Yazılım Geliştirmede Uygunsuzluk Önleyici bir Güç Olarak Yazılım Kalite Güvencesi

Ahmet Gökhan Aktürk
MilSOFT Yazılım Teknolojileri
A.Ş, Ankara
aakturk@milsoft.com.tr

Özetçe

Yazılım kalite güvencesi, yazılım ürün ve süreçlerinin yazılım geliştirme projesi yaşam döngüsü boyunca tanımlı gereklere uygunluğuna dair güvence vermek amacıyla planlanan, uygulamaya konan ve uygulanan faaliyetler bütünüdür[1]. Burada güvence vermeyi bir çok farklı şekilde yorumlamak ve uygulamak mümkün olmasına karşın, genel yaklaşım kalite güvenceyi ürün geliştirmede uyulması beklenen gereksinimlere uygunluğun ilgili faaliyetin tamamlanması sonrasında doğrulanması olarak kabul etmek şeklindedir. Bu yaklaşıma uygun bir kalite güvence mevcut problemleri ortaya koyma ve bunların giderilmesi için bir girişim sağlamanın ötesine gitmemektedir. Oysa kalite güvencenin gereksinimlere uygunluk açısından geçmiş performansı değerlendirmenin ötesine geçerek, gelecekteki performansla ilgili fikir sunacak şekilde kurgulanması durumunda, olası problemlere dair geri besleme sağlanabilir, bu veriye dayanarak alınacak önleyici işlemlerin sonucunda kalite güvencenin çok daha etkin ve fayda – maliyet oranı çok daha yüksek olması söz konusu olabilir. Burada, bir firmada önleyici yaklaşımları temel alarak tasarlanan, uygulamaya konan ve etkinliği takip edilen bir kalite güvence altyapısı anlatılmıştır

1. Giriş

İlk yazılım ürünlerinin üretildiği günlerden bugüne, yazılım ile kastedilen kavram sürekli bir gelişim içinde olmuştur. Kısa süre öncesine kadar sadece zaman alan veri işleme operasyonlarının otomasyonu gibi temel işlevler için kullanılan yazılım ürünlerinin yerini bugün çok daha gelişmiş teknik altyapılar kullanan ve çok daha karmaşık işlemleri yerine getirebilen yazılımlar almıştır. Günümüzde yazılım artık sadece yoğun olarak bilgisayar kullanımı gerektiren sektörlerde kullanılmaktan çıkmış, otomobillerden ev aletlerine, telekomünikasyon cihazlarından oyuncaklara kadar uzanan bir yelpazede kullanım alanı bulmuş, çoğu zaman bir sistemin parçası olarak sunulur olmuştur. Ortaya çıkan bu hızlı gelişme yazılım geliştirmenin artan karmaşıklığını yönetebilmek için arayışlar doğmasına neden olmuştur.

Bu arayışlar bir taraftan bazı standartların yazılım geliştirmede nasıl yorumlanabileceği sorusunu akıllara getirirken, öte yandan bu sektöre özgü yaklaşımlar için de girişimler başlamasına yol açmıştır. Bu arayışların sonunda yazılım geliştirme sektöründe genel kabul gören standart ve modeller arasında SEI CMMI modeli ile ISO 9001 ve NATO AQAP-160 standartları sayılabilir

Bu standart ve modeller, karmaşık ürün geliştirme faaliyetlerinin etkin biçimde yönetilebilmesini sağlamak için kalite güvenceyi gerekli araçlardan biri olarak göstermişlerdir. Kalite güvencenin uygulanmasına dair esasları farklı detaylandırsalar da özünde kalite güvenceyi ilgili faaliyetlerin ve bu faaliyetler sonucu ortaya çıkan ürünlerin tanımlı gereksinimlere uygun biçimde gerçekleştirildiğine dair bağımsız ve objektif bir güvence sağlamak olarak tanımlamışlardır[2,3,4].

