanım senaryosunun değeri düşer. Çok sayıda alternatif yolun (akışın) doğru bir şekilde tanımlanması, test kapsamının detaylı olması açısından önem taşımaktadır. Kullanım senaryoları, birer kılavuz olarak değerlendirilmelidir. Gereksinim kümesinin tümünü açık bir şekilde tanımlayamayabileceklerinden test edilmesi gereken unsurlara dair bütüncül bir açıklama olarak görülmemelidir. Kullanım senaryosunu onaylamak ve testin doğruluğunu iyileştirmek için kullanım senaryosu anlatılarından akış şemaları gibi diğer modellerin oluşturulması da fayda sağlar.

Kapsam

Bir kullanım senaryosunun minimum kapsamı ana (pozitif) yol için bir test senaryosu ve alternatif yolların her biri için bir test senaryosu hazırlamaktan oluşmaktadır. Alternatif yollar, istisnai ve hatalı yolları kapsar. Alternatif yollar, kimi zaman ana yolun uzantılarıdır. Kapsam yüzdesi, test edilen yolların sayısı alınıp ana ve alternatif yolların toplam sayısına bölünerek elde edilir.

Hata Türleri

Hatalar, tanımlanan senaryoların yanlış bir şekilde ele alınmasını, alternatif yolların hazırlanmamasını, mevcut koşulların yanlış şekilde işlenmesini ve hataların yanlış bir şekilde raporlanmasını kapsar.

3.2.8 Kullanıcı Hikayesi Testi

Scrum gibi bazı çevik yazılım geliştirme yöntemlerinde gereksinimler küçük fonksiyonel birimleri temsil eden kullanıcı hikayeleri şeklinde hazırlanır. Bu hikayeler tek bir döngüde tasarlanabilen, geliştirilebilen ve test edilebilen büyüklüktedir. [Cohn04]. Söz konusu kullanıcı hikayeleri, uyarlanacak fonksiyonalitenin açıklanması, fonksiyonel olmayan kriterleri ve kullanıcı hikayesinin tamamlanmış sayılabilmesi için yerine getirilmesi gereken kabul kriterlerini içerir.

Uygulanabilirlik

Kullanıcı hikayeleri, esasen çevik ve benzeri döngüsel ve artımlı metodolojilerde kullanılır. Bunlar, hem fonksiyonel hem de fonksiyonel olmayan testlerde kullanılır. Kullanıcı hikayeleri tüm test seviyelerinde kullanılabilir.

Sınırlamalar/Zorluklar

Kullanıcı hikayeleri, fonksiyonalitenin küçük artımları olduğundan sunulan fonksiyonalite parçasını gerçekten test edebilmek için gerekli sürücüleri ve taklit uygulamaları oluşturmak gerekebilir. Bu, genellikle programlama ve API test araçları gibi teste yardımcı olacak araçları kullanma becerilerini gerektirir. Sürücülerin ve taklit uygulamaların hazırlanması, genellikle yazılımcının sorumluluğudur ancak teknik test analisti, söz konusu kodun hazırlanması ve API test araçlarının kullanılması süreçlerine katılabilir. Çoğu çevik yazılım geliştirme projesinde olduğu gibi kesintisiz entegrasyon modeli kullanılıyorsa sürücü ve taklit uygulamalara yönelik ihtiyaç daha düşüktür.

Kapsam

Bir kullanıcı hikayesinin minimum kapsamı, belirtilen kabul kriterlerinden her birinin karşılandığını onaylamayı gerektirir.

Hata Türleri

Bu test tekniği kullanılarak bulunan hatalar genellikle fonksiyonel hatalardır. Mevcut fonksiyonaliteye ilişkin yeni bir hikaye hazırlandığında bunun entegrasyonu sırasında da hatalar bulunabilmektedir. Hikayeler bağımsız olarak geliştirilebileceğinden performans, arayüz ve hataların ele alınması ile ilgili sorunlar görülebilir. Test analistinin, sunulan bağımsız fonksiyonaliteyi test etmesinin yanı sıra yeni hikayenin teste hazırlanmasının hemen ardından entegrasyon testini yürütmesi de çok önemlidir.

3.2.9 Alan Analizi

Alan, bir değerler kümesi olarak tanımlanır. Küme, tek bir değişkene bağlı değerler aralığı olabileceği gibi (tek boyutlu alan, örn "24 ile 66 yaşları arasındaki erkekler") etkileşim içindeki değişkenlerin değer aralıkları da (çok boyutlu alan, örn. "24 ile 66 yaşları arasındaki VE 69 kg ile 90 kg arasındaki erkekler") olabilir. Çok boyutlu alanlara yönelik test senaryolarının her biri, kapsam dahilinde bulunan değişkenlerin her birine yönelik uygun değerleri içermelidir.

Tek boyutlu bir alanın alan analizinde tipik olarak denklik paylarına ayırma ve sınır değer analizi kullanılır. Paylar tanımlandıktan sonra test analisti, payın içinden (IN), dışından (OUT), payın sınırından (ON) ve payın sınırlarının hemen dışından (OFF) bir değeri temsil edecek, her paydan çeşitli değerler seçer. Söz konusu değerler belirlendiğinde her pay, sınır koşullarıyla birlikte test edilir. [Black07]

Çok boyutlu alanlarda söz konusu yöntemler kullanılarak oluşturulan test senaryolarının sayısı, dahil olan değişken sayısıyla doğru orantılı bir şekilde artarken alan teorisine dayanan bir yaklaşım, yatayda büyüme gösterir. Buna ek olarak, genel yaklaşımda, denklik paylarına ayırma ve sınır değer analizinin bulunmadığı bir hata teorisi (arıza modeli) kullanıldığında daha büyük, sezgisel test kümelerinin kaçıracağı hataları, daha küçük test kümeleri çok boyutlu alanlarda kolaylıkla bulabilmektedir. Çok boyutlu alanlarla işlem yapılırken test modeli, karar tablosu (ya da alan matrisi) şeklinde oluşturulabilir. Üçün üzerindeki çok boyutlu alanlarda test senaryosu değerlerinin tanımlanması, bir araç desteği gerektirecektir.

Uygulanabilirlik

Alan analizi, karar tabloları, denklik paylarına ayırma ve sınır değer analizine yönelik kullanılan teknikleri bir araya getirerek önemli alanları ve olası arıza alanlarını kapsayacak küçük test kümeleri oluşturmaktadır. Potansiyel olarak etkileşim içine girebilecek değişkenlerin sayısı yüksek olduğundan karar tablolarının kullanılamayacağı yerlerde kullanılır. Alan analizi, testin herhangi bir seviyesinde gerçekleştirilebilir ancak en sık uygulandığı seviyeler, entegrasyon ve sistem testi seviyeleridir.

Sınırlamalar/Zorluklar

Ayrıntılı bir alan analizinin yapılması için, alanlar arasındaki olası etkileşimlerle çok sayıdaki alanı tanımlamak üzere yazılımın iyi bir şekilde anlaşılması gerekir. Alanın tanımlanmaması durumunda testte belirgin eksiklikler olacaktır. Alan analizi, test alanlarının tanımlanması konusunda bir yazılımcıyla çalışılırken kullanılabilecek güçlü bir tekniktir.

Kapsam

Alan analizine yönelik minimum kapsam, alanların her birindeki IN, OUT, ON ve OFF değerlerinin her birine yönelik bir test yapılmasını gerektirir. Değerler arasında herhangi bir çakışma olması durumunda (örneğin, bir alanın OUT değerinin diğer alanın IN değeriyle aynı olması halinde) testleri tekrarlamaya gerek yoktur. Bu nedenle gerekli olan gerçek testler, alan başına dört taneden daha azdır.

Hata Türleri

Hatalar, alan, sınır değeri kullanımı, değişken etkileşim problemlerini ve hataların ele alınması sırasında karşılaşılan fonksiyonel problemleri içerir (özellikle, geçerli bir alanda bulunmayan değerler için).

3.2.10 Tekniklerin Kombinasyonu

Kimi zaman, test senaryolarının oluşturulması için teknikler bir araya getirilir. Örneğin, bir karar tablosunda tanımlanan koşullar, koşullardan birinin sağlanabileceği çok sayıda yöntemi tespit etmek için denklik paylarına ayırmaya tabi tutulabilir. Bunun ardından test senaryoları, sadece koşul kombinasyonlarını içermez, aynı zamanda, paylara ayrılanlara ilişkin kombinasyonları da içerir. Denklik paylarını kapsam dahiline almak için ek test senaryoları oluşturulacaktır. Uygulanacak özel teknik seçilirken test analisti, testin uygulanabilirliğini, sınırlamalarını ve zorluklarını ve tespit edilecek hatalarla kapsam bağlamında testin hedeflerini göz önünde bulundurmalıdır. Bir duruma yönelik tek bir "en iyi" teknik bulunmayabilir. Bir araya getirilen teknikler, söz konusu teknikleri uygulamak için yeterli zaman ve beceri bulunduğu varsayıldığında en geniş kapsamı sunacaktır.

3.3 Hata Bazlı Teknikler

3.3.1 Hata Bazlı Tekniklerin Kullanılması

Hata bazlı test tasarım tekniği, aranan hata türünün test tasarımının esasını teşkil ettiği ve testlerin, hata türü hakkında bilinenlerden sistematik olarak elde edildiği bir tekniktir. Testlerini, gereksinimlere dayanarak hazırlayan spesifikasyon bazlı testin aksine hata bazlı testler, test edilmekte olan yazılımdan tamamen bağımsız olabilecek hata sınıflandırmalarını (diğer bir deyişle, kategori listelerini) kullanarak hazırlanır. Sınıflandırmalar, hata türlerini, kök nedenleri, arıza belirtilerini ve diğer hatayla ilgili verileri içerebilir. Hata bazlı test, testin hedeflerine temel teşkil etmesi için tanımlanmış risk listelerini ve risk senaryolarını da kullanabilir. Bu test tekniği, test uzmanının belirli bir hata türünü hedeflemesine veya belirli bir türdeki yaygın ve bilinen hataların hata sınıflandırması vasıtasıyla sistematik olarak ele almasına olanak tanır. Test analisti, belirli bir türde hatayı tespit etmek şeklinde açıklanabilecek testin amacını belirlemek üzere sınıflandırma verilerinden faydalanır. Test analisti, söz konusu bilgileri kullanarak mevcut olması halinde hatanın kendisini göstermesine neden olacak test senaryolarını ve test koşullarını oluşturur.

Uygulanabilirlik

Hata bazlı test, herhangi bir test seviyesinde uygulanabilir ancak genellikle sistem testi sırasında uygulanır. Farklı yazılım türleri için farklı hata sınıflandırma standartları bulunmaktadır. Yazılım yerine hataya odaklı olan bu test türü farklı hataların yakalanmasına yardımcı olur. Sektöre özel sınıflandırmalar kullanılarak hataların oluşumuna yönelik metrikler projelerde ve kurumlarda takip edilebilir.

Sınırlamalar/Zorluklar

Çok sayıda hata sınıflandırması bulunmakta ve bu sınıflandırmalar kullanılabilirlik gibi farklı test türlerine odaklanabilmektedir. Mevcut olması durumunda, test edilmekte olan yazılıma uygun bir sınıflandırmanın seçilmesi çok önemlidir. Örneğin, yenilikçi bir yazılım için kullanılabilecek herhangi bir sınıflandırma olmayabilir. Bazı kurumlar, sıklıkla görülen ya da görülme ihtimali olan hatalara ilişkin kendi sınıflandırmalarını derlemiştir. Hata sınıflandırması kullanıldığı zaman teste başlamadan önce beklenen kapsamın tanımlanması önemlidir.

Kapsam

Kullanılan hata bazlı test tekniğinde ele alınan hata kategorileri test kapsam yüzdesini belirler. Pratikte, hata bazlı tekniklere yönelik kapsam kriteri, spesifikasyon bazlı tekniklerinkinden daha az sistematiktir. Bunun nedeni, kapsama dair sadece genel kuralların verilmesi ve kullanışlı kapsamın sınırlarını teşkil eden özel kararların gizli olmasıdır. Diğer tekniklerde olduğu gibi kapsam kriteri, tüm testlerin bittiği anlamına gelmemektedir. Bunun yerine, test tekniğinde ele alınan hata kategorilerinin, herhangi bir yeni test çeşidine olanak vermediği anlamına gelir.

Hata Türleri

Tespit edilen hata türleri genellikle kullanılmakta olan sınıflandırmaya dayanmaktadır. Kullanıcı arayüzü sınıflandırması kullanılıyorsa tespit edilen hataların çoğunluğu, kullanıcı arayüzüyle ilgili olacaktır ancak testin yan ürünlerinden biri olarak farklı hatalar da tespit edilebilir.

