Yazılım Yapısal Kapsama Analizi

Nezir ERTÜRK[1] Erhan YÜCEER[2] Cem POLAT[3] Mehmet ÖNER[4]

[1,2,3,4]Aydın Yazılım ve Elektronik Sanayi A.Ş. (AYESAŞ)
Silikon Blok 1.Kat No:1 Teknokent-ODTÜ 06531 ANKARA

[1]e-posta: nezire@ayesas.com [2]e-posta: erhany@ayesas.com
[3]e-posta: cemp@ayesas.com [4]e-posta: mehmeto@ayesas.com

Özet

Bu bildiride Yazılım Doğrulama Sürecinin alt süreçlerinden olan Yapısal Kapsama Analizi (YKA), amacı ve AYESAŞ'ın yaklaşımı anlatılmaktadır. Bu bildiri ile AYESAŞ edindiği YKA deneyimini ulusal yazılım endüstrisine aktarmayı amaçlamaktadır. Yazılım Doğrulama Sürecinin kompleks bölümlerinden biri olan YKA, yazılımın testler tarafından koşturulamayan bölümlerini tespit edip, ilave testler ile kapsamayı arttırmayı hedefleyen bir süreçtir. Bildiri güvenlik-kritik sistemlerin ve bu sistemlerin doğrulama süreçlerinin bir tanımı ile başlamaktadır. Daha sonra, analiz ile ilgili yaklaşımlar ve yapısal kapsama analizinin çeşitleri anlatılmaktadır. Sürecin kompleks ve uygulanması zor olması nedeniyle çeşitli yazılım araçları ile desteklenmesi tavsiye edilmektedir. YKA sürecinin uygulanması sırasında AYESAŞ tarafından kullanılan yazılım araçları da bu bildiride özetlenmiştir. YKA uygulanan yazılımın genellikle güvenlik-kritik yazılım olması sebebi ile AYESAŞ bu süreci iyi tanımlanmış bir iş akışı dahilinde yürütmektedir. Bildiri bu iş akışını anlatan bir bölümle tamamlanmaktadır.

Abstract

In this paper, Structural Coverage Analysis (SCA) process, its purpose and AYESAS' approach is explained. The purpose of this paper is to share the SCA experience of AYESAS with Turkey's software industry. Being one of the most complex areas of software verification, SCA is the process of finding areas of software that is not exercised by a set of tests, creating additional tests to increase coverage and also resolving a quantitative measure of code coverage. The paper starts with a description of safety-critical systems and the verification process activities of safety-critical software. Then the analysis approach and the types of SCA are explained. Since this process is complex and implementation is tough, tool support is recommended. The tools that are used in AYESAS for performing SCA are also described. Since the software that is under SCA is safety-critical, there is a strict flow of operation for SCA in AYESAS. The paper concludes with a brief explanation of this procedural flow of SCA.

1. Giriş

Günümüzün artan yazılım mühendisliği ihtiyaçlarına cevap verebilmek için bir çok farklı süreç modeli ve standart ortaya konulmuştur. AYESAŞ savunma, havacılık ve bilişim sektörlerinde üstlendiği yazılım projelerinde bu farklı süreç modeli ve standartları uygulama ve deneyim kazanma şansı bulmaktadır. Havacılık sektöründeki güvenlik-kritik (safety-critical) yazılım

gereksinimlerine cevap veren DO-178B [1]/ ED-12B [2] uygulama maliyeti en yüksek standartlardan biridir. Seviye (Level) A, B, C, D ve E olmak üzere 5 faklı güvenlik seviyesi için farklı hedefler ortaya koyan standart ilk üç seviye için "Yapısal Kapsama Analizi" (YKA) (Structural Coverage Analysis-SCA) yapılmasını istemektedir.

En basit anlatımı ile yapısal kapsama analizi, nesne kodun tamamının gereksinim tabanlı olarak test edildiğini doğrulamaktır. YKA, yazılım geliştirme ve doğrulama aktiviteleri esnasında üretilen gereksinimler, tasarım, kaynak kod, nesne kodu, test senaryoları, izlenebilirlik (traceability) matrisleri gibi tüm ürünlerin birbiri ile uyumlarını ve olası bütün senaryoların sistem kullanılmaya başlamadan laboratuvar ortamında oluşturularak test edilmesini garanti altına almaktadır.