Bu tanıma uygun olarak, genel kabul gören yazılım kalite güvence uygulaması, yazılım geliştirme faaliyetlerinin ve yazılım ürünlerinin gereksinimlere uygunluğunu düzeltici bir bakış açısıyla ele almak şeklindedir. Yazılım geliştirme faaliyetlerinin tamamlanmasını takip eden kontroller ile bu faaliyetlerin gereksinimlere uygunluğunun değerlendirilmesi, yazılım ürününün testlere ve incelemelere tabi tutularak ürünün gereksinimlere uygunluğuna dair güvence sağlanması benimsenen uygulama biçimleri olarak verilebilir.

Yazılım kalite güvencesinin, yukarıda belirtildiği gibi, ilgili faaliyetin tamamlanması veya ürünün ortaya çıkmasından sonra uygulanacak bir faaliyet olarak tasarlanması durumunda, getirisi gereksinimlere uygun olmayan durumların saptanması ve bunların giderilmesi için bir yol sağlaması olarak belirtilebilir. Ancak bu durumda, saptanan uygunsuzlukların giderilmesinin maliyeti etkilenen faaliyet ve / veya üründeki uygunsuzluğun boyutuyla orantılı olarak büyüyebilmektedir.

Buna karşın, kalite güvencenin altyapısı uygun olmayan durumları oluşmadan saptayacak şekilde tasarlandığı takdirde, önleyici işlemlerin alınması ve böylelikle uygunsuzluğun önlenerek gereksinimlere uyum sağlanması mümkün olacaktır.

MilSOFT'ta geçmişte uygulanan kalite güvence yaklaşımlarının çoğunlukla dayandığı nokta, yazılım geliştirme faaliyetlerinin çeşitli kalite güvence faaliyetleri – ki bunlar arasında denetimler, gözden geçirmeler, kontroller sayılabilir – vasıtasıyla değerlendirilmesi ile yazılım ürünlerinin üretildikten sonra teste tabi tutulmaları şeklinde olmuştur. Bu yaklaşım yazılım kalitesini sağlamakta pozitif sonuçlar verdiyse de, saptanan sorunların giderilmesi maliyetinin yüksek olabilmesi, MilSOFT'u arayışlara itmiştir. Bu bağlamda, MilSOFT uygunsuzlukları önleyici bir metodoloji için harekete geçmiştir.

MilSOFT'un tasarlayıp uygulamaya koyduğu metodolojinin temelinde, uygunsuzlukların ortaya çıkmadan saptanabilmesini sağlamak bulunmaktadır. Bu metodoloji; daha önce uygulanmakta olan kalite güvence yaklaşımı ile MilSOFT'un benimsemiş olduğu modeller ve standartlardaki kalite güvence tanımları da biraraya getirilerek oluşturulmuştur. Bu amaçla kullanımda olan sistem incelenmiş, model ve standartların beklentileri ve gereksinimleri analiz edilmiş, geçmiş dönemlerde kalite güvence uygulamaları ile ilgili geri beslemeler dikkate alınmıştır.

2. bölümde genelde kabul gören düzeltme odaklı kalite güvence anlayışına değinilmektedir. Söz konusu yaklaşımın yerine önerilen önleme odaklı anlayış ve bileşenleri ise bir sonraki bölümde ele alınmaktadır. 4. bölümde, önerilen kalite güvence metodolojisinin MilSOFT'ta hayata geçirilmesine dair sonuçların değerlendirilmesi için tanımlanan kriterler verilmektedir. Önerilen metodolojinin

risklerine 5. bölümde değinilmekte, yaklaşıma dair tartışmalar ise elde edilen sonuçlar ve riskler ışığında 6. bölümde verilmektedir.

2. Bir Düzeltme Aracı Olarak Kalite Güvence ve Sorunları

Günümüzde üretilen yazılım ürünleri, on yıl öncesinde üretilen yazılımlara kıyasla çok daha karmaşık ve büyük boyuttadır. Buna verilebilecek en çarpıcı örneklerden biri, Microsoft'un Windows NT işletim sisteminin 3.1 versiyonunun 1993'te piyasaya sürüldüğünde 4 ila 5 milyon satır koddan oluşurken, bundan sadece 10 yıl sonra 2003'te piyasaya çıkan Windows Server 2003'ün yaklaşık 50 milyon satır kod içermesidir[5]. Böylesi karmaşık ürünlerin beklenen gereksinimleri karşılaması için yazılım geliştirmenin belli kurallar çerçevesinde yapılması zaruri hale gelmiştir. Bu nedenle, yazılım geliştirmede genel geçer standartlar ve yazılıma hitap eden modellerin kullanımı artık kabul gören bir yaklaşımdır.