3.3.2 Hata Sınıflandırmaları

Hata sınıflandırmaları, hata türlerinin kategorilere ayrılmış listesidir. Söz konusu listeler oldukça genel olup üst seviye kılavuz olarak kullanılabileceği gibi oldukça spesifik de olabilirler. Örneğin, kullanıcı arayüzü hatalarına yönelik sınıflandırmalar; fonksiyonalite, grafik ve performans gibi genel öğeleri içerebilir. Ayrıntılı bir sınıflandırma, tüm olası kullanıcı arayüzü nesnelerini içeren bir listeyi kapsayabilir (özellikle grafiksel kullanıcı arayüzü için) ve söz konusu nesnelerin aşağıdaki gibi yanlış kullanımlarını belirleyebilir:

- Metin alanı
    - Geçerli bilgiler kabul edilmiyor
    - Geçersiz bilgiler kabul ediliyor
    - Girdinin uzunluğu kontrol edilmemiş
    - Özel karakterler dikkate alınmamış
    - Hata mesajları yeterince bilgilendirici değil
    - Kullanıcı hatalı davranışını düzeltemiyor
    - Kurallara uyulmuyor
- Tarih alanı
    - Geçerli tarihler kabul edilmiyor
    - Geçersiz tarihler reddedilmiyor
    - Tarih aralıkları kontrol edilmiyor
    - Hassas bilgiler doğru şekilde ele alınmıyor (örn. ss:dd:ss)
    - Kullanıcı hata hakkında net bir şekilde bilgilendirilmiyor
    - İş ve mantık kurallarına uyulmuyor (örn. bitiş tarihi, başlangıç tarihinden daha sonraki bir tarih olmalıdır.)

Lisansı alınabilecek hata sınıflandırmalarından tutun da çeşitli yazılımlar için farklı amaçlar doğrultusunda tasarlananlara kadar çok sayıda hata sınıflandırması bulunmaktadır. Şirket içinde geliştirilen hata sınıflandırmaları özel hataları bulmak için de kullanılabilir. Yeni bir hata sınıflandırması oluştururken ya da mevcut olanı özelleştirirken ilk olarak sınıflandırmanın hedeflerini tanımlamak büyük önem taşır. Örneğin hedef, yazılımdaki kullanıcı arayüzü sorunlarını tespit etmek olabileceği gibi, girdi alanlarının kullanımıyla ilgili sorunların belirlenmesi de olabilir. Bir sınıflandırma oluşturmak için:

1. Bir hedef belirleyin ve istenilen seviyede detay tanımlayın
2. Esas olarak kullanılmak üzere belirli bir sınıflandırmayı seçin
3. Şirkette ve/veya uygulama sırasında tecrübe edilen yaygın hataları tanımlayın

Sınıflandırma ne kadar ayrıntılı olursa standardı geliştirmek ve sürdürmek için gerekli süre de o kadar artacaktır. Fakat bu beraberinde testlerde daha yüksek bir tekrarlanabilirliği de getirecektir. Ayrıntılı hata sınıflandırmaları gereğinden fazla bilgi içerebilir ancak test ekibinin, herhangi bir bilgi ya da kapsam kaybı olmaksızın testi yürütmesini mümkün kılmaktadır. Uygun hata sınıflandırması seçildikten sonra söz konusu sınıflandırma, test koşullarının ve test senaryolarının oluşturulması için kullanılabilir. Risk bazlı sınıflandırma, testin belirli bir risk alanına odaklanmasına yardımcı olabilir. Sınıflandırmalar, aynı zamanda, kullanılabilirlik, performans, vb gibi fonksiyonel olmayan alanlarda da kullanılabilir. Sınıflandırma listeleri, IEEE tarafından yayınlanan çoğu yayında ve internet üzerinde bulunmaktadır.

3.4 Tecrübeye Dayalı Teknikler

Tecrübeye dayalı testler, test uzmanlarının benzer uygulamalar veya teknolojiler konusundaki deneyimlerinin yanı sıra becerilerinden ve sezgilerinden de faydalanır. Söz konusu testler, hataların bulunması açısından etkindir ancak belirli test kapsamı seviyelerinin elde edilmesi ya da yeniden kullanılabilir test prosedürlerinin oluşturulması gibi konularda kullanımı diğer teknikler kadar uygun değildir. Dokümantasyonunun yetersiz olduğu veya test ekibinin test edilecek sistem konusunda iyi bir deneyime sahip olduğu durumlarda tecrübeye dayalı testler, daha sistematik yaklaşımlara göre iyi bir alternatif olabilir. Tecrübeye dayalı test, ayrıntılı test dokümantasyonu, yüksek tekrar edilebilirlik seviyesi veya test kapsamının hassas bir şekilde değerlendirilmesini gerektiren sistemlerde kullanıma pek uygun değildir.

Dinamik ve sezgisel değerlendirmeler kullanılırken test uzmanları normal olarak tecrübeye dayalı testleri kullanmaktadır ve testler, önceden planlanan test yaklaşımlarıyla kıyaslandığında olaylara karşı daha tepkiseldir. Buna ek olarak yürütme ve değerlendirme eşzamanlı olarak gerçekleştirilmektedir. Tecrübeye dayalı testlerde bazı durumlarda sistematik yaklaşımlar uygulanabilir. Bu tür yaklaşımlarda testin tasarlanması ve yürütülmesi eş zamanlı olarak yapılmayabilir. Tecrübeye dayalı teknikler resmi kapsam kriterlerine sahip değildir.

3.4.1 Hata Tahminleme

Test analisti, hata tahminleme tekniğini kullanırken deneyimlerinden yararlanarak kodun tasarımı ve geliştirilmesi aşamasında ortaya çıkabilecek olası hataları tahmin eder. Beklenen hatalar tanımlandıktan sonra test analisti, bu hataları ortaya çıkarmak için kullanılabilecek en uygun yöntemleri belirler. Örneğin, test analisti, geçersiz bir parola girildiğinde yazılımda arıza yaşanmasını bekliyorsa testleri hataya yol açacak şekilde parola alanına çok sayıda farklı değeri girecek şekilde tasarlayabilir.

Bir test tekniği olarak kullanılmasının yanı sıra hata tahminleme, olası arıza durumlarını tanımlamak için risk analizi sırasında da kullanılabilir. [Myers79]

Uygulanabilirlik

Hata tahminleme, esasen entegrasyon ve sistem testi sırasında gerçekleştirilir ancak testin herhangi bir seviyesinde kullanılabilir. Bu teknik, sıklıkla diğer tekniklerle birlikte kullanılır ve mevcut test senaryolarının kapsamını genişletmeye yardımcı olur. Hata tahminleme, daha yoğun ve sistematik bir test yürütülmeden önce yaygın hataları ve yanlışları bulmak amacıyla etkin bir şekilde kullanılabilir. Kontrol listeleri ve hata sınıflandırması, bu tür testlere kılavuzluk etmesi açısından faydalı olabilir.

Sınırlamalar/Zorluklar

Kapsamın değerlendirilmesi oldukça zordur. Test analistinin becerileri ve deneyimine bağlı olarak büyük ölçüde değişiklik gösterir. Bu teknik ele alınan hata türlerine aşina, deneyimli bir test uzmanı tarafından en iyi şekilde kullanılacaktır. Hata tahminleme yaygın bir şekilde
kullanılmaktadır ancak çok fazla dokümante edilememektedir; dolayısıyla, diğer test teknikleriyle kıyaslandığında tekrarlanabilirliği daha düşük olabilir.

Kapsam

Hata sınıflandırması tekniğinde kapsam, ele alınacak veri hataları ve hata türleri ele alınarak belirlenir. Belli bir hata sınıflandırma yaklaşımı kullanılmadığında kapsam, test uzmanının deneyimi ve bilgi birikimiyle ve mevcut zamanla sınırlıdır. Bu teknikten elde edilen verim, test uzmanının problemli alanları hedef alma becerisine göre değişiklik gösterecektir.

Hata Türleri

Tipik hatalar genellikle ilgili hata sınıflandırmasında belirtilenler ya da spesifikasyon bazlı test kapsamında yer almayıp test analisti tarafından öngörülenlerdir.

3.4.2 Kontrol Listesine Dayalı Test Etme

Kontrol listesi bazlı teknikleri uygulanırken test analisti, dikkat edilmesi, kontrol edilmesi ve hatırlanması gereken unsurlardan oluşan üst seviye genel bir listeyi ya da ürünle karşılaştırılarak onaylanması gereken bir dizi kural ya da kriteri kullanır. Söz konusu kontrol listeleri, standartlara, deneyime ve göz önünde bulundurulan diğer unsurlara dayanarak hazırlanır. Bir uygulamanın test esası olarak kullanılan kullanıcı arayüzü standartları kontrol listesi, kontrol listesine dayalı test etmeye örnek olarak verilebilir.

Uygulanabilirlik

Kontrol listesine dayalı test etme, test edilmekte olan yazılıma ya da kontrol listesi kapsamındaki alana aşina deneyimli bir test ekibi tarafından yürütülen projelerde verimli bir şekilde kullanılabilir (örn. bir kullanıcı arayüzü kontrol listesini başarılı bir şekilde uygulamak isteyen test analisti, kullanıcı arayüz testine aşina olabilir ancak test edilmekte olan yazılıma aşina olmayabilir). Kontrol listeleri üst seviyeli olduğundan ve yaygın olarak test senaryolarında ve test prosedürlerinde bulunan ayrıntılı aşamaları barındırmadıklarından test uzmanının bilgi birikimi, eksiklikleri doldurmak için kullanılır. Ayrıntılı aşamalar ortadan kaldırıldığında kontrol listeleri daha az bakım gerektirir. Kontrol listeleri, farklı test seviyelerine yönelik kullanılabilir. Kontrol listeleri, regresyon testleri ve duman testleri için de kullanılır.

Sınırlamalar/Zorluklar

Kontrol listelerinin üst seviye yapısı, test sonuçlarının tekrarlanabilirliğini etkileyebilir. Birden fazla test uzmanının kontrol listelerini farklı şekillerde yorumlaması ve kontrol listesinde yer alan öğeleri yerine getirmek üzere farklı yaklaşımlar izlemesi olasıdır. Bu durum ise her ne kadar aynı kontrol listesi kullanılsa da farklı sonuçların ortaya çıkmasına yol açabilir. Bu, kapsama alanının genişlemesini sağlayabilir ancak kimi zaman tekrarlanabilirlik özelliğinden feragat edilmesi gerekmektedir. Kontrol listeleri, elde edilen kapsam seviyesi konusunda gereğinden fazla güvene neden olabilir çünkü gerçek test, test uzmanının kanaatine bağlıdır. Kontrol listeleri, daha detaylı test senaryolarından ya da listelerden elde edilebilir ve zaman içinde büyüme

gösterebilir. Kontrol listelerinin, test edilmekte olan yazılımın önemli alanlarını kapsadığından emin olmak için kontrol listeleri sürekli güncellenmelidir.

Kapsam

Kontrol listelerinin üst seviye yapıları nedeniyle sonuçlar, kontrol listesini dolduran test analistine bağlı olarak değişiklik gösterecektir.

Hata Türleri

Bu teknikle bulunan tipik hatalar, değişen bilgilerden, aşamaların sıralanışından ya da test sırasındaki genel iş akışından kaynaklanan arızaları içermektedir. Kontrol listelerinin kullanılması, test sırasında yeni veri ve süreç kombinasyonlarının dahil edilmesine izin verildiğinden testin daima güncel kalmasına yardımcı olur.

3.4.3 Keşif Testi

Keşif testi, yazılım test edildikçe hataların bulunup yazılım hakkında daha fazla bilgiye ulaşıldığı ve bu bilgiler ışığında yeni testlerin planlandığı ve tasarlandığı sürekli öğrenmeyi gerektiren bir yaklaşımdır.

Test uzmanı, testin yürütülmesi sırasında testin hedeflerini dinamik olarak ayarlar ve sadece basit dokümanlar hazırlar. [Whittaker09]

Uygulanabilirlik

İyi bir keşif testi, planlıdır, interaktiftir ve yaratıcıdır. Test edilecek sistem konusunda çok az dokümana gerek duyar ve dokümantasyonun bulunmadığı veya diğer test teknikleri için yeterli olmadığı durumlarda sıklıkla kullanılır. Keşif testi, diğer testleri güçlendirmek ve ek test senaryolarının geliştirilmesine temel teşkil etmek için kullanılır.

Sınırlamalar/Zorluklar

Keşif testinin yönetilmesi ve hayata geçirilmesi kolay olmayabilir. Kapsam dağınık ve tekrarlama zorlu olabilir. Test oturumunda ele alınacak alanları belirlemek için başlatma belgelerinin ve test için ayrılmış zamanı belirlemek üzere zaman kutucuklarının kullanılması, keşif testlerini yönetmek için takip edilebilecek yöntemlerden biridir. Test oturumunun ya da oturumların sonunda test yöneticisi, testin sonuçlarını toplamak ve yeni oturumlara yönelik başlatma belgelerini belirlemek için bir bilgilendirme oturumu düzenleyebilir. Bilgilendirme oturumlarının, büyük test ekipleri veya büyük projeler için ölçeklendirilmesi zordur.

Keşif oturumlarıyla ilgili bir diğer zorluk, bunları test yönetim sisteminde doğru bir şekilde takip edebilmekle ilgilidir. Bu kimi zaman gerçekten keşif oturumuna özel test senaryolarının oluşturulmasıyla çözülebilir. Bu, keşif testine yönelik zaman tahsis edilmesini ve planlanan kapsamın diğer test çalışmalarıyla birlikte takip edilmesini mümkün kılar.

Keşif testlerinde hatanın tekrarlanabilirliği zorlukla sağlanabileceğinden bir arızanın tekrarlanması sırasında atılan adımları yeniden oluşturmak gerektiğinde bu durum problem yaratabilir. Bazı kurumlar, keşif testi uzmanı tarafından atılan adımları kayıt altına almak için test otomasyonu aracının yakala/tekrar oynat özelliğinden faydalanmaktadır. Bu, keşif oturumu (ya da deneyim bazlı test oturumu) sırasında atılan tüm adımların kayıt altına alınmasını sağlar. Arızanın gerçek nedeninin bulunması için ayrıntıların deşilmesi yorucu olabilir ancak en azından atılan tüm adımlara dair bir kayıt bulunmaktadır.