Bir çok yazılım ürününün uyumluluğunun sorgulandığı bu analiz farklı seviyelerde çok sayıda veriyi birlikte kullanmayı gerektiren kompleks bir süreçtir. 6 yıllık DO-178B [1] deneyimi ile AYESAŞ'ın geliştirdiği yöntem ve araçlar bu zorlukların aşılmasını ve sürecin gerektirdiği aktivitelerin uygulanmasını kolaylaştırmaktadır.

2. AYESAŞ Hakkında

AYESAŞ, yüksek teknoloji gerektiren komuta kontrol haberleşme ve bilgi sistemleri alanında ulusal savunma sanayimizin kabiliyetlerini arttırmak üzere Milli Savunma Bakanlığımızın yönlendirmeleri ile 1990 yılında kurulmuştur. Geçen 15 yıllık sürede kuruluş misyonuna hizmet edecek şekilde ulusal hava savunma sistemimizin omurgasını teşkil eden C4I sistemi dahil olmak üzere bir çok projede başarı ile yer almış, sistem mühendisliği, yazılım, donanım ve üretim kabiliyetlerini zenginleştirerek bugüne taşımıştır. AYESAŞ'ın sahip olduğu başlıca kabiliyetler:

- C4I Sistemleri Tasarımı, Üretimi ve Depo Seviyesi Bakımı
- Radar Entegrasyonu
- Taktik Veri İletişimi
- Gerçek Zamanlı Yazılım Geliştirme
- Aviyonik (hava elektroniği) Sistemler için Yazılım Geliştirme (DO-178B Uyumlu)
- Bağımsız Yazılım Doğrulama ve Geçerleme (IV&V) (DO-178B Uyumlu)
- Tersine Mühendislik (DO-178B Uyumlu)
- Elektronik ve Elektromekanik Sistem Tasarımı
- Elektronik ve Elektromekanik Üretim ve Birleştirme
- Sistem Mühendisliği ve Sistem Entegrasyonu
- Hava Platformları Kablaj Tasarımı ve Üretimi
- Elektronik Ekipmanlar İçin Güçlendirme (Ruggedization)
- Simülasyon Sistemleri
- Bilişim Teknolojileri ve Danışmanlığı

AYESAŞ'ın misyonuna kurulduğu ilk günden bu yana destek veren Yazılım Grubu, komuta kontrol, gerçek zamanlı yazılım, taktik veri iletişimi, aviyonik (hava elektroniği) yazılımlar ve simülasyon sistemleri konularında deneyimli yaklaşık 100 kadar çalışanı ve 15 yıllık deneyimi ile askeri ve sivil sektöre ihtiyaca yönelik ürün ve hizmetler sunmaktadır. Yazılım Grubunun yazılım geliştirme süreçleri SEI SW CMM 3 olgunluk seviyesindedir. Bunun yanısıra Yazılım Grubu, ISO 9001:2000 ve NATO AQAP-150 kalite belgelerine de sahiptir.

1999 yılından bu yana sürdürülen aviyonik yazılım projeleri sayesinde AYESAŞ güvenlik-kritik yazılım geliştirme ve doğrulama alanlarında önemli deneyim kazanmıştır. Havacılık sektörü tarafından ortaya konulan ve kullanılan DO-178B dokümanı kısa sürede güvenlik-kritik yazılım ihtiyacı olan diğer sektörler tarafından da kullanılmaya başlanılmıştır. AYESAŞ DO-178B standardına uygun olarak 400 bin kod satırından fazla yazılımın geliştirilmesi ve/veya doğrulanmasında görev almıştır. Tüm bu projelerde sertifikasyon otoritesi ABD'nin FAA (Federal Aviation Administration) kuruluşudur [1,2].

3. Yapısal Kapsama Analizi (Structural Coverage Analysis - SCA)

Geliştirilen bir yazılımın doğrulanmasında kullanılan en geçerli ve bilinen yöntem "Gereksinim Tabanlı Test" prosedürleri geliştirilmesi ve bu testlerin koşturulması ile gereksinimlerin sağlanıp sağlanmadığının gözlenmesidir. Eğer tüm testler başarılı olarak tamamlanmışsa, bu durumda yazılım, gereksinimler odaklı doğrulanmış demektir. Fakat özellikle güvenlik-kritik yazılımlarda bu yeterli görülmemektedir. Tüm gereksinimlerin doğrulanmasının yanı sıra, yazılımın istenilen dışında bir davranış göstermemesi de zorunludur. Bu yüzden başka analiz yöntemleri gerekmektedir.