Bu kapsamda, bu standart ve modellerde tarif edilen bir fonksiyon olan kalite güvence de, gereksinimlerin karşılandığına dair güvence vermek amacıyla kullanılmaktadır. Bu kullanım genelde yazılım geliştirme faaliyetlerinin tanımlanan şekilde yürütülüp yürütülmediğinin kontrol edilerek, sapma belirlenmesi durumunda düzeltme amaçlı işlem başlatma şeklinde olmaktadır.

Kalite güvencenin yazılımdaki yeri altbölüm 2.1'de, genel kabul gören yazılım kalite güvencesi yaklaşımı altbölüm 2.2'de ve bu yaklaşımla ilgili muhtemel sorunlar altbölüm 2.3'te detaylandırılmıştır.

2.1. Kalite Güvencenin Yazılımda Yeri

Bir yazılım ürününü bir çok diğer üründen ayıran özelliklerden biri, yazılımın belirlenen gereksinimleri karşılamak amacıyla özel bir çözüm şeklinde tasarlanarak üretilmesidir; her yazılımı tanımlayan gereksinimler farklı olduğundan ortaya konan ürün de seri üretim ürünlerin aksine hemen her zaman eşsiz olmaktadır. Kullanılan teknolojilerin ve tasarımda izlenen yaklaşımların değişkenliği de bu etkiyi arttırmaktadır. Bu nedenle yazılım geliştirme faaliyetinin üretim ile hizmet sağlama arasındaki sınırda olduğu ve aynı zamanda hem ürün, hem hizmet olarak tanımlanabileceği söylenebilir[6].

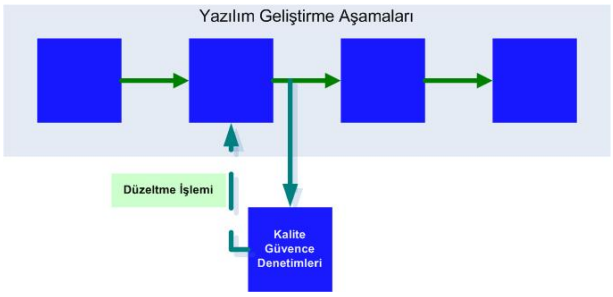
Yazılımın seri üretime bir üründen farklılıklar göstermesi ve bazı açılardan hizmet olarak görülebilmesi nedeniyle, kalite güvencenin de bu farklılıklara cevap verecek yaklaşımlar getirmesi gerekmektedir.

Yazılım geliştirmenin bu özel durumu nedeniyle, yazılım kalite güvencesinde ürün odaklı yaklaşımlar kadar, süreç temelli yaklaşımlar da kullanılmakta; ürünün gereksinimlere uygunluğunu güvence altına almak için tasarım ve üretim aşaması süreçlerinin tanımlandıkları şekilde uygulanmalarının takibi öne çıkan bir yaklaşım olmaktadır.

Yazılım kalite güvencesinde yaygın olarak kullanılan yöntemlerden biri, standartlara, gereksinimlere ve iş süreçlerine uygunluğun denetimlerle belirlenmesidir. Bu yöntemde, müşteri gereksinimleri, yasal gereksinimler ve organizasyonun standart ve modellerin analizi ile oluşturduğu dahili gereksinimleri temel alınır ve yazılım geliştirme faaliyetlerinin bu gereksinimlerde tanımlı biçimde yürütülüp yürütülmediğini saptamak amacıyla bağımsız denetçiler planlı veya plansız denetimler gerçekleştirirler.