Kapsam

Bu teknikte kullanılan başlatma belgeleri, görevleri, hedefleri ve çıktıları kapsar. Bunun ardından söz konusu hedeflere ulaşmak için keşif oturumları planlanır. Başlatma belgesi, test çalışmalarının nereye odaklanması gerektiğini, test oturumunun kapsamını ve planlanan testleri tamamlamak için hangi kaynaklarının kullanılması gerektiğini de tanımlayabilir. Oturum, özel hata türlerine ve betiğe dayalı testin formaliteleri olmaksızın ele alınabilecek diğer olası problemli alanlara odaklanmak için kullanılabilir.

Hata Türleri

Keşif testiyle tespit edilen tipik hatalar, sistematik kurgulanmış fonksiyonel testler sırasında gözden kaçan senaryo bazlı sorunlar, fonksiyonel sınırlar arasında kalan sorunlar ya da iş akışıyla ilgili sorunlardır. Performans ve güvenlikle ilgili sorunlar, kimi zaman keşif testi sırasında ortaya çıkar.

3.4.4 En İyi Tekniğin Uygulanması

Hata ve deneyim bazlı tekniklerde, test uzmanı hata bulma oranını artırmak için bilgi birikimini daha fazla ön plana çıkarmalıdır. Bu tür testler, test uzmanının resmi olarak herhangi bir planlamada bulunmadığı "hızlı testlerden" başlayıp önceden planlanan ve betiğe dayalı oturumları baz alan daha sistematik test yaklaşımlarına kadar büyük değişiklik gösterir. Her zaman kullanışlıdırlar ancak aşağıdaki durumlarda büyük önem taşırlar:

- Herhangi bir gereksinim bulunmadığı durumlar
- Test edilmekte olan sistem hakkında yeterli dokümantasyon bulunmadığında
- Ayrıntılı testleri tasarlamak ve oluşturmak için yeterli zaman ayrılmadığında
- Test uzmanları alan ve/veya teknoloji konusunda deneyim sahibi olduğunda
- Betiğe dayalı testte çeşitliliğin test kapsamını genişletmek için bir hedef olduğu hallerde
- Operasyonel arızalar analiz edileceğinde

Hata ve deneyim bazlı teknikler, spesifikasyon bazlı tekniklerle birlikte kullanıldıklarında da oldukça faydalıdır zira söz konusu tekniklerdeki sistematik zayıflıklardan kaynaklanan test kapsamındaki eksiklikleri doldururlar. Spesifikasyon bazlı tekniklerde olduğu gibi tüm durumlara uygun kusursuz bir teknik bulunmamaktadır. Test analistinin, tekniklerin her birinin avantajlarını ve dezavantajlarını anlaması, ve proje türünü, programını, bilgiye erişimi, test uzmanının becerilerini ve seçimi etkileyebilecek diğer etmenleri göz önünde bulundurarak duruma uygun en iyi tekniği ya da teknikler kümesini seçebilmesi çok önemlidir.

4. Yazılım Kalite Karakteristiğini Test Etme

120 dak.

Anahtar Sözcükler

erişilebilirlik testi, doğruluk testi, çekicilik, sezgisel değerlendirme, birlikte çalışabilirlik testi, öğrenilebilirlik, işletilebilirlik, kullanışlılık testi, SUMI, anlaşılabilirlilik, kullanılabilirlik testi, WAMMI

Öğrenme Hedefleri

4.2 İş Alanı Testine Yönelik Kalite Karakteristikleri

TA-4.2.1 (K2) Doğruluğu, kullanışlılığı, birlikte çalışabilirliği ve uyumluluk karakteristiklerini uygun bir şekilde test etmek için kullanılabilecek test tekniklerini örnekleriyle açıklayın

TA-4.2.2 (K2) Doğruluk, kullanışlılık ve birlikte çalışabilirlik karakteristikleri için hedef alınacak tipik hataları tanımlayın

TA-4.2.3 (K2) Doğruluk, kullanışlılık ve birlikte çalışabilirlik gibi karakteristiklerin her birinin yaşam döngüsü içinde ne zaman test edilmesi gerektiğini tanımlayın.

TA-4.2.4 (K4) Bir proje içinde kullanılabilirlik gereksinimlerinin uyarlandığını ve kullanıcının beklentilerinin karşılandığını doğrulamak ve sağlamak için uygun olabilecek yaklaşımları genel hatlarıyla açıklayın.

4.1 Giriş

Bir önceki bölümde test uzmanının kullanabileceği özel teknikler tanımlanırken, bu bölümde söz konusu tekniklerin yazılımın kalitesini tanımlamak üzere kullanılan başlıca karakteristiklerin değerlendirilmesi sırasındaki kullanımı ele alınmaktadır.

Bu ders programı, test analisti tarafından değerlendirilebilecek kalite karakteristiklerini ele almaktadır. Teknik test analisti tarafından değerlendirilecek özellikler, ileri seviye teknik test analisti ders programında işlenmiştir. ISO 9126'da sunulan ürün kalite karakteristiği açıklaması, karakteristiklerin tanımlanması sırasında bir kılavuz görevi görmüştür. ISO 25000 [ISO25000] serisi gibi diğer standartlar (ISO 9126'nın yerine geçen) da kullanılabilir. ISO kalite karakteristikleri, her birinin alt karakteristikleri (alt özellikleri) olabilecek ürün kalite karakteristiklerine (özelliklerine) bölünmüştür. Hangi karakteristiğin/alt karakteristiğin test analisti ya da teknik test analisti ders programında ele alındığı aşağıdaki tabloda verilmiştir:

Karakteristik	Alt Karakteristik	Test Analisti	Teknik Test Analisti
Fonksiyonalite	Doğruluk, kullanışlılık, birlikte çalışabilirlik, uyumluluk	X	
	Güvenlik		X
Güvenirlik	Olgunluk (sağlamlık), hata toleransı, kurtarılabilirlik, uyumluluk		X
Kullanılabilirlik	Anlaşılabilirlik, öğrenilebilirlik, işletilebilirlik, çekicilik, uyumluluk	X	
Verimlilik	Performans (zamana bağlı davranış), kaynak kullanımı, uyumluluk		X
Sürdürülebilirlik	Çözümlenebilirlik, değiştirebilirlik, kararlılık, test edilebilirlik uyumluluk		X
Taşınabilirlik	Uyarlanabilirlik, kurulabilirlik, bir arada varolma, değiştirilebilirlik, uyumluluk		X

Test analisti, fonksiyonalite ve kullanılabilirlik yazılım kalite karakteristiklerine odaklanmalıdır. Erişilebilirlik testi, test analisti tarafından yürütülmelidir. Her ne kadar bir alt karakteristik olarak listelenmemiş olsa da erişilebilirlik, genellikle kullanılabilirlik testinin bir parçası olarak değerlendirilir. Diğer kalite karakteristiklerinin test edilmesi, genellikle teknik test analistinin yükümlülüğündedir. Her ne kadar farklı kurumlarda görevlerin tahsisi değişiklik gösterebilse de yukarıda listelenen görevlendirme ISTQB ders programlarında takip edilmektedir.

Uyumluluk alt karakteristiği, kalite karakteristiklerinin her biri için gösterilir. Emniyetin önem taşıdığı ya da düzenleme kapsamındaki ortamlarda kalite karakteristiklerinin her biri, özel standartlara ve yönetmeliklere uygun olmalıdır (örn. fonksiyonalite uyumluluğu fonksiyonalitenin, örneğin, bir çipten veri göndermek/veri almak için gerekli iletişim protokolünü kullanan özel standartlara uygun olduğunu göstermeli). Söz konusu standartlar sektöre bağlı olarak büyük ölçüde değişiklik gösterdiğinden burada derinlemesine ele alınmayacaklardır. Test analistinin uyumluluk gereksinimlerinden etkilenen bir ortamda çalışıyor olması halinde söz konusu gereksinimleri anlamak ve hem testin hem de test dokümantasyonunun uyumluluk gereksinimlerini yerine getireceğinden emin olmak çok önemlidir.

Bu bölümde ele alınan kalite karakteristiklerinin ve alt karakteristiklerin tümü göz önünde bulundurulduğunda uygun bir test stratejisinin oluşturulması için tipik riskler değerlendirilmelidir. Kalite karakteristiği testi, yaşam döngüsünün zamanlamasına, gerekli araçlara, yazılıma ve dokümantasyona ulaşılabilirliğe ve teknik uzmanlığa önem verilmesini gerektirir. Karakteristiklerin her biriyle ve benzersiz test ihtiyaçlarıyla ilgilenmek için bir strateji planlanmazsa test uzmanının, programda yeterli planlamaya, iyileştirmelere ve test yürütme süresine sahip olması mümkün olmayacaktır. Kullanılabilirlik testi gibi bazı testler, projeye özel katılımcıların tahsisini, kapsamlı planlamayı, özel laboratuvarların kurulmasını, özel araçların teminini, özel test becerilerini ve çoğu durumda oldukça uzun bir süreyi gerektirebilir. Bazı durumlarda kullanılabilirlik testi, ayrı kullanılabilirlik grubu uzmanları ya da kullanıcı deneyimi uzmanları tarafından gerçekleştirilebilir.

Kalite karakteristiği alanlarından her biri kendisine özel ihtiyaçlar barındırmakta, özel sorunları ele almakta ve aşağıdaki bölümlerde de tartışıldığı üzere yazılım geliştirme yaşam döngüsü sırasında farklı anlarda ortaya çıkabilmektedir.

Her ne kadar test analisti, daha teknik bir yaklaşım gerektiren kalite karakteristiklerinden sorumlu tutulamasa da test analistinin diğer karakteristiklerin bilincinde olması ve teste yönelik kesişen alanları anlaması önemlidir. Örneğin, performans testinde başarısız olan bir ürün, kullanıcının etkin bir şekilde kullanılması için yeterince hızlı değil ise kullanılabilirlik testinde de başarısız olacaktır. Benzer şekilde, bazı bileşenlerinde birlikte çalışabilirlik sorunu olan bir ürün taşınabilirlik testine hazır değildir zira ortam değiştiği zaman daha temel işlemlerde problemler yaşamaya başlayacaktır.

4.2 İş Alanı Testlerine Yönelik Kalite Karakteristikleri

Fonksiyonel testler, test analistinin başlıca odak noktasıdır. Fonksiyonel testler, ürünün "ne" yaptığına odaklanır. Fonksiyonel testlerin test esası, genellikle gereksinimler veya analiz dokümanı, özel alan uzmanlığı veya dolaylı ihtiyaçlardır. Fonksiyonel testler, yürütüldükleri test seviyesine göre değişiklik gösterebilir ve yazılım geliştirme yaşam döngüsünden etkilenebilir. Örneğin, entegrasyon testi sırasında yürütülen bir fonksiyonel test, etkileşim içindeki modüllerin fonksiyonalitesini test edecektir. Sistem test seviyesindeki fonksiyonel testler, uygulamanın fonksiyonalitesini bir bütün olarak test etmeyi gerektirir. Farklı sistemlerden oluşmuş sistemler fonksiyonel testler, öncelikle entegre edilen sistemler boyunca uçtan uca testlere odaklanacaktır. Çevik metodolojilerde, fonksiyonel testler genellikle belirli bir döngüdeki ya da sprintteki fonksiyonaliteyle sınırlıdır ancak bir döngüdeki regresyon testi, geride kalan tüm fonksiyonları içerebilir.

Fonksiyonel test sırasında geniş bir yelpazeye sahip test teknikleri uygulanmaktadır (Bakınız Bölüm 3). Fonksiyonel testler, özel olarak bu konuyla görevli bir test uzmanı, bir alan uzmanı ya da bir yazılımcı (genellikle birim seviyesinde) tarafından gerçekleştirilebilir.

Bu bölümde ele alınan fonksiyonel testlere ek olarak, test analistinin sorumluluk alanına ilişkin fonksiyonel olmayan (ürünün fonksiyonu "nasıl" sunduğuna odaklanır) test alanları olarak değerlendirilen iki kalite karakteristiği bulunmaktadır. Söz konusu iki özellik, kullanılabilirlik ve erişilebilirliktir.

Bu bölümde aşağıdaki kalite karakteristikleri ele alınmıştır:

- Fonksiyonel kalite alt karakteristikleri
    - Doğruluk
    - Kullanışlılık
    - Birlikte çalışabilirlik
- Fonksiyonel olmayan kalite karakteristikleri
    - Kullanılabilirlik
    - Erişilebilirlik

4.2.1 Doğruluk Testi

Fonksiyonel doğruluk, uygulamanın belirtilen ya da ima edilen gereksinimlere uyumunun test edilmesini gerektirir ve hesaba dayalı doğruluğu da kapsayabilir. Doğruluk testinde, Bölüm 3'te açıklanan test tekniklerinin çoğu kullanılır ve daha önce yapılan sistem testi sonuçları tahmin etme mekanizması olarak işlev görebilir. Doğruluk testi, yaşam döngüsünde istenilen aşamada yürütülebilir ve verilerin veya durumların yanlış bir şekilde ele alınmasına odaklanmaktadır.

4.2.2 Kullanışlılık Testi

Kullanışlılık testi, bir fonksiyon kümesinin hedeflenen görevlerine uygunluğunun değerlendirilmesini ve sağlanmasını içerir. Söz konusu test, kullanım senaryolarına dayanabilir. Kullanışlılık testi, sistem testi sırasında yürütülür ancak entegrasyon testinin sonraki aşamalarında gerçekleştirilmesi de mümkündür. Bu testte bulunan hatalar, ilgili sistemin kullanıcının ihtiyaçlarını kabul edilebilir seviye karşılayıp karşılayamayacağına dair göstergelerdir.