Yapısal Kapsama Analizi, gereksinim tabanlı test prosedürlerinin var olan kodun yapısının ne kadarını çalıştırdığının bir ölçütüdür. Bu analiz ile, istemsiz olarak yazılımın içerisinde yer almış kod parçacıklarını belirlemek olasıdır. Aynı zamanda, bu analiz hazırlanmış olan "Gereksinim Tabanlı Test" prosedürlerinin yetkinliği hakkında da bilgi sahibi olunmasını sağlar.

YKA'nın getirileri aşağıdaki şekilde özetlenebilir [3,6]:

- Gereksinimlerde belirtilen işlevsellik dışında, gerekmeyen kod parçalarının bulunması – Eğer gereksinim tabanlı testler istenilen seviyeye göre yeterli olarak yazılmışsa, bu testlerin tümünün başarıyla tamamlanmasına rağmen kapsanmamış kod parçaları gereksiz olabilir.
- Gereksinim tabanlı testlerin yeterliliğinin ve kalitesinin doğrulanması – Yapılan analizler sonucunda kapsanmamış olan kodun aslında fazla olmadığı, fakat yapılmış olan gereksinim tabanlı testlerin eksik olduğu anlaşılabilir. Böylece testler tamamlanarak ve/veya düzeltilerek yazılımın daha iyi test edilmesi sağlanır.
- Eksik gereksinimlerin ortaya çıkarılması – Bazı durumlarda kodu geliştiren kimselerin alan bilgisi de oldukça yüksek olabilir ve gereksinimlerde yazmamasına rağmen bazı kod parçalarının daha yazılımda yer alması gerektiğini bilerek kodlayabilirler. Bu durumda yapılan analizlerde kapsanmamış olan bu koda ait gereksinim yazılması gerekliliği ortaya çıkabilir.

Güvenlik-kritik yazılımlar çalışacakları yer ve yapacakları işlere göre farklı seviyelere ayrılırlar. Yapısal Kapsama Analizi belirlenen kritiklik seviyesine göre farklı tiplerde uygulanır. Aşağıdaki analiz tipleri daha az kritik seviyelerden daha kritik seviyelere doğru dizilmiştir.

3.1. İfade Kapsama (Statement Coverage):

İfade Kapsama sağlayabilmek için, programda bulunan çalışabilir her satır kodun gereksinim tabanlı testler sırasında en azından bir kez çalıştırılmış olması gerekmektedir. Bu durum bize tüm kod satırlarının çalışma sırasında ulaşılabilir olduğunu gösterir. İfade kapsama zayıf bir kriterdir, çünkü bazı kontrol yapılarına duyarsızdır. Aşağıdaki örnekte eğer x = 2, y = 0 ve z = 4 girdileri ile test yapılırsa bu ifadeler kapsanmış olur. Fakat ilk ifadenin kodlaması sırasında ***and*** yerine ***or*** yazılmışsa bunu test etmek olası değildir.

```
if (x > 1) and (y = 0) then
      z := z / x;
   end if;

if (z = 2) and (y > 1) then
      z := z + 1;
   end if;
```

3.2. Karar Kapsama (Decision Coverage):

Karar Kapsama, bir kontrol yapısının sonucunun hem "Doğru" hem de "Yanlış" olduğu durumları analiz eder. Gereksinim tabanlı testler ile bir "karar"ın hem "Doğru" hem de "Yanlış" akışının test edilmesi ve çalıştırılması gerekmektedir. Basit "karar"lar için bu yöntem tüm kontrol yapısının tamamıyla analizini sağlasa da kompleks karar yapıları için bu geçerli olmayabilir. Aşağıdaki örnekte (*A-B*) girdileri için (*Doğru-Yanlış*) ve (*Yanlış-Yanlış*) değerleri sağlandığında "karar" kapsanmış olur. Fakat B parametresinin programa etkisi test edilmemiş durumdadır.

```
if (A or B) then
   ...
   end if;
```