2.2. Düzeltme Amaçlı Kalite Güvence

Eğer kalite güvence denetimlerinde benimsenen yaklaşım, uygunsuzlukların giderilmesi ise, yöntem projedeki uygulamaların tanımlandıkları şekilden farklılık belirlendiğinde bu durumun kayıt altına alınarak düzeltilmeleri için işlem alınması şeklindedir. Burada amaç, tüm yazılım geliştirme boyunca faaliyetlerin gereksinimlerle uyumlu yürütülmesini sağlayarak üretilecek yazılımın da gereksinimlere uygun olmasına dair güvence sağlamaktır.



Şekil 1 Düzeltme Odaklı Kalite Güvence Yaklaşımı

Kalite güvencede amaç, uygunsuzlukları saptamak ve düzeltilmelerini sağlamaksa, denetim mekanizmasının temelinde uygunsuzlukların etkin saptanması bulunacak; denetimlerin genel işleyişi de, olan ile olması gerekeni kıyaslamak ve farkları ortaya koymak şeklinde olacaktır. Bu işleyişte akışın şu adımlardan oluştuğu söylenebilir:

1. Tanımlı gereksinimlerin analiz edilmesi,
2. Denetimin yapıldığı nokta itibariyle gereksinimlerin belirlenmesi,
3. Uygulama sonuçlarının (çıktılarının) incelenmesi,
4. Çıktılar ile beklenenin kıyaslanması,
5. Uygunsuzlukların (sapmaların) belirlenmesi.

Şekil 2 Düzeltme Odaklı Kalite Güvencede Denetim

Şekil 2'de genel çerçevesi verilen bu işleyişin tamamlanmasından sonra, uygunsuzlukların düzeltilmesi için girişim ayrı bir akışla olacaktır. Bu akışın amacı, uygunsuzluğun ortadan kaldırılarak uygulamanın gereksinimleri karşılamasıdır. Bunun için de, aşağıdaki adımların kullanılması mümkündür:

1. Uygunsuzlukların dokümante edilmesi,
2. Uygunsuzlukların düzeltilmesine yönelik işlem planı oluşturulması,
3. Düzeltme amaçlı işlem alınması,
4. İşlemin takibi ve başarısının değerlendirilmesi.

2.3. Yaklaşımın Kısıtları

Yazılım geliştirme faaliyet ve ürünlerinde, bu faaliyetlerin tamamlanması ve ürünlerin ortaya çıkmasından sonra gerçekleştirilen denetimler, yukarıda da belirtildiği gibi, gereksinimlerden sapmaları belirlemekte kullanılabilir; bu sapmaların uygunsuzluk olduğu kanısı olduğu takdirde giderilmeleri ve ortaya gereksinimleri karşılayan bir ürün çıkması sağlanabilir. Bu yaklaşım, kalite güvencenin esas amacı olan gereksinimlere uygunluğa dair güvence sağladığı için, başarılı ve etkin bir kalite güvence uygulaması olarak değerlendirilebilir.

Ancak, bu yaklaşımı uygularken, bu başarının maliyetini gözardı etmemek gerekir. Yazılım geliştirme faaliyetlerinin denetimleri ve saptanan uygunsuzlukların giderilmesi için harcanan işgücü, bu maliyete dahildir. Uygunsuzluğun alınacak işlemlerle düzeltilmesi

mümkünse, bu işlemler için harcanacak işgücü, uygunsuzluğun gerçekleşmeden önce saptanarak önlenmesi için gerektiği tahmin edilen işgücüne göre fazla olabilecektir. Bu takdirde, gerçekleştirilen kalite güvence faaliyetinin yeterli verimi sağlamadığı ifade edilebilir.

Bazı durumlarda ise, saptanan uygunsuzlukların giderilmesi, ilgili faaliyet tamamlandıktan sonra o faaliyete geri dönüşün ya mümkün olmaması, ya da çok maliyetli olması nedeniyle söz konusu olamayabilir. Örneğin, performansla ilgili gereksinimlerin söz konusu olduğunda, tasarımın ilgili gereksinimleri karşılayacak şekilde yapılmaması, ve sonraki aşamalarda bu tasarımın temel alınması durumunda, ürünün baştan tasarlanması mümkün olamayabilir.