4.2.3 Birlikte Çalışabilirlik Testi

Birlikte çalışabilirlik testleri, iki ya da daha fazla sistemin veya bileşenin, bilgi alışverişi yapabilme ve daha sonrasında bu bilgiyi kullanabilme seviyelerini test eder. Test, veri alışverişinin sorunsuz bir şekilde gerçekleşmesi için hedeflenen tüm ortamları kapsamalıdır (donanımdaki, yazılımdaki, işletim sistemindeki, vb varyasyonlar dahil olmak üzere). Gerçekte, söz konusu durum sadece çok az sayıda ortam için uygun olabilir. Bu durumda birlikte çalışabilirlik testi, sadece seçilen bir ortam grubuyla sınırlı olabilir. Testlerin birlikte çalışabilirlik için belirlenmesi, hedef ortam kombinasyonlarının tanımlanmasını, yapılandırılmasını ve test ekibine sunulmasını gerektirir. Bunun ardından, söz konusu ortamlar, ortamda mevcut olan çok sayıda veri alışverişi noktasını kullanan fonksiyonel test senaryoları kullanılarak test edilir.

Birlikte çalışabilirlik, yazılım sistemlerinin birbirleriyle ne derece farklı etkileşimlere girdiğiyle ilgilidir. Birlikte çalışabilirlik özellikleri iyi olan bir yazılım, herhangi bir önemli değişiklik yapılmaksızın çok sayıda farklı sisteme entegre edilebilir. Değişiklik sayısı ve söz konusu değişiklikleri yapmak için gerekli çaba, birlikte çalışabilirlik ölçütü olarak kullanılabilir.

Yazılımın birlikte çalışabilirliğinin test edilmesi sırasında aşağıdaki gibi tasarım özelliklerine odaklanmak mümkündür:

- XML gibi sektör genelinde kullanılan iletişim standartları
- Etkileşime girdiği sistemin iletişim ihtiyaçlarını otomatik olarak tespit etme ve buna uygun düzenlemeler yapma becerisi

Birlikte çalışabilirlik testi, birden fazla sistemden oluşan sistemleri geliştirmekte olan kurumlar ve paket yazılım ve araç geliştirmekte olan kurumlar açısından özellikle ilgi çekici olabilir.

Bu test türü, bileşen entegrasyonu ve sistemin ortamla etkileşimine odaklanan sistem testi sırasında gerçekleştirilir. Sistem entegrasyon seviyesinde söz konusu test türü, tamamen geliştirilmiş bir sistemin diğer sistemlerle ne derece doğru etkileşime girdiğini belirlemek amacıyla yürütülür. Sistemler çok sayıda seviyede birlikte çalışabileceğinden test analisti, söz konusu etkileşimleri anlamalı ve çeşitli etkileşimlerin oluşturulabileceği koşulları sağlayabilmelidir. Örneğin, iki sistem arasında veri alışverişi olacaksa test analisti, veri takasını gerçekleştirmek için gerekli veri ve işlemleri oluşturabilmelidir. Tüm etkileşimlerin gereksinimler dokümanlarında açık bir şekilde ifade edilmeyebileceğinin unutulmaması gerekir. Bunun yerine, söz konusu etkileşimlerin çoğu sadece sistem mimarisi ve tasarım dokümanlarında tanımlanacaktır. Test analisti, tümünün test edildiğinden emin olmak amacıyla sistemler ve sistemlerle ortamlar arasındaki bilgi takası noktalarını belirlemek için söz konusu dokümanları incelemeye hazır olmalıdır. Karar tabloları, durum geçişi şemaları, kullanım senaryoları ve kombinasyonlu testler gibi teknikler, birlikte çalışabilirlik testlerinde kullanılabilir. Etkileşim içindeki bileşenler arasında yanlış veri alışverişi yapılması tipik olarak karşılaşılan hatalardandır.

4.2.4 Kullanılabilirlik Testi

Kullanıcıların sistemi kullanırken zorlanma nedenlerinin anlaşılması önemlidir. Bu bilgiyi elde etmek için "kullanıcı" teriminin, BT uzmanlarından çocuklara ve engelli kişilere kadar uzanan geniş bir yelpazeyi kapsadığının unutulmaması gerekir.

Bazı ulusal kurumlar (örn. Görme Engelliler İçin İngiltere Ulusal Enstitüsü), web sayfalarının engelli, görme engelli, kısmi görme engelli, hareket engelli, duyma engelli ve bilişsel olarak engelli kullanıcılar tarafından kullanılabilmesini önermektedir. Uygulamaların ve web sitelerinin yukarıda bahsi geçen kullanıcılar için uygun olduğunu kontrol etmek, kullanılabilirliği herkes için iyileştirecektir. Erişilebilirlik aşağıda detaylı bir şekilde ele alınmıştır.

Kullanılabilirlik testleri, kullanıcının belirli bir bağlamda belirli bir hedefe ulaşmak için sistemi kullanmayı öğrenme ya da kullanma kolaylığını test eder. Kullanılabilirlik testleri, aşağıdaki ölçmeyi hedeflemektedir:

- Etkinlik – Yazılımın belirtilen kullanım bağlamında doğru ve eksiksiz bir biçimde kullanıcının belirtilen hedeflere ulaşmasına yardımcı olma becerisidir.
- Verimlilik - Yazılımın belirtilen kullanım bağlamında elde edilen etkinliğe ilişkin olarak kullanıcının kaynakları uygun şekilde kullanmasına yardımcı olma becerisidir.
- Memnuniyet – Yazılımın belirtilen kullanım bağlamında kullanıcıları tatmin etme becerisidir.

Ölçülebilecek özelliklerden bazıları şunlardır:

- Anlaşılabilirlik – Yazılımın, mantıksal konsepti anlamak ve uygulanabilirliğini tespit etmek için kullanıcının harcaması gereken eforu etkileyen özelliktir.
- Öğrenilebilirlik – Yazılımın, kullanıcının uygulamayı kullanmayı öğrenmesi için harcaması gereken eforu etkileyen özelliktir.
- İşletilebilirlik – Yazılımın, kullanıcı tarafından etkin ve verimli bir şekilde kullanılabilmesi için harcaması gereken eforu etkileyen özelliktir.
- Çekicilik – Yazılımın kullanıcı tarafından beğenilme becerisidir.

Kullanılabilirlik testi genellikle iki aşamada gerçekleştirilir:

- Biçimlendirici Kullanılabilirlik Testi – Tasarımdaki kullanılabilirlik hatalarını tanımlayarak tasarımı (ya da biçime) yardımcı olmak için tasarım prototip aşamaları sırasında döngüsel bir şekilde yürütülen testlerdir.
- Özetleyici Kullanılabilirlik Testi – Tamamlanmış bir bileşen ya da sistemdeki kullanılabilirliği ölçmek ve problemleri tanımlamak için uyarlamadan sonra yürütülen testlerdir.

Kullanılabilirlik test uzmanları, aşağıdaki alanlarda deneyime veya bilgi birikime sahip olmalıdır:

- Sosyoloji
- Psikoloji
- Ulusal standartlara uyum (erişilebilirlik standartlara dahil olmak üzere)
- Ergonomi

4.2.4.1 Kullanılabilirlik Testlerinin Yürütülmesi

Kullanılabilirlik testinin sistemin kullanılacağı şartlara mümkün olduğunca yakın koşullar altında yapılması gerekir. Yazılım ekibinin, gerçek sistemin gerçek kullanıcılar üzerindeki etkisini gözlemleyebilmesi için video kameralarla donatılmış kullanılabilirlik laboratuvarları, kullanıcıların gerçek çalışma ortamlarına erişimi, gözden geçirme toplantılarına katılım gibi olanaklara sahip olması gerekebilir. Resmi kullanılabilirlik testlerinde kullanıcılara hazır test senaryoları ya da takip etmeleri gereken talimatlar verilerek "kullanıcıların" hazırlanması gerekmektedir (bunlar gerçek kullanıcı ya da kullanıcı temsilcileri olabilir). Diğer serbest formatta düzenlenen testler, kullanıcının yazılımı denemesini sağlar; bu şekilde, gözlemciler, kullanıcının görevleri ne kadar kolaylıkla ya da zorlukla yerine getirdiğini belirleyebilir.

Kullanılabilirlik testlerinin çoğu, fonksiyonel sistem testi gibi diğer testlerin bir parçası olarak test analisti tarafından yürütülebilir. Yaşam döngüsünün her aşamasındaki kullanılabilirlik hatalarının tespiti ve raporlanması konularına dair tutarlı bir yaklaşım elde etmek için kullanılabilirlik kılavuzlarından yararlanılabilir. Kullanılabilirlik kılavuzları "kabul edilemez" kullanılabilirlik kriterlerinin ne olduğunu belirlemek açısından önemlidir. Örneğin, kullanıcının bir uygulamada oturum açması için 10 tıklama yapması makul sayılabilir mi? Test analisti, söz konusu kılavuzlara sahip olmazsa yazılımcının kapatmak istediği hata raporlarını savunurken zor durumda kalabilir zira yazılım, "tasarlandığı şekilde" çalışmaktadır. Tüm benzer projelere uygulanan bir dizi kullanılabilirlik kılavuzuna ek olarak gereksinimlerde tanımlanan kullanılabilirlik gereksinimlerine sahip olmak çok önemlidir. Kılavuzlar, talimatların erişilebilirliği, yönlendirmelerin açıklığı, bir işlemi yerine getirmek için gerekli tıklama sayısı, hata mesajları, işlem göstergeleri (kullanıcıya, sistemin şu anda bir işlem yapmakta olduğunu ve daha fazla girdi kabul edemeyeceğini ifade eden göstergeler), ekran düzeni kılavuzları, renklerin ve seslerin kullanımı gibi öğeleri ve kullanıcının deneyimini etkileyen diğer etmenleri kapsamalıdır.

4.2.4.2 Kullanılabilirlik Test Spesifikasyonu

Kullanılabilirlik testine yönelik başlıca teknikler şunlardır:

- Teftiş, değerlendirme veya gözden geçirme
- Prototiplerle dinamik olarak etkileşimde bulunma
- Yazılımın doğrulanması ve sağlanması
- Anketlerin yapılması

Teftiş, değerlendirme veya gözden geçirme

Kullanılabilirlik testlerine gereksinim ve tasarım aşamasından başlanması, kullanıcıların katılımıyla kullanılabilirlik perspektifinden gereksinim ve tasarımların teftiş edilmesi veya gözden geçirilmesi, hataların erkenden tespit edilmesini sağlayacağından maliyeti düşürebilir. Bu süreçte ayrıca sezgisel değerlendirmelerden de yararlanılabilir. Bu durumda, kullanılabilirlik uzmanlarının arayüzü incelemesi ve ele alınan kullanılabilirlik prensipleriyle uyumu konusunda bir karara varması gerekebilir ("sezgisel"). Kullanıcı arayüzü görünür bir hal aldığında gözden geçirmeler daha verimli olacaktır. Örneğin, belirli bir ekrandaki fonksiyonalitenin öyküsel açıklaması ile anlaşılması ve yorumlanması daha kolaydır. Görselleştirme, dokümantasyonun kullanılabilirlik açısından gözden geçirilmesi konusunda önem taşımaktadır.

Prototiplerle dinamik olarak etkileşimde bulunma

Prototipler geliştirildiğinde test analisti, prototipler üzerinde çalışmalı ve kullanıcıların geribildirimlerini tasarımla harmanlayarak yazılımcıların prototipi geliştirmesine yardımcı olmalıdır. Bu şekilde, prototipler geliştirilebilir ve kullanıcı, nihai ürünün nasıl görüneceği konusunda daha gerçekçi bir görüşe sahip olabilir.

Yazılımın onaylanması ve sağlanması

Gereksinimlerin, yazılıma yönelik kullanılabilirlik karakteristiklerini tanımladığı durumlarda (örn. belirli bir hedefe ulaşmak için gerekli tıklama sayısı) yazılımın geliştirilmesi sırasında söz konusu karakteristiklerin göz önünde bulundurulduğunu onaylamak amacıyla test senaryoları oluşturulmalıdır.

Geliştirilen yazılımın sağlaması yapılırken fonksiyonel sistem testi için tanımlanan testler, kullanılabilirlik test senaryoları olarak geliştirilebilir. Söz konusu test senaryoları, fonksiyonel ürünlerdense öğrenilebilirlik ya da işletilebilirlik gibi özel kullanılabilirlik karakteristiklerini ölçer.

Kullanılabilirliğe yönelik test senaryoları, söz dizimini ve anlambilimi test etmek üzere geliştirilebilir. Söz dizimi, bir arayüzün yapısı ya da dil bilgisiyken (örn. girdi alanına girilebilecekler) anlambilimi, arayüzün anlamını ve amacını tanımlar (örn. kullanıcıya sunulan makul ve anlamlı sistem mesajları ve çıktı).

Kara kutu tekniklerinden (örneğin Bölüm 3.2'de tanımlananlar) ve özellikle düz metin ya da UML şeklinde tanımlanabilen kullanım senaryoları kimi zaman kullanılabilirlik testlerinde faydalanılır.

Kullanılabilirlik testine yönelik test senaryolarının kullanıcı talimatlarını, talimatları anlatmak ve geribildirim almak için test öncesi ve sonrası görüşmeler için ayrılan zamanı ve oturumların gerçekleştirilmesi için üzerinde mutabık kalınan protokolü içermesi gerekir. Bu protokol, testin nasıl gerçekleştirileceğine, zamanlamalara, not alımına ve oturumun kayıt altına alınmasına ve kullanılacak görüşme ve anket tekniklerine dair açıklamalar içermelidir.