Örnek 1

3.3. Koşul Kapsama (Condition Coverage):

Koşul Kapsama bir "karar" içindeki tüm "koşul"ların en az bir kez tüm olası değerleri almasını gerektirir. Fakat bu "karar"ın tüm sonuçlarının en az bir kez gerçekleşmiş olması gerektiği anlamını taşımaz. Örnek 1'de (*A-B*) girdileri için (*Doğru-Yanlış*) ve (*Yanlış-Doğru*) değerleri sağlandığında "koşul"lar kapsanmış olur. Fakat "karar" olası tüm değerlere, yani "Yanlış" sonucuna ulaşmamıştır. Bu durumda else ile başlayan ifadeler test edilmemiş demektir.

3.4. Koşul/Karar Kapsama (Condition/Decision Coverage):

Koşul/Karar Kapsama "koşul kapsama" ve "karar kapsama"nın gereksinimlerini birleştirir. Yani hem tüm olası "koşul" kombinasyonları sağlanmalı ve aynı zamanda da "karar" tüm olası değerleri almalıdır. . Örnek 1'de (*A-B*) girdileri için (*Doğru-Doğru*) ve (*Yanlış-Yanlış*) değerleri sağlandığında hem tüm "koşul"lar kapsanmış olur hem de "karar" olası tüm değerlere ulaştığından "karar kapsama" sağlanır. Buradaki tehlike ise ***or*** yerine ***and*** yazılırsa testler bunu yakalayamamış olur.

3.5. Değiştirilmiş Koşul/Karar Kapsama (Modified Condition/Decision Coverage, MC/DC):

"Koşul/Karar Kapsama"nın geliştirilmiş bir türüdür. Her "koşul"un bağımsız olarak "karar"ın sonucunu etkilemesi gerekliliği ana esastır. Bağımsız etkileme gereksinimi, her bir koşulun diğer koşullara göreceli olarak test edilmesini sağlar. . Örnek 1'de (*A-B*) girdileri için (*Doğru-Yanlış*), (*Yanlış – Doğru*) ve (*Yanlış-Yanlış*) değerleri sağlandığında hem tüm "koşul"lar kapsanmış olur hem de "karar" olası tüm değerlere ulaştığından "karar kapsama" sağlanır. "Koşul/Karar Kapsama" da bahsedilen tehlike ise artık burada yoktur [4].
Bu yöntem şimdiye kadar anlatılan tüm diğer yöntemlerden daha fazla test gerektirmektedir.

3.6. Çoğul Koşul Kapsama (Multiple Condition Coverage):

Çoğul Koşul Kapsama, bir "karar"ı etkileyen "koşul"ların tüm olasılıklarının test edilmesi anlamına gelir. Pratikte hızla büyüyen bir test seti gerektirir [5]. Eğer bir "karar" içerisinde *n* "koşul" varsa 2^n test yapılması anlamına gelir. . Örnek 1'deki gibi bir kodu test etmek için 128 adet test gerekir ki neredeyse sadece bir karar için bile pratik olmaktan uzaktır.

4. Araç Desteği

Kapsama verisini elde etmek için bir yapısal kapsama analiz aracı kullanılır. AYESAŞ'da bu amaçla Rational TestMate MC/DC ve Vector CAST/Cover araçları kullanılmaktadır. Bu araçlar, projenin gereği olarak ifade, karar ya da MC/DC (Değiştirilmiş Koşul/Karar) seviyesinde kapsama analizi yapabilmektedirler. YKA araçları, kaynak kod içerisine ek kod parçaları ilave ederler (enstrümantasyon), ve bu değiştirilmiş kod kullanılarak gereksinim tabanlı testler yapıldığında kapsama verisinin elde edilmesini sağlarlar. Bu kapsama verisi, yapısal kapsama analiz araçları tarafından analiz edilerek içinde kapsanmamış kod satırlarının işaretlendiği kapsama analizi çıktı dosyalarından oluşur.

AYESAŞ'da YKA DO-178B standardının seviyelerine göre uygulanmaktadır. Bu seviyeler Tablo 1'de gösterilmiştir.