Düzeltme odaklı yaklaşımda, böyle bir durumla karşılaşıldığında; ya ilgili faaliyetin tekrarlanması maliyeti kabul edilerek uygunsuzluk giderilebilir, ya da uygunsuzluk kabul edilerek ilgili riskler üstlenilebilir. Ancak hem pazara sunmak amacıyla yazılım geliştiren, hem belirli bir müşterinin gereksinimlerini karşılayan özelleşmiş bir çözüm sunan organizasyonlarda zaman kısıtının etkisi yok sayılamayacağından[7], bu uygulanabilir bir yaklaşım olmayabilir.

Bu kısıtlar, klasik kalite güvence yaklaşımlarının uygulanmasının, süreç polisliği yapmaya benzer biçimde görülebileceğine dair birer işaret oluşturmaktadır[8].

2.4. Yaklaşıma Dair MilSOFT'un Deneyimleri

Hem ana faaliyet alanı yazılım olması, hem de kalite güvenceyi ve bu fonksiyonu beraberinde getiren standart ve modelleri benimseyen bir yaklaşıma sahip olması açısından MilSOFT, yazılım kalite güvenceyi kuruluşundan beri etkin biçimde uygulamayı hedef olarak almıştır.

MilSOFT'un yakın bir döneme dek benimsediği anlayış, denetim ve diğer aktivitelerle gereksinimlere uygunluğu değerlendirerek, uygunsuzlukların düzeltilmesi için bir yöntem sunmak şeklinde tanımlanmıştır. MilSOFT'un kalite güvence sistemi, bu dönemde şu adımlara cevap verir biçimde yapılanmıştır:

1. Yasal, organizasyonel ve sözleşmesel gereksinimlerin analizi ile gereksinimlerin uygulanması gereken kapsamın belirlenmesi,
2. Denetimlerle gereksinimlere uygunsuzluğun belirlenmesi,
3. Uygunsuzlukların kritiklik açısından değerlendirilerek derecelendirilmesi,
4. Uygunsuzlukların raporlanması,
5. Düzeltme faaliyetinin planlanması ve takvimlendirilmesi,
6. Düzeltme faaliyetinin kontrol edilmesi.

Bu yaklaşım ile MilSOFT, uygunsuzlukların saptanması ve giderilmesi için etkin bir yapı kurmuş, uygunsuzluklar müşteriye yansımadan ve benimsenen standart ve modellere göre gerçekleştirilen değerlendirmelerde saptanacak noktaya ulaşmadan düzeltilebilmiştir. Bu açıdan, MilSOFT'un önceki yaklaşımı etkin olarak tanımlanabilir. Ancak 2.3'teki kısıtlar çeşitli derecelerde MilSOFT için de geçerlidir.

Maliyet açısından bakıldığında, kalite güvence faaliyetleri sonucu saptanan uygunsuzlukların düzeltilmesinin maliyetinin toplam proje maliyetine oranının yazılımın karmaşıklığı arttıkça üstel olarak arttığı gözlenmiştir. Bu durumda, giderek daha kapsamlı gereksinimlere uygun projeler üstlenen MilSOFT için uygunsuzlukların düzeltilmesi maliyetinin de giderek daha yüksek olacağı öngörülebilir.

Zaman açısından incelendiğinde de, rekabet baskısının yoğun olduğu savunma sektöründe, göreceli olarak agresif takvimlere göre çalışıldığı söylenebileceğinden, uygunsuzlukların, gereksinimleri en iyi şekilde karşılayabilecek biçimde düzeltilmesi takvim baskısı ile mümkün olamayabilir. Böyle bir durum, savunma sanayiinin diğer bir çok sektöre oranla hataya toleransının az olması nedeniyle kabul edilemeyebilir. Gereksinimleri tam olarak karşılayamamanın, MilSOFT'un üstlenmesi imkansız bir risk olduğu söylenebilir.

Bu kısıtlar, MilSOFT'u daha verimli ve hızlı bir yaklaşım aramaya itmiştir. Uygunsuzlukları düzeltme maliyetinin yüksekliği, ve ilgili faaliyetin kimi zaman tekrar edilmesi gerekliliğinin getirdiği takvim riskinin kabul edilemez oluşu, bu arayışı desteklemektedir. 3. bölümde anlatılan yaklaşım, MilSOFT'un arayışlarının sonucunda tasarladığı altyapıyı tariflemektedir.