Anketlerin yapılması

Anket teknikleri, sistemdeki kullanıcı davranışına dair gözlem ve geribildirimleri toplamak üzere uygulanabilir. Yazılım Kullanılabilirlik Ölçüm Envanteri (SUMI) ve Web Sayfası Analizi ve Ölçüm Envanteri (WAMMI) gibi standart ve halka açık anketler, daha önceki kullanılabilirlik ölçümlerini içeren bir veritabanıyla kıyaslamalar yapmayı mümkün kılmaktadır.

Buna ek olarak SUMI kullanılabilirliği dair somut ölçümler sunduğundan tamamlanma/kabul kriterlerinden oluşan bir küme elde edilebilir.

4.2.5 Erişilebilirlik Testi

Özel ihtiyaçları ya da kısıtlamaları olan kullanıcıların yazılıma erişebilirliğinin değerlendirilmesi önemlidir. Bu, engelli kişileri de kapsamaktadır. Erişilebilirlik Testi, Web İçeriği Erişilebilirlik Kuralları gibi ilgili standartları ve Engellilere Karşı Ayrımcılık Kanunu (BK, Avustralya) ve Bölüm 508 (ABD) gibi yasaları da göz önünde bulundurmalıdır. Kullanılabilirliğe benzer şekilde erişilebilirlik de tasarım aşamaları sırasında göz önünde bulundurulmalıdır. Test, genellikle entegrasyon seviyesinde gerçekleştirilir, sistem testi boyunca devam eder ve kabul testi seviyesinde son bulur. Hatalar genellikle, yazılımın belirlenen yönetmeliklere veya yazılıma yönelik tanımlanan standartlara uygun olmaması durumunda tespit edilir.

5. Gözden Geçirme – 165 dak.

Anahtar Sözcükler

yok

Gözden Geçirmelere Yönelik Öğrenme Hedefleri

5.1 Giriş

TA-5.1.1 (K2) Gözden geçirme hazırlığının test analisti için neden önemli olduğunu açıklayın

5.2 Gözden Geçirme Sırasında Kontrol Listelerinin Kullanılması

TA-5.2.1 (K4) Kullanım senaryosunu veya kullanıcı arayüzünü analiz edin ve ders programında verilen kontrol listesi bilgilerine göre problemleri tespit edin

TA-5.2.2 (K4) Gereksinimleri ya da kullanıcı hikayesini analiz edin ve ders programında verilen kontrol listesi bilgilerine göre problemlerini tespit edin

Yazılım Test ve Kalite Derneği | www.turkishtestingboard.org | info@turkistesingboard.org
Tel: +90 212 276 06 48 | Faks: +90 212 276 06 47

5.1 Giriş

Başarılı bir gözden geçirme süreci, planlama, katılım ve takip gerektirir. Test analistleri, kendi düşüncelerini paylaştıkları gözden geçirme süreci boyunca aktif katılım sergilemelidir. Herhangi bir gözden geçirme sürecindeki rollerini daha iyi anlayabilmeleri için resmi gözden geçirme eğitimine katılmaları gerekir. Tüm gözden geçirme katılımcıları, iyi bir gözden geçirmenin faydalarının bilincinde olmalıdır. Doğru bir şekilde gerçekleştirildiğinde gözden geçirmeler, yazılımda kaliteyi sağlayan en büyük ve en tasarruflu öğelerden biri olabilir.
Yürütülmekte olan gözden geçirmenin türünden bağımsız olarak test analistinin hazırlanmak için yeterli zamana ihtiyacı vardır. Bu zaman, gözden geçirme zamanını, çapraz kontrol için incelenmesi gereken dokümanlara ayrılan zamanı ve yazılımın eksikliklerini belirlemek için gerekli zamanı kapsar. Test analistinin hazırlanmak için yeterli zamanı olmazsa farklı bir bakış açısına sahip olamayacaktır. İyi bir gözden geçirme, incelenen dokümanların anlaşılmasını, eksikliklerin belirlenmesini ve tanımlanan yazılımın geliştirilmiş veya geliştirilmekte olan diğer yazılımlarla tutarlı olduğundan emin olunmasını kapsar. Örneğin, entegrasyon test planı gözden geçirilirken test analisti, entegre edilmekte olan öğelerin tümünü göz önünde bulundurmalıdır. Entegrasyona hazır olmaları için ihtiyaç duyulan koşullar nelerdir? Dokümante edilmesi gereken bağımlılıklar var mı? Entegrasyon noktalarını test etmek için kullanılabilecek veriler var mı? Bir gözden geçirme, gözden geçirilmekte olan çalışma ürünüyle sınırlı değildir; söz konusu öğenin, sistemdeki diğer öğelerle etkileşiminin de gözden geçirilmesi gereklidir.

Gözden geçirilmekte olan bir ürünün tasarımcısı kolaylıkla eleştirildiğini hissedebilir. Test analisti, gözden geçirme sırasında yaptığı yorumları mümkün olan en iyi ürünü oluşturmak için yaptığını unutmamalıdır. Bu yaklaşım kullanıldığında yorumlar yapıcı bir şekilde kelimelere dökülecek ve tasarımcıyı değil, çalışma ürününü hedef alacaktır. Örneğin gereksinim net olmadığında "Test senaryosunu yazabilmek için bu gereksinimde anlamadığım yerler var. Bunu anlamama yardımcı olur musun?" demek “Bu gereksinim çok belirsiz. Kimse burada ne demek istendiğini anlayamayacaktır.” demekten çok daha iyidir. Test analistinin gözden geçirme aşamasındaki görevi, çalışma ürünü hakkında verilen bilgilerin, test çalışmalarını desteklemeye yetecek nitelikte olmasını sağlamaktır. Bilginin eksik olması, açık olmaması ya da yeterli seviyede ayrıntı sunmaması durumunda hatanın tasarımcı tarafından düzeltilmesi gerekebilir. Eleştirel bir yaklaşımdansa olumlu bir yaklaşımın kullanılması durumunda yorumlar daha iyi anlaşılacak ve toplantı daha verimli geçecektir.

5.2 Gözden Geçirme Sırasında Kontrol Listelerinin Kullanılması

Kontrol listeleri, katılımcılara gözden geçirme sırasında belirli noktaları kontrol etmelerini hatırlatmak için kullanılır. Kontrol listeleri, gözden geçirmenin kişisellikten uzaklaştırılmasını sağlar, örn. "Bu, her gözden geçirme sırasında kullandığımız kontrol listesidir, sadece sizin çalışma ürününüzü hedef almıyoruz." Kontrol listeleri genel olabilir ve tüm gözden geçirmelerde kullanılabilir ya da belirli kalite karakteristiklerine, alanlara ya da doküman türlerine odaklanabilir. Örneğin, genel bir kontrol listesi bir dokümanı tanımlayan ID’nin olup olmadığına, doğru format ve uyumluluk öğelerine değinilebilir. Gereksinimler için özel olarak hazırlanan bir kontrol listesinde ise "olacak" ve "olması gerekir" gibi yardımcı fiillerin doğru kullanımlarına, ifade edilen gereksinimlerin test edilebilirliğine dair kontrollere ve benzer unsurlara odaklanılabilir. Gereksinimlerin formatı, kullanılacak kontrol listesinin türünü de belirtebilir. Öyküsel biçimde hazırlanmış bir gereksinimler dokümanının gözden geçirme kriterleri,

diyagramlara dayalı dokümanlarından farklı olacaktır. Kontrol listeleri programcılarının, mimarların veya test uzmanlarının becerilerine yönelik de olabilir.

Test analistleri söz konusu olduğunda, test uzmanlarının becerilerini baz alan kontrol listesinin kullanılması daha uygun olacaktır. Söz konusu kontrol listeleri aşağıdakilere benzer öğeler içerebilir:

Gereksinimler, kullanım senaryoları ve kullanıcı hikayeleri için kullanılan kontrol listelerinin odak noktası, genellikle kod veya mimari için kullanılanların odak noktasından farklıdır. Söz konusu gereksinim odaklı kontrol listeleri aşağıdaki öğeleri içerebilir:

- Gereksinimlerin her birinin test edilebilirlik durumu
- Gereksinimlerin her birine yönelik kabul kriterleri
- Mümkün olması halinde kullanım senayosu çağırma yapısının mevcudiyeti
- Gereksinimlerin/kullanım senaryolarının/kullanıcı hikayelerinin her birinin bir ID ile tanımlama
- Gereksinimlerinin/kullanım senaryolarının/kullanıcı hikayelerinin her birinin versiyonları
- Gereksinimlerin her birinin işletme/pazarlama gereksinimlerinden izlenebilmesi
- Gereksinimler ve kullanım senaryoları arasında izlenebilirlik

Yukarıdaki maddelerin sadece örnek teşkil etmesi amaçlanmıştır. Bir gereksinimin test edilemez olması halinde, diğer bir deyişle, test analistinin testi nasıl gerçekleştireceğini belirleyemeyeceği bir formatta tanımlanması durumunda, gereksinimde bir hata olduğu unutulmamalıdır. Örneğin, "Yazılımın kullanımı kolay olmalıdır" şeklindeki bir gereksinim test edilemez. Test analistinin, yazılımın kullanımının kolay olduğunu belirlemesi mümkün değildir? Bunun yerine söz konusu gereksinim "Yazılım, kullanılabilirlik standartları dokümanında belirtilen kullanılabilirlik standartlarına uyumlu olmalıdır" şeklinde olsaydı ve kullanılabilirlik standartları dokümanı gerçekten var olsaydı söz konusu gereksinim test edilebilir bir gereksinim olacaktı. Buna ek olarak söz konusu gereksinim oldukça kapsamlı bir gereksinimdir zira tek başına, arayüzdeki tüm öğeleri kapsamaktadır. Bu durumda bahsi geçen gereksinim, kapsamlı bir uygulamada bağımsız test senaryolarına kolaylıkla bölünebilir. Bu gereksinimden test senaryolarına izlenebilirliğin olması durumunda bahsi geçen kullanılabilirlik gereksiniminin değişmesi halinde tüm test senaryolarının gözden geçirilmesi ve güncellenmesi gerekecektir.

Test uzmanının testin başarılı ya da başarısız olduğunu belirleyemiyor olması durumunda ya da başarılı veya başarısız olacak bir test oluşturamıyor olması halinde de gereksinim test edilemez. Örneğin, "Sistem, kesintisiz olarak, günün 24 saati, haftanın 7 günü, yılın 365 (366) günü çalışmalıdır" şeklindeki bir gereksinim test edilemez. Kullanım senaryosu gözden geçirmelerine yönelik basit bir kontrol listesi aşağıdaki soruları içerebilir:

- Ana senaryo açık bir şekilde tanımlanmış mı?
- Alternatif senaryolar tanımlanmış mı, hataların nasıl ele alınacağı konusunda bilgi içeriyor mu?
- Kullanıcıya verilecek mesajlar tanımlanmış mı?
- Tek bir ana senaryo mu var (bir tane olmalısır, aksi halde, birden fazla kullanım senaryosu olacaktır)?
- Senaryoların her biri test edilebilir yapıda mı?

Bir uygulamanın kullanıcı arayüzünün kullanılabilirliğine yönelik basit bir kontrol listesi aşağıdakileri içerebilir:

- Alanların ve fonksiyonların her biri tanımlanmış mı?
- Tüm hata mesajları tanımlanmış mı?
- Tüm kullanıcı mesajları tanımlanmış mı ve tutarlı mı?
- Alanların sekme sıralaması tanımlanmış mı?
- Fare hareketine alternatif klavye kısa yolları bulunuyor mu?
- Kullanıcıya yönelik tanımlanmış "kısayol" tuş kombinasyonları var mı (örn. kopyala ve yapıştır)?
- Alanlar arasında bağımlılık var mı (örn. bir tarihin diğer bir tarihten daha önce ya da sonra olması gerekiyor mu)?
- Ekran düzeni var mı?
- Ekran düzeni belirtilen gereksinimlere uygun mu?
- Sistem işlem yaparken kullanıcıyı uyaracak bir gösterge var mı?
- Ekran minimum fare tıklaması gereksinimini karşılıyor mu (tanımlanmışsa)?
- Arayüz üzerinde dolaşım, kullanım senaryosu bilgilerine dayanarak kullanıcı açısından mantıklı bir şekilde sağlanabiliyor mu?
- Ekran öğrenilebilirliğe yönelik herhangi bir gereksinimi karşılıyor mu?
- Kullanıcıya sunulan herhangi bir yardım metni var mı?
- Kullanıcıya sunulan herhangi bir pop-up metni var mı?
- Kullanıcı bunun "çekici" olduğunu düşünecek mi (nesnel değerlendirme)?
- Renklerin kullanımı diğer uygulamalara ve kurumsal standartlara uygun mu?
- Ses efektleri uygun bir şekilde kullanılıyor mu ve özelleştirilebilir mi?
- Yerelleştirmeye yönelik herhangi bir gereksinimi karşılıyor mu?
- Kullanıcı ne yapacağını (anlaşılabilirlik) belirleyebiliyor mu (nesnel değerlendirme)?
- Kullanıcı ne yapması gerektiğini (öğrenilebilirlik) hatırlayabilecek mi (nesnel değerlendirme)?