Tablo 1. DO-178B Seviyeleri

DO-178B Seviyesi	Kapsamı	Analiz Seviyesi
A	MC/DC + İfadeler	Koşullar
B	Kararlar + İfadeler	Kararlar
C	İfadeler	İfadeler

AYESAŞ bünyesinde doğrulama sürecinde yazılım ürünlerine ait raporların standart bir şekilde üretilmesi, saklanması ve durum takibi amacıyla Kapsama Analizi Veritabanı (Coverage Analysis Database, CAD) adında bir sistem oluşturulmuştur. Kapsama Analizi Veritabanı portalı, yapısal kapsama analizi sonuçlarının saklandığı Kapsama Analizi Veritabanı ve kullanıcı erişimini sağlayan bir Web sunucusundan oluşmaktadır. Intranet ortamında çalışan bu sistemde kullanıcılar ancak

görevlendirildikleri projelere erişim hakkına sahiptir. Kapsama Analizi Veritabanı kullanarak doğrulama sürecinde üretilen yazılım ürünlerine ait;

- Test Sonuç Raporu,
- Yapısal Kapsama Analizi Raporu,
- Gözden Geçirme Raporu ve
- Yapısal Kapsama Analizi Özet Tabloları (SVTPR)

oluşturulabilir. Kapsama Analizi Veritabanı ayrıca bu raporlara ait durum bilgisini listeleyerek durum takibini yapmaya da olanak sağlamaktadır.

Test sonuç raporu yapısal kapsama analizi için koşturulan her bir gereksinim tabanlı test için doldurulur. Bu rapor, testi yapan kişi, test tarihi, yazılım test dokümanı versiyonu gibi alanları ve test sonuç kontrol listesini içerir. Test sonuçları ve test sonuç raporu ilgili gözden geçirme toplantısında incelenerek test sonucu doğrulanır.

YKA raporu her bir kaynak kod dosyası için doldurulur. Bu raporun doldurulabilmesi için testler sonucu kaynak kod dosyasının tamamının kapsanmış olması ya da kaynak kod dosyasına ait tüm testler tamamlanmışken, kapsanmamış kod parçalarının tamamının açıklanabiliyor olması gerekmektedir. YKA raporu;

- Kaynak kod dosyasının adı, kapsama analizi çıktı dosyasının adı, versiyonu, kapsama analizi seviyesi gibi alanlarla beraber,
- Varsa, kapsanmamış kod parçalarına ait problemlerin analiz sonuçlarını,
- Kaynak kod dosyasına ait kapsama analizi çıktı dosyası verileri kullanılarak oluşturulan kapsama analizi metrik tablosunu,
- Test-kaynak kod dosyası izlenebilirlik tablosu kullanılarak oluşturulmuş ilgili testlere ait durum listesini,
- YKA sonrasında, yapılan aktivitelerin kontrol edildiği Yapısal Kapsama Analizi Kontrol Listesini

içermektedir. Kapsama analizi çıktı dosyası ve Yapısal Kapsama Analiz raporu ilgili gözden geçirme toplantısında incelenerek kaynak kod dosyası doğrulanır.

Gözden geçirme raporu yazılım ürünleri için yapılan gözden geçirme toplantıları öncesinde doldurulur. Gözden geçirme toplantıları Yazılım Kalite Mühendisi'nin koordinasyonu ile düzenlenir. Gözden geçirme raporunda, gözden geçirme toplantısının türü, tarihi, incelenecek yazılım ürünlerinin listesi ve katılımcıların listesi gibi alanlar bulunur. Bu toplantıda problem ile karşılaşılmaması durumunda kalite mühendisinin onayıyla, yazılım ürünlerinin resmi olarak kullanılabileceği ilan edilir.