3. Önleme Aracı Olarak Kalite Güvence ve Bileşenleri

Uygunsuzlukların oluştuktan sonra düzeltilmesinin uygulamada yarattığı sorunlar ve belirtilen kısıtlar, gereksinimlere tam uygunluğun temel beklentilerden biri olduğu savunma sanayiinde düzeltme odaklı yaklaşımın yeterince verimli sonuçlar veremediğini göstermektedir.

Düzeltme odaklı yaklaşımın kısıtlarının ortadan kaldırılması ve bu yolla, uygunsuzluk maliyetlerinin azaltılması ve zaman baskısına bağlı risklerin giderilmesi, verimli bir kalite güvence fonksiyonunun hedefi olarak tanımlanabilir. Bu durumda, uygunsuzlukların gerçekleşmeden tespiti ve önlenmesi, ilk akla gelen çözüm olmaktadır.

Ancak, yaklaşımı oluştururken, iki faktör öne çıkmaktadır:

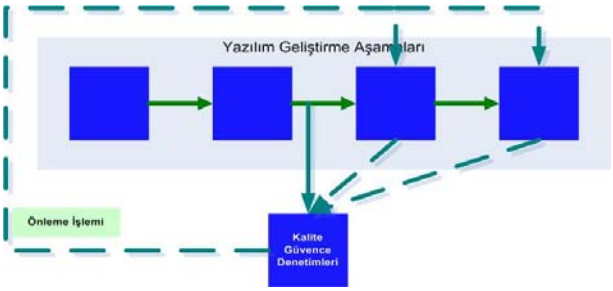
- Uygunsuzlukları oluşmadan belirleyen metodolojinin, düzeltici kalite güvence yaklaşımına benzer maliyet yapısına sahip, verimli tasarımlanması gerekmektedir.
- Uygunsuzlukları oluşmadan önleme maliyetinin, düzeltme maliyetine kıyasla düşük olması gerekmektedir.

O halde, önleyici kalite güvence, mevcut altyapının maliyetini ciddi biçimde arttırmamalı; önleyici işlemlerin erken alınmasını sağlayarak maliyetlerinin düşük kalmasını sağlamalıdır.

Bir sonraki altbölümde, MilSOFT'un tasarladığı yaklaşımın bileşenleri tariflenmektedir. Bu bileşenlerin biraraya getirilişi ve etkileşimi, 3.2'de verilmiştir. 3.3'te, düzeltme odaklı yapıdan önleme odaklı yapıya geçişte dikkat edilmesi gerekenler belirtilmektedir.

3.1. Gerekli Bileşenler

MilSOFT'ta tecrübelere dayanarak tasarlanan yaklaşım, temelde düzeltme odaklı yaklaşımın bileşenlerini kullanmaktadır. Bu yaklaşımda da araç olarak denetim vardır. Ancak denetim mekanizmasının hedefi, olası ve mevcut uygunsuzlukları belirlemek olarak değiştirilmiş, o yüzden girdi ve çıktılarında değişiklik olmuştur. Bu nedenle, kalite güvencenin yazılım geliştirme aşamaları ile ilişkisi de değişmekte, Şekil 3'teki gibi olmaktadır.



Şekil 3 Önleme Odaklı Kalite Güvence

Söz konusu yaklaşımı yerleştirebilmek için gereklerden biri, yazılım geliştirme faaliyetlerinin akışını, etkileşimlerini kavramak ve bu akışta uygunsuzlukların ortaya çıkmasına müsait noktaları anlayabilmektir. Bunun için, tamamlanmış yazılım geliştirme projesi uygulamaları ve uygunsuzlukları kaynak olarak kullanılmalıdır.

Bu noktada, geçmişte karşılaşılan uygunsuzlukların kaynağını, düzeltme yöntemlerini ve maliyetlerini analiz ederek uygulamada zayıf olunan faaliyetleri belirlemek ve kalite güvencenin verimliliğini