Çevik yazılım projelerinde gereksinimler genellikle kullanıcı hikayeleri şeklindedir. Söz konusu hikayeler, müşteriye sunulabilecek fonksiyonaliteyi küçük birimler halinde temsil eder. Kullanıcı hikayelerine yönelik kontrol listeleri aşağıdakileri içerebilir:

- Hikaye hedef döngüye/sprinte uygun mu?
- Kabul kriterleri tanımlanmış mı ve test edilebilir durumda mı?
- Fonksiyonalite açık bir şekilde tanımlanmış mı?
- Bu hikaye ile diğerleri arasında herhangi bir bağımlılık var mı?
- Hikaye öncelik sırasına koyulmuş mu?
- Hikaye fonksiyonalitenin sadece bir öğesini mi kapsıyor?

Hikayenin yeni bir arayüzü tanımlaması halinde genel hikaye kontrol listesinin (yukarıdaki gibi) ve ayrıntılı bir kullanıcı arayüz kontrol listesinin kullanılması uygun olacaktır.

Kontrol listeleri aşağıdakilere dayanarak özelleştirilebilir:

- Kurum örn. şirket politikaları, standartları ve anlaşmalarının göz önünde bulundurulması)
- Proje / yazılım çalışmaları (örn. odak noktası, teknik standartlar, riskler)
- Gözden geçirilen nesne

İyi kontrol listeleri problemleri bulacak ve kontrol listesinde bahsedilmeyen diğer öğeler konusunda tartışmaların başlatılmasına yardımcı olabilecektir. Kontrol listelerinin kombinasyonlu olarak kullanılması, gözden geçirme sonucunda en yüksek kaliteye sahip çalışma ürününün elde edilmesini sağlamak açısından takip edilebilecek güçlü bir yöntemdir. Temel seviye ders programında belirtilenler gibi standart kontrol listelerinin kullanılması ve yukarıda belirtilenler gibi kuruma özel kontrol listelerinin geliştirilmesi, test analistinin gözden geçirmeler sırasında etkin bir rol oynamasına yardımcı olacaktır.

Gözden geçirmeler ve teftişler konusunda ayrıntılı bilgi için [Gilb93] ve [Wiegers03] dokümanlarını inceleyin.

6. Hata Yönetimi

Anahtar Sözcükler

Hata sınıflandırması, faz içerme, kök neden analizi

Hata Yönetimi için Öğrenme Hedefleri

6.2 Hata Ne Zaman Tespit Edilebilir?

TA-6.2.1 (K2) Faz içerme yaklaşımının maliyeti nasıl düşüreceğini açıklayın

6.3 Hata Raporu Alanları

TA-6.3.1 (K2) Fonksiyonel olmayan bir hata dokümante edilirken ihtiyaç duyulabilecek bilgileri açıklayın

6.4 Hata Sınıflandırma

TA-6.4.1 (K4) Belirli bir hataya yönelik sınıflandırma verilerini tanımlayın, toplayın ve kayıt altına alın

6.5 Kök Neden Analizi

TA-6.5.1 (K2) Kök neden analizinin amacını açıklayın

6.1 Giriş

Test analistleri, işletmenin ve kullanıcıların ihtiyaçları bağlamında sistem davranışını değerlendirir, örn. kullanıcı bu mesajla ya da davranışla karşılaştığında ne yapılması gerektiğini biliyor mu? Beklenen ve gerçekleşen sonuçları kıyaslayan test analisti, sistemin doğru davranıp davranmadığını belirler. Anomali (olay olarak da adlandırılır) ayrıntılı inceleme gerektiren beklenmedik bir oluşumdur. Anomali, bir hatanın yol açtığı arıza olabilir. Anomali, hata raporunun oluşturulmasına neden olabilir. Hata çözülmesi gereken gerçek bir problemdir.

6.2 Hata Ne Zaman Tespit Edilebilir?

Hatalar statik test sırasında tespit edilebilir, hatanın sonucu olan arıza ise dinamik test vasıtasıyla tespit edilebilir. Yazılım geliştirme yaşam döngüsünün her aşaması, olası arızaların tespit edilmesine ve ortadan kaldırılmasına yönelik yöntemler sunmalıdır. Örneğin, geliştirme aşaması sırasında kod ve tasarım gözden geçirmeleri hataları tespit etmek için kullanılabilir. Dinamik test sırasında test senaryoları arızaları tespit etmek için kullanılır.

Bir hata ne kadar erken tespit edilir ve düzeltilirse sistemin kalite maliyeti o kadar düşer. Örneğin, statik testler, henüz dinamik testlerin gerçekleştirilemediği durumlarda hataları bulabilir. Bu, yüksek kaliteli yazılımların üretilmesi sırasında statik testin düşük maliyetli bir yaklaşım olmasının nedenlerinden biridir.

Hata takip sistemi, test analistinin hatanın ortaya çıktığı yaşam döngüsü aşamasını ve tespit edildiği aşamayı kayıt altına almasını mümkün kılmalıdır. İki aşamanın da aynı olması durumunda kusursuz bir faz içerme işleminin yapıldığı anlaşılır. Bu, hatanın aynı aşamada ortaya çıktığı ve tespit edildiği, dolayısıyla, sonraki aşamalara "sızmadığı" anlamına gelir. Gereksinim gözden geçirmesi sırasında bulunan ve bu aşamada düzeltilen yanlış bir gereksinim bu duruma örnek olarak verilebilir. Bu şekilde gereksinim gözden geçirmesi verimli bir şekilde kullanılmakla kalmaz, aynı zamanda kurum için daha fazla maliyet teşkil edebilecek ek çalışmaların yapılmasına engel olur. Yanlış bir gereksinimin gereksinim gözden geçirmesinden kaçması ve sonraki aşamalarda yazılımcı tarafından kodlanması daha sonra test analistinin yaptığı testlerden gözden kaçması ve kullanıcı kabul testi sırasında kullanıcı tarafından yakalanması durumunda söz konusu gereksinim üzerinde yapılan tüm çalışmalar boşa gitmiş demektir (bu noktada, kullanıcının sisteme karşı güveni dahi kaybedilebilir).

Faz içerme, hataların maliyetini düşürmek açısından etkili bir yöntemdir.

6.3 Hata Raporu Alanları

Hata raporundaki alanlara (parametreler) dayanarak düzeltmenin yapılabilmesi için hata raporunun yeterli miktarda bilgi içermesi gerekmektedir. Hayata geçirilebilecek bir hata raporu:

- Tamamlanmış olmalıdır - tüm gerekli bilgiler raporda yer almalıdır
- Kısa ve öz olmalıdır - raporda gereksiz herhangi bir bilgi bulunmamalıdır
- Doğru olmalıdır - rapordaki bilgiler doğru olmalıdır ve hatayı tekrar üretmek için gerekli adımların yanı sıra beklenen ve gerçekleşen sonuçları açık bir şekilde ifade etmelidir
- Tarafsız olmalıdır - raporda gerçekler profesyonel bir biçimde ifade edilmelidir

Hata raporunda kayıt altına alınan bilgiler alanlara bölünmelidir. Alanlar ne kadar iyi tanımlanırsa hataları rapor etmek, eğilim raporlarını ve diğer özet raporları hazırlamak o kadar kolay olacaktır. Bir alana ilişkin belirli bir sayıda seçenek olması durumunda geçerli değerleri içeren bir seçenekler listesinin kullanılması, hatanın kayıt altına alınması sırasında harcanan zamanı kısaltabilir. Seçenekler listesinin, sadece geçerli seçenek sayısının kısıtlı olduğu durumlarda kullanımı daha uygundur; kullanıcı, doğru seçeneği bulmak için uzun bir listede uygun seçeneği aramak zorunda kalmamalıdır. Farklı hata raporu türleri, farklı bilgilerin girilmesini gerektirir ve hata yönetim aracı, hata türüne bağlı olarak doğru alanlara yönlendirecek esnekliğe sahip olmalıdır. Veriler, farklı alanlarda kayıt altına alınmalıdır; veri girişindeki hataları ortadan kaldırmak ve etkin raporlamayı sağlamak üzere ideal olarak veri, onay alanlarıyla birlikte sunulmalıdır.

Hata raporları, fonksiyonel ve fonksiyonel olmayan testler sırasında tespit edilen arızalara yönelik yazılır. Hata raporundaki bilgiler, beklenen ve gerçekleşen sonuçların yanı sıra senaryoyu yeniden oluşturmak için gerekli aşamaları ve verileri içermeli ve problemin tespit edildiği senaryoyu açık bir şekilde tanımlamalıdır. Fonksiyonel olmayan hata raporları, ortam, diğer performans parametreleri (örn. yükün boyutu), aşamaların sıralaması ve beklenen sonuçlar konusunda daha fazla detay gerektirebilir. Kullanılabilirliğe ilişkin bir arıza raporlanırken kullanıcının yazılımdan beklentilerinin ifade edilmesi önemlidir. Örneğin, kullanılabilirlik standardı, çalışmanın dört fare tıklamasından daha kısa sürede tamamlanacağı şeklinde ise hata raporu, belirtilen standart karşısında kaç tıklamanın gerekli olduğunu açık bir dille ifade etmelidir. Bir standardın bulunmadığı ve gereksinimlerin yazılımın fonksiyonel olmayan kalite unsurlarını ele almadığı durumlarda test uzmanı, kullanılabilirliğin kabul edilemez seviyede olduğunu tespit etmek için "uzman görüşü" testinden faydalanabilir. Bu durumda "uzman kişinin" beklentileri, hata raporunda açık bir şekilde ifade edilmelidir. Analiz dokümanlarında kimi zaman fonksiyonel olmayan gereksinimlere yer verilmediğinden test sırasında fonksiyonel olmayan arızaların dokümante edilmesi, test uzmanının "beklenen" sonuçlarla "gerçekleşen" sonuçları karşılaştırması sırasında çeşitli zorluklar arz edebilir.

Her ne kadar hata raporunun yazılmasının genel amacı bir probleme yönelik çözümün tespit edilmesi olsa da hata bilgileri, doğru sınıflandırmayı, risk analizini ve süreç iyileştirmeyi desteklemek açısından da temin edilmelidir.

6.4 Hata Sınıflandırma

Bir hatanın yaşam döngüsü boyunca hatayla ilgili girilebilecek çok sayıda sınıflandırma seviyesi bulunmaktadır. Doğru hata sınıflandırması, doğru hata raporlamasının ayrılmaz bir parçasıdır. Sınıflandırmalar, hataları gruplandırmak, testin etkinliğini değerlendirmek, yazılım geliştirme yaşam döngüsünün etkinliğini değerlendirmek ve eğilimleri belirlemek için kullanılır.

Yeni tanımlanan hatalara yönelik yaygın sınıflandırma bilgileri şunları içerir:

- Tespit edilen hataya yol açan proje faaliyeti - örn. gözden geçirme, denetim, teftiş, kodlama, test
- Hatanın ortaya çıktığı proje aşaması (biliniyorsa) - örn. analiz, tasarım, ayrıntılı tasarım, kod
- Hatanın tespit edildiği proje aşaması - örn. analiz, tasarım, ayrıntılı tasarım, kod, kod gözden geçirme, birim testi, entegrasyon testi, sistem testi, kabul testi
- Hatanın şüphelenilen nedeni - örn. gereksinim, tasarım, arayüz, kod, veri
- Tekrar edilebilirlik - örn. bir kere, kesintili, yinelenebilir
- Belirti (semptom) - örn. çökme, askıda kalma, kullanıcı arayüzü hatası, sistem hatası, performans

Yazılım Test ve Kalite Derneği | www.turkishtestingboard.org | info@turkistesingboard.org
Tel: +90 212 276 06 48 | Faks: +90 212 276 06 47

Hata incelendikten sonra daha ayrıntılı bir sınıflandırma yapılabilir:

- Kök neden – hataya yol açan yanlış, örn. süreç, kodlama hatası, kullanıcı hatası, test hatası, yapılandırma sorunu, veri sorunu, üçüncü şahıslara ait yazılım, harici yazılım problemi, dokümantasyon problemi
- Kaynak - yanlışın yapıldığı çalışma ürünü, örn. gereksinim, tasarım, ayrıntılı tasarım, mimari, veritabanı tasarımı, kullanıcı dokümantasyonu, test dokümantasyonu
- Tip - örn. mantık problemi, hesaplama problemi, zamanlama sorunu, veri işleme, geliştirme

Hata ortadan kaldırıldığında (ya da ertelendiğinde veya onaylanmadığında) aşağıdakiler gibi daha ayrıntılı bir sınıflandırmanın yapılması mümkündür:

- Çözüm - örn. kod değişikliği, dokümantasyon değişikliği, ertelenme, hata değil, tekrar raporlanmış hata
- Düzeltici eylem - örn. gereksinim gözden geçirme, kodu gözden geçirme, birim testi, yapılandırma dokümantasyonu, veri hazırlığı, değişiklik yapılmadı

Söz konusu sınıflandırma kategorilerine ek olarak hatalar sıklıkla önem derecelerine ve önceliklerine göre sınıflandırılır. Buna ek olarak, projeye bağlı olmak kaydıyla, güvenlik üzerindeki etkisine, proje programı üzerindeki etkisine, proje maliyetlerine, proje riskine ve proje kalitesi üzerindeki etkisine göre sınıflandırmak da makul olabilir. Söz konusu sınıflandırmalar, bir çözümün ne kadar zaman içinde sunulacağına ilişkin sözleşmelerde göz önünde bulundurulabilir.

Sınıflandırmanın son aşaması, nihai çözümdür. Hatalar sıklıkla çözümlerine göre bir arada gruplandırılır. Örn. çözüldü/onaylandı, kapatıldı/hata değil, ertelendi, açık/çözülmedi. Bu sınıflandırma, genellikle proje boyunca yapılır zira hatalar, yaşam döngüleri boyunca takip edilirler.

Kurum tarafından kullanılan sınıflandırma değerleri genellikle özelleştirilir. Yukarıda, sektörde kullanılan yaygın değerlerden bazıları örnek olarak verilmiştir. Sınıflandırma değerlerinin, kullanışlı olmaları için tutarlı bir şekilde kullanılmaları gerekmektedir. Gereğinden fazla sınıflandırma alanı olması, hatanın açılışının ve işleme koyulmasının fazla zaman almasına neden olacaktır. Bir araç tarafından toplanan sınıflandırma değerlerinin özelleştirilmesine yönelik beceri genellikle araç seçiminde göz önünde bulundurulan önemli bir etmendir.

6.5 Kök Neden Analizi

Kök neden analizinin amacı, hatanın ortaya çıkmasına neden olan unsuru belirlemek ve hataların büyük bir kısmından sorumlu olan kök nedenleri ortadan kaldıracak süreç değişikliklerini desteklemektir. Kök neden analizi, genellikle hatayı inceleyip problemi çözen ya da problemin çözülmemesi gerektiğine ya da çözülemeyeceğine karar veren kişi tarafından gerçekleştirilir. Bu ise çoğunlukla yazılımcının kendisidir. Birincil kök neden değerinin belirlenmesi, problemin nedenine yönelik bir tahmin yapabilecek olan test analisti tarafından gerçekleştirilir. Çözüm onaylanırken test analisti, yazılımcı tarafından girilen kök nedeni onaylayacaktır. Kök nedenin belirlendiği noktada hatanın ortaya çıktığı aşamanın belirlenmesi veya onaylanması da yaygın görülen bir eylemdir. Tipik kök nedenlerden bazıları şunlardır:

- Belirsiz gereksinim
- Eksik gereksinim
- Yanlış gereksinim
- Yanlış tasarım
- Yanlış Arayüz
- Kodda mantık hatası
- Hesaplama hatası

- Donanım hatası
- Arayüz hatası
- Geçersiz veri

Kök neden bilgisi, hataların ortaya çıkmasına neden olan yaygın sorunların belirlenmesi için bir araya getirilir. Örneğin hataların büyük bir kısmı belirsiz gereksinimlerden kaynaklanıyorsa etkin gereksinim gözden geçirmelerinin yürütülmesi için daha fazla çaba harcanması gerekir. Benzer şekilde arayüz hataları yazılım gruplarında karşılaşılan yaygın bir sorun ise müşterek tasarım görüşmelerinin yapılması gerekebilir.

Süreç iyileştirmesi için kök neden bilgilerinin kullanılması, kurumun etkin süreç değişikliklerinin faydalarını izleyebilmesini ve belirli bir kök nedene bağlanabilecek hataların maliyetini nicel olarak görebilmesini mümkün kılar. Bu, proje planının değiştirilmesinin yanı sıra ek araç ve ekipmanların satın alınmasını gerektirebilecek süreç değişiklikleri konusunda bütçe sağlanmasına da yardımcı olabilir. ISTQB Uzman Seviye ders programı “Test Sürecinin İyileştirilmesi” [ISTQB_EL_ITP] kök neden analizini daha ayrıntılı bir şekilde ele almaktadır.

7. Test Araçları – 45 dak.

Anahtar Sözcükler

Aksiyon kelimesi güdümlü test, test verisi hazırlama aracı, test tasarım aracı, test yürütme aracı

Test Araçlarına Yönelik Öğrenme Hedefleri

7.2 Test Araçları ve Otomasyon

TA-7.2.1 (K2) Test verisi hazırlama araçlarının, test tasarım araçlarının ve test yürütme araçlarının kullanımının faydalarını açıklayın

TA-7.2.2 (K2) Aksiyon kelimesi güdümlü otomasyonda test analistinin rolünü açıklayın

TA-7.2.3 (K2) Test otomasyonunda sorun giderilirken atılacak adımları açıklayın

Yazılım Test ve Kalite Derneği | www.turkishtestingboard.org | info@turkistesingboard.org
Tel: +90 212 276 06 48 | Faks: +90 212 276 06 47

7.1 Giriş

Test araçları, test çalışmalarının verimini ve doğruluğunu büyük ölçüde iyileştirebilir ancak bu yalnızca doğru araçların doğru şekilde kullanılmasıyla mümkün olur. İyi bir şekilde yürütülen testlerde göz önünde b ulundurulması gereken unsurlardan biri de test araçlarının yönetimidir. Test araçlarının karmaşıklığı ve uygulanabilirliği, geniş ölçüde farklılık göstermektedir ve sektördeki araçlar kesintisiz bir şekilde değişmektedir. Otomasyon araçlarına, çok sayıda ü cretsiz yazılım ya da paylaşımlı yazılım sağlayan araç tedarikçilerinin yanı sıra ticari araç tedarikçilerinden de ulaşılabilir.

7.2 Test Araçları ve Otomasyon

Test analistinin görevlerinin büyük bir kısmını yerine getirebilmesi için araçları verimli bir şekilde kullanabilmesi gerekir. Hangi aracın ne zaman kullanılacağını bilmek, test analistinin verimini artırabilir ve verilen zaman içinde daha iyi bir test kapsamının elde edilmesine yardımcı olabilir.

7.2.1 Test Tasarım Araçları

Test tasarım araçları, test senaryolarının oluşturulmasına ve test verilerinin test için ele alınmasına yardımcı olmak üzere kullanılırlar. Söz konusu araçlar, test analisti tarafından temin edilen özel gereksinim dokümanı formatlarını, modelleri (örn. UML) ya da girdileri kullanarak çalışabilir. Test tasarımı araçları, genellikle özel gereksinim yönetim araçları gibi özel ürünlerle ve özel formatlarla birlikte çalışacak şekilde tasarlanır ve yapılandırılır.

Test tasarımı araçları; test kapsamında ya da ürün riski azaltma faaliyetlerinde hedeflenen seviyenin elde edilmesi için gerekli olan test türlerini belirlerken test analistinin kullanabileceği bilgileri temin edebilir. Örneğin, sınıflandırma ağacı araçları, seçilen kapsam kriterine dayanarak tam kapsama ulaşmak açısından gerekli kombinasyonlar kümesini oluşturur (ve görüntüler). Bu bilgi, yürütülmesi gereken test senaryolarını belirlemek üzere test analisti tarafından kullanılabilir.

7.2.2 Test Verisi Hazırlama Araçları

Test verisi hazırlama araçları çok sayıda fayda sunar. Bazı test verisi hazırlama araçları, gerekli kapsam seviyesine ulaşmak amacıyla test sırasında gerekli verileri tespit etmek için analiz dokümanı gibi bir dokümanı ya da kaynak kodu analiz edebilir. Diğer test verisi hazırlama araçları, canlı ortamdan bir veri kümesini alabilir ve verinin dahili bütünlüğünü korurken kişisel bilgileri silmek için söz konusu veri kümesini "anonimleştirebilir". Anonimleştirilen veri, kişisel bilgilerin kötüye kullanımına ya da güvenlik sızıntılarına dair herhangi bir risk olmaksızın testte kullanılabilir. Büyük hacimli gerçekçi verilerin gerekli olduğu durumlarda bu özellikle önemlidir. Diğer veri üretimi araçları, test verilerini belirli giriş parametrelerinden oluşan kümelerden üretmek için kullanabilir (diğer bir deyişle, rastgele testte kullanılması için). Bunlardan bazıları, test analistinin gerekli göreceği girdileri belirlemek üzere veritabanı yapısını analiz edecektir.

7.2.3 Test Yürütme Otomasyonu Araçları

Test yürütme araçları, genellikle testleri gerçekleştirmek ve testlerin sonuçlarını kontrol etmek amacıyla testin farklı seviyelerinde test analisti tarafından kullanılır. Test yürütme aracının kullanımına ilişkin hedef, tipik olarak aşağıdakilerden biri ya da birden fazlasıdır:

- Maliyeti düşürmek (çaba ve/veya zaman bağlamında
- Daha fazla test gerçekleştirmek
- Aynı testi farklı ortamlarda gerçekleştirmek
- Test yürütümünü daha tekrarlanabilir kılmak
- Manuel olarak gerçekleştirilmesi mümkün olmayan testleri gerçekleştirmek (diğer bir deyişle, büyük veri onaylama testleri)

Söz konusu hedefler sıklıkla maliyeti düşürme ve kapsamı genişletme hedefleriyle kesişmektedir.

7.2.3.1 Uygulanabilirlik

Test yürütme araçlarına yapılan yatırımın geri dönüşü, regresyon testleri otomasyona geçirilirken en yüksek seviyeye ulaşır. Bunun nedeni, bakımın düşük seviyeli olması ve testlerin tekrarlanarak yürütülmesidir. Duman testlerinin otomasyona geçirilmesi de etkili olmaktadır. Çünkü bu testler sıklıkla kullanılmakta, sonuçlarına hemen ihtiyaç duyulmakta ve her ne kadar bakım maliyeti yüksek olsa da yeni sürümün kesintisiz hayata geçirilebilmesi için diğer testlerle hızlıca entegrasyonuna ihtiyaç duyulmaktadır.

Test yürütme araçları, sistem ve entegrasyon testi seviyelerinde yaygın bir şekilde kullanılmaktadır. API test araçları başta olmak üzere bazı araçlar, birim testi seviyesinde de kullanılabilir. Araçları en uygun oldukları yerde kullanmak, yatırımın geri dönüşünün alınmasına yardımcı olacaktır.

7.2.3.2 Test Otomasyonu Araçları Hakkında Temel Bilgi

Test yürütme araçları, sıklıkla betik dili olarak adlandırılan programlama dilinde yazılmış bir dizi talimatı yürüterek çalışır. Aracın kullandığı talimatlar; girdileri, girdi sıralamasını, girdiler için kullanılan belirli değerleri ve beklenen çıktıları açıklayan oldukça ayrıntılı bir seviyededir. Bu, özellikle aracın grafiksel kullanıcı arayüzüyle (GUI) etkileşim kurduğu durumlarda ayrıntılı betikleri, test edilmekte olan yazılımdaki (software under test - SUT) değişikliklere karşı duyarlı hale getirir.

Test yürütme araçlarının çoğu, gerçekleşen sonuçlarla belirlenmiş beklenen sonuçları kıyaslama becerisi olan bir karşılaştırıcı içerir.

7.2.3.3 Test Otomasyonu Uyarlaması

Test otomasyonunda genel eğilim (tıpkı programlamada olduğu gibi) ayrıntılı alt seviye komutlardan daha üst seviye dillere doğru hareket ederken kütüphaneleri, makroları ve alt programları kullanmak şeklindedir. Anahtar kelime odaklı ve aksiyon kelimesi odaklı gibi tasarım teknikleri, söz konusu "anahtar kelimenin" ya da "aksiyon kelimesinin" yanı sıra bir dizi komutu ve referansı da içerir. Bu, test analistinin test senaryolarını insan dilinde yazarken altta yatan programlama dilini ve alt seviye fonksiyonları göz ardı etmesine olanak tanır. Söz konusu modüler yazım tekniğinin kullanılması, fonksiyonalitede ve test edilmekte olan yazılımın arayüzünde değişiklik yapılması sırasında bakım çalışmalarını kolaylaştırır. [Bath08] Anahtar kelimelerin otomatik betiklerde kullanılması aşağıda detaylı olarak ele alınmıştır.

Anahtar kelimelerin ya da aksiyon kelimelerinin oluşturulması sırasında rehberlik etmesi açısından modellerden faydalanılabilir. Test analisti, sıklıkla gereksinimlerde ele alınan iş süreçlerine bakarak test edilmesi gereken önemli süreçleri belirleyebilir. Süreçler sırasında ortaya çıkabilecek karar noktaları dahil olmak üzere söz konusu süreçlerin aşamaları bunun ardından belirlenebilir. Karar noktaları aksiyon kelimelerine dönüşebilir. İş süreci modellemesi, iş süreçlerinin dokümante edilmesine yönelik bir yöntemdir ve önemli süreçleri ve karar noktalarını tanımlamak üzere kullanılabilir. Modelleme, iş kurallarına ve süreç tanımlamalarına dayanarak elde edilen girdiler üzerinden hareket edecek araçlar kullanılarak ya da manuel olarak gerçekleştirilebilir.

7.2.3.4 Test Otomasyonunda Başarı

Hangi testlerin otomasyona geçirileceği konusuna karar verilirken aday test senaryolarının ya da aday test gruplarının her biri, otomasyona uygun olup olmadıkları açısından değerlendirilmelidir. Başarısız olan otomasyon projelerinin çoğu, otomasyondan gerçekten fayda sağlanıp sağlanmayacağı değerlendirilmeksizin hali hazırda mevcut manuel test senaryolarının otomasyona geçirilmesinden kaynaklanmaktadır. Test senaryosu grubunun manuel, yarı otomatik ve tamamen otomatik testleri içinde bulundurması olasıdır.

Test otomasyonu projesi hayata geçirilirken aşağıdaki hususların göz önünde bulundurulması gerekir:

Olası faydaları:

- Test otomasyonunda test yürütme zamanı daha öngörülebilir olacaktır
- Test otomasyonuyla regresyon testi ve hata onaylama süreçleri projenin sonlarına doğru daha hızlı ve güvenilir sonuçlar verecektir
- Test uzmanının ya da test ekibinin teknik gelişimi otomasyona geçilmesiyle daha da zenginleştirilebilir
- Test otomasyonu, her sürüm ya da döngü için regresyon testlerinin daha kolay yapılmasını sağladığından tekrarlamalı ve artımlı yazılım geliştirme yaşam döngülerinde oldukça faydalı olabilir
- Belirli test türlerinin hayata geçirilmesi, ancak test otomasyonu ile mümkün olabilir (örn. büyük veri çalışmaları)

- Test yürütümünün otomasyonu, hızlı ve tutarlı girdi ve onaylama sunduğundan büyük veri girdileri, dönüşüm ve karşılaştırma testleri açısından manuel testlerden daha düşük maliyetli olabilir.

Olası riskler:

- Tamamlanmış, etkisiz ya da yanlış manuel testler "oldukları gibi" otomasyona geçirilebilir.
- Test yazılımının bakımı zor olabilir; test edilmekte olan yazılım değiştirildiği zaman birden fazla değişiklik yapılması gerekebilir.
- Test uzmanının, test yürütmesine doğrudan katılımı düşebilir ve söz konusu durum daha az sayıda hatanın algılanmasına yol açabilir.
- Test ekibi, otomatik araçları etkin bir şekilde kullanmak için gerekli becerilere sahip olmayabilir.
- Genel test kapsamına katkıda bulunmayan ilgisiz testler otomasyona geçirilebilir.
- Yazılımın stabil bir hal aldıkça testler üretkenliğini yitirebilir (pesticide paradoksu – antibiyotik direnci)

Manuel testleri olduğu gibi otomasyona geçirmek çok katma değerli değildir; otomasyondan daha çok faydalanmak amacıyla test senaryolarının yeniden tanımlanması gerekebilir. Bu, test senaryolarının yeniden düzenlenmesini, sabit tanımlanmış değerler yerine değişkenlerin kullanımını ve test aracının özelliklerinden daha fazla faydalanılmasını kapsar. Test yürütme araçları, analiz ve raporlama olanakları sunarken birden fazla testi her iki yönde ilerletme, testleri gruplama, tekrarlama ve yürütme sıralamasını değiştirme becerilerine sahiptir.

Test otomasyonu araçlarının çoğunda etkin ve verimli test yapılabilmesi için programlama becerilerine sahip olunması gerekir. Test gruplarının özenle oluşturulmaması halinde zorlukla güncellendiği ve yönetildiği görülmektedir. Araçların sunduğu tüm avantajlardan yararlanmak adına test uzmanlarına, programlama ve tasarım teknikleri konusunda yeterli eğitim verilmelidir.

Test planlaması sırasında girdi verilerinin geçerliliğinin ve kapsamının gözden geçirilmesinin yanı sıra testin nasıl çalıştığına dair bilgi edinmek ve doğru şekilde çalıştığından emin olmak amacıyla otomatik test senaryolarının düzenli aralıklarla manuel olarak gerçekleştirilmesi için zaman ayırmak çok önemlidir.

7.2.3.5 Anahtar Kelime Odaklı Otomasyon

Anahtar kelimeler (kimi zaman aksiyon kelimeleri olarak da adlandırılır) genellikle bir sistemle kurulan üst seviye etkileşimi temsil etmek üzere kullanılır. Anahtar kelimelerin her biri, tipik olarak bir aktör ile test edilmekte olan sistem arasındaki detaylı bir dizi etkileşimi temsil etmek üzere kullanılır. Anahtar kelime sıralamaları (ilgili test verileri dahil olmak üzere) test senaryolarını tanımlamak için kullanılır. [Buwalda01]

Test otomasyonunda anahtar kelimeler, bir ya da daha fazla yürütülebilir test betiği olarak uyarlanır. Araçlar, bir dizi anahtar kelimeyle yazılan test senaryolarını okur. Betikler, belirli anahtar kelimelere kolaylıkla eşlenmeleri açısından oldukça modüler bir şekilde uya rlanır. Söz konusu modüler betikleri uyarlamak için programlama becerilerine ihtiyaç duyulur.

Anahtar kelime odaklı otomasyonun başlıca avantajları şunlardır:

- Belirli bir uygulamayla ya da iş alanıyla ilgili olan anahtar kelimeler alan uzmanları tarafından tanımlanabilir. Bu, test senaryosu oluşturma sürecini daha verimli bir hale getirir.
- Esasen alan konularında deneyimli olan bir kişi, altta yatan otomasyon kodunu anlamasına gerek olmaksızın otomatik test senaryosu yürütümünden faydalanabilir (anahtar kelimeler betik olarak uyarlandıktan sonra).
- Anahtar kelimeler kullanılarak yazılan test senaryolarının bakımı daha kolaydır zira test edilmekte olan yazılımın ayrıntıları değişirse modifikasyona ihtiyaç duyma olasılıkları düşüktür.
- Test senaryosu uyarlamalarından bağımsızdır. Anahtar kelimeler, farklı betik dilleri ve araçları kullanılarak uyarlanabilir.

Anahtar kelime/aksiyon kelimesi bilgisini kullanan otomasyon betikleri (otomasyon kodu) genellikle yazılımcılar veya teknik test analistleri tarafından yazılırken test analistleri, genellikle anahtar kelime/aksiyon kelimesi verilerini oluşturur ve bunların bakımını yapar. Anahtar kelime odaklı otomasyon genellikle sistem testi aşamasında gerçekleştirilse de kod yazılımı, entegrasyon aşamalarıyla aynı anda başlatılabilir. Tekrarlamalı ortamlarda test otomasyonu geliştirme kesintisiz bir süreçtir.

Girdi anahtar kelimeleri ve veriler oluşturulduktan sonra test analistleri genellikle anahtar kelime odaklı test senaryolarını yürütme ve ortaya çıkabilecek arızaları analiz etme sorumluluklarını üstlenir. Herhangi bir anomali tespit edildiği zaman test analisti, problemin anahtar kelimelerle, girdi verileriyle, otomasyon betiğiyle ya da test edilmekte olan uygulamayla ilgili olup olmadığını belirlemek için arızanın nedenini incelemelidir. Genellikle sorun gidermenin ilk adımı, arızanın uygulamanın kendisinde olup olmadığını görmek için aynı testi aynı verilerle manuel olarak yürütmektir. Herhangi bir arızanın yaşanmaması

durumunda test analisti, problemin bir önceki aşamada (muhtemelen yanlış veriler üreterek) oluşup sürecin sonraki aşamalarına kadar ortaya çıkmadığından emin olmak için arızaya yol açan test dizilimini gözden geçirmelidir. Test analistinin arızanın nedenini belirleyememesi halinde sorun giderme bilgileri, ayrıntılı analiz için teknik test analistine ya da yazılımcıya devredilmelidir.

7.2.3.6 Otomasyon Çalışmalarındaki Başarısızlıkların Nedenleri

Test otomasyon projeleri, genellikle hedeflerine ulaşamazlar. Söz konusu başarısızlıklar, test aracının kullanımına yönelik yetersiz esneklikten, test ekibinin programlama becerilerindeki yetersizliklerden ya da test otomasyonu ile çözülebilecek problemlere dair gerçekçi olmayan beklentilerden kaynaklanabilir. Test otomasyonunun herhangi bir yazılım projesindeki gibi yönetim, çaba, beceri ve ilgi gerektirdiğinin unutulmaması önemlidir. Uygun mimarinin oluşturulması, uygun tasarım uygulamalarının takibi, yapılandırma yönetiminin sağlanması ve iyi kodlama uygulamalarının takibi için zaman ayrılmalıdır. Otomatik test betiklerinin test edilmesi gerekir zira bunlarda da bir yazılımda olabileceği gibi çeşitli hatalar bulunabilir. Betiklerin performansını iyileştirmek için düzenlenme gerekebilir. Araçların kullanılabilirliğinin sadece yazılımcı açısından değil, aynı zamanda betikleri yürütmek amacıyla aracı kullanacak olan insanlar açısından da göz önünde bulundurulması gerekmektedir. Araç ile kullanıcı test senaryolarına erişim sağlayacak bir arayüzün tasarlanması gerekli olabilir.

8. Referanslar

8.1 Standartlar

- [ISO25000] ISO/IEC 25000:2005, Yazılım Mühendisliği - Yazılım Ürünü Kalite Gereksinimleri ve Değerlendirme (SQuaRE)
  Bölüm 1 ve 4
- [ISO9126] ISO/IEC 9126-1:2001, Yazılım Mühendisliği - Yazılım Ürünü Kalitesi, Bölüm 1 ve 4
- [RTCA DO-178B/ED-12B]: Havacılık Sistemlerinde Yazılımlarla İlgili Göz Önünde Bulundurulacak Hususlar ve Ekipman Sertifikasyonu, RTCA/EUROCAE ED12B.1992.
  Bölüm 1

8.2. ISTQB Dökümanları

- [ISTQB_AL_OVIEW] ISTQB İleri Seviyeye Genel Bakış, Versiyon 1.0
- [ISTQB_ALTM_SYL] ISTQB İleri Seviye Test Yöneticisi Ders Programı, Versiyon 1.0
- [ISTQB_ALTTA_SYL] ISTQB İleri Seviye Teknik Test Analisti Ders Programı, Versiyon 1.0
- [ISTQB_FL_SYL] ISTQB Temel Seviye Ders Programı, Versiyon 2011
- [ISTQB_GLOSSARY] ISTQB Yazılım Testi Terimler Sözlüğü, Versiyon 2.2, 2012

8.3. Kitaplar

[Bath08] Graham Bath, Judy McKay, “The Software Test Engineer’s Handbook”, Rocky Nook, 2008, ISBN 978-1-933952-24-6

[Beizer95] Boris Beizer, “Black-box Testing”, John Wiley & Sons, 1995, ISBN 0-471-12094-4

[Black02]: Rex Black, “Managing the Testing Process (2nd edition)”, John Wiley & Sons: New York, 2002, ISBN 0-471-22398-0

[Black07]: Rex Black, “Pragmatic Software Testing”, John Wiley and Sons, 2007, ISBN 978-0-470-12790-2

[Buwalda01]: Hans Buwalda, “Integrated Test Design and Automation”, Addison-Wesley Longman, 2001, ISBN 0-201-73725-6

[Cohn04]: Mike Cohn, “User Stories Applied: For Agile Software Development”, Addison-Wesley Professional, 2004, ISBN 0-321-20568-5

[Copeland03]: Lee Copeland, “A Practitioner's Guide to Software Test Design”, Artech House, 2003, ISBN 1-58053-791-X

[Craig02]: Rick David Craig, Stefan P. Jaskiel, “Systematic Software Testing”, Artech House, 2002, ISBN 1-580-53508-9

[Gerrard02]: Paul Gerrard, Neil Thompson, “Risk-based e-business Testing”, Artech House, 2002, ISBN 1-580-53314-0

[Gilb93]: Tom Gilb, Graham Dorothy, “Software Inspection”, Addison-Wesley, 1993, ISBN 0-201-63181-4

[Graham07]: Dorothy Graham, Erik van Veenendaal, Isabel Evans, Rex Black “Foundations of Software Testing”, Thomson Learning, 2007, ISBN 978-1-84480-355-2

[Grochmann94]: M. Grochmann (1994), Test case design using Classification Trees, in: conference proceedings STAR 1994

[Koomen06]: Tim Koomen, Leo van der Aalst, Bart Broekman, Michiel Vroon "TMap NEXT, for result driven testing", UTN Publishers, 2006, ISBN 90-72194-80-2

[Myers79]: Glenford J. Myers, “The Art of Software Testing”, John Wiley & Sons, 1979, ISBN 0-471-46912-2

[Splaine01]: Steven Splaine, Stefan P. Jaskiel, “The Web-Testing Handbook”, STQE Publishing, 2001, ISBN 0-970-43630-0

[vanVeenendaal12]: Erik van Veenendaal, “Practical risk-based testing – The PRISMA approach”, UTN Publishers, The Netherlands, ISBN 9789490986070 [Wiegers03]: Karl Wiegers, “Software Requirements 2”, Microsoft Press, 2003, ISBN 0-735-61879-8

[Whittaker03]: James Whittaker, “How to Break Software”, Addison-Wesley, 2003, ISBN 0-201-79619-8

[Whittaker09]: James Whittaker, “Exploratory Software Testing”, Addison-Wesley, 2009, ISBN 0-321-63641-4

8.4 Diğer Referanslar

Aşağıdaki referanslar, internet üzerindeki ve farklı ortamlardaki mevcut bilgilere işaret etmektedir. Her ne kadar bu İleri Seviye ders programının yayınlanma tarihinde söz konusu referanslar kontrol edilmiş olsa da ISTQB, söz konusu referansların geçerli olmaması durumunda kesinlikle sorumlu tutulamaz.

- Bölüm 3
  - Czerwonka, Jacek www.pairwise.org
  - Hata Sınıflandırması: http://www.testingeducation.org/a/bsct2.pdf
  - Boris Beizer’ın çalışmasına dayalı Örnek Hata Sınıflandırması: inet.uni2.dk/~vinter/bugtaxst.doc
  - Çeşitli sınıflandırmalara iyi bir genel bakış: testingeducation.org/a/bugtax.pdf
  - James Bach imzalı Bulgusal Risk Bazlı Test
  - “Keşif & Risk-Bazlı Test (2004) http://www.testingeducation.org"
  - Keşif Testini Keşfetmek, Cem Kaner ve Andy Tikam , 2003
  - Pettichord, Bret, “An Exploratory Testing Workshop Report”,

- Bölüm 4
  - http://www.testingstandards.co.uk

9. Dizin