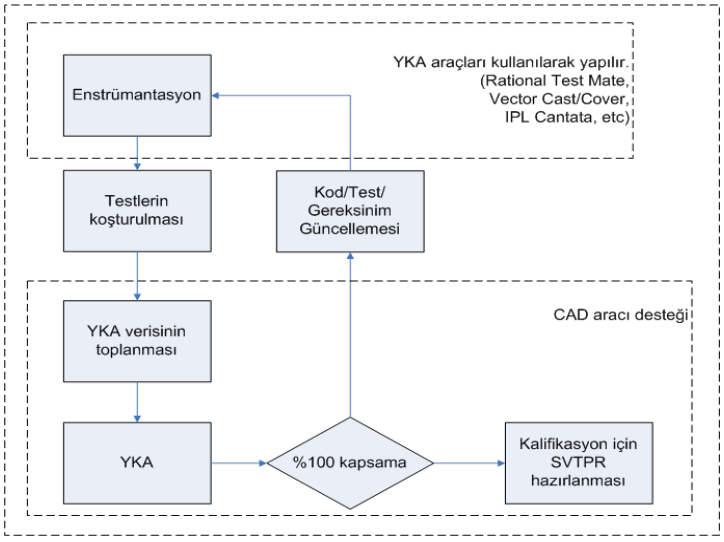
Böylece süreçteki yazılım ürünleri doğrulanmış olur.

SVTPR tüm testler ve kaynak kod dosyaları doğrulandıktan sonra Kapsama Analizi Veritabanı kullanılarak üretilen bir sonuç tablosudur. Bu tabloda her bir kaynak kod dosyasına ait kapsama analizi metrikleri, hangi testlerin sonucunda kapsandığı, kapsanmama sebebi ve kapsanma yüzdesi bilgileri bulunur. SVTPR farklı bölümlerde bulunan farklı kapsama seviyeleri için ayrı ayrı üretilir.

Doğrulama süreci sonunda, üretilen SVTPR, test sonuçları, test sonuç raporları, kapsama analizi çıktı dosyaları, yapısal kapsama analizi raporları sertifikasyon için hazır hale gelir.

5. AYESAŞ'da YKA

AYESAŞ tarafından yürütülen yapısal kapaman analizinin iş akışı şematik olarak Şekil 1'de verilmiştir. Bu aktivite, kodun hangi yapılarının kapsanamadığını bulmaya yarar.



Şekil 1. YKA akış şeması

5.1. Analize Giriş

Bu aktiviteye başlamak için gereken girdiler şunlardır:

- SRS (Yazılım Gereksinimleri)
- SDD (Yazılım Tasarımı)
- STD (Yazılım Test Dokümanı)
- STR (Yazılım Test Sonuçları)
- Kaynak kodu
- İzlenebilirlik verisi (Gereksinim-Kod izlenebilirliği)

5.2. Aktiviteler

5.2.1. YKA Verisinin Toplanması için Gereksinim Tabanlı Testlerin Koşturulması

YKA verisinin toplanması için, testlerin enstrümente edilmiş kod üzerinde koşması gerekir. Bu enstrümantasyon işlemi için yardımcı araçlar kullanılır (Rational TestMate, Vector CAST/Cover, IPL Cantata, vb.). Testler gereksinin tabanlı olup, STD dokümanında anlatılır. Testlerin ve sonuçlarının, analize devam edebilmek için, dokümente edilmesi ve doğrulanması gerekir. Ayrıca, Gözden Geçirme sürecine uygun olarak, test sonuçları incelenerek doğrulanır.

5.2.2. YKA Aşaması

Bu aktivite, gereksinimlere uygun testlerin, kodun tümünü çalıştırıp çalıştırmadığını bulmak için yapılır. Eğer %100 kapsama sağlanan kod birimleri var ise, bu birimler üstünde bir analiz yapılmaz. Kapsanamamış yerler için bu analize başlanır. Kapsanamama nedenleri şunlar olabilir:

- Gereksinimlerin ve/veya testlerin yetersizliği: Bu durumda gereksinim veya testler değiştirilir ve aktiviteye baştan başlanır.
- Ölü (dead), pasif (de-activated), koruyucu (defensive), kullanılmayan (unused) kod, vs.

5.2.3. YKA Sonuçlarının Doğrulanması

YKA sonuçları elde edildikten sonra, her kod modülü için, Gözden Geçirme sürecine uygun olarak doğrulama yapılır.

5.2.4. Analiz Çıktıları

YKA'nın çıktıları şunlardır:

- Yapısal Kapsama Analizi Raporu
- Yapısal Kapsama Analizi Kontrol Listeleri
- Kapsama verisi
- Gereksinim tabanlı ilave testler
- Yazılım değişiklik istekleri (kod, gereksinim, vs.)
- Test Sonuç Raporları

Tablo 2 ve 3'te örnek bir YKA sonuç raporu görmekteyiz. Tabloda, koşul sayısı, YKA araçları ile kapsanan koşul sayısı, araç kullanmadan yapılan analiz ile kapsanan koşullar, kapsanamayan koşullar ve bunun nedeni yazılmıştır.

Toplam Koşul Sayısı	YKA Aracı ile Kapsananlar	Geçmiş Kapsama Değeri	Manuel Analiz ile Kapsananlar	YKA'de Bulunan Kod Yetersizlikleri (inadequacies)
41	18	15	7	1

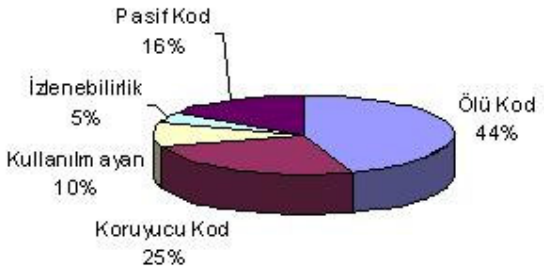
Tablo 2. YKA Sonuç Raporu

Sorun No	AYESAS Kod Değişim İsteği No	Müşteri Kod Değişim İsteği No	Kapsanamama Nedeni	Toplam Koşul Sayısı
368	-	-	Manuel Analiz	7
369	-	-	Pasif Kod	1

Tablo 3. YKA Analiz ve Yetersizlik Raporu

5.3 AYESAŞ'ta Elde Edilen Çaba Göstericileri

Ayesaş bünyesinde, birden çok projede YKA yapılmakta ve proje sonunda elde edilen göstericiler kaydedilmektedir. Şekil 3'te Seviye A bir projeden elde edilen sonuçlar verilmektedir.



Şekil 3. Seviye A örnek göstericileri

Verilen örnekten de anlıyacağımız gibi YKA sonucunda sıklıkla rastlanılan sorun ölü ve koruyucu koddur. Böyle kod türlerinin yazılımda sık bulunması, YKA sürecini uzatacak ve zorlaştıracaktır. Bu nedenle yazılım geliştirilme aşamasında böyle kodların yazılımdan mümkün olduğunca ayıklanması, doğrulama sürecine katkıda bulunur. Şekil 4'te ise bu projeden elde edilen çaba göstericileri verilmektedir.

Şekil 4. Seviye A örnek çaba göstericileri

Örnek çaba değerlerinden de anlaşılacağı gibi, YKA'ya harcanan zaman bütün proje içerisinde büyük bir paya sahiptir. Bu nedenle süreçlerin doğru uygulanması projeler için kritik olmaktadır.

6. Sonuç

Yazılım ürünlerinin hayatımızda her geçen gün daha fazla yer aldığı bir gerçektir. Özellikle güvenlik-kritik yazılım geliştirme ve doğrulama süreci, ülkemizde nispeten az uygulanan

süreçlerden biridir. Bu sürecin önemli bölümlerinden biri olan Yapısal Kapsama Analizi genellikle yaşam döngüsünün son adımında uygulanmakta ve tüm ürünlerin (gereksinimler, tasarım, kaynak kod, nesne kodu, test senaryoları ve izlenebilirlik verileri) uyumluluğunu garanti altına almaktadır.

Uygulanabilmesi için yeterli araç ve yöntem desteği gereken bu süreç ile ilgili AYESAŞ'ın mevcut deneyim ve uygulamaları bu bildiride özetlenmiştir. Daha detaylı bilgi için info@ayesas.com adresinden bilgi talep edilebilir.

Kaynakça

[1]. RTCA DO-178B, "Software Considerations in Airborne Systems and Equipment Certification", 1992

[2]. EUROCAE/ED-12B, "Software Considerations in Airborne Systems and Equipment Certification", 1992

[3]. Beizer, Boris, "Software Testing Techniques", 2nd edition, New York: Van Nostrand Reinhold, 1990

[4]. John Joseph Chilenski and Steven P. Miller, "Applicability of Modified Condition/Decision Coverage to Software Testing", Software Engineering Journal, September 1994.

[5]. Ntafos, Simeon, "A Comparison of Some Structural Testing Strategies", *IEEE Trans. Software Eng.*, Vol.14, No.6, June 1988.

[6]. Roper, Marc, "Software Testing", London, McGraw-Hill Book Company, 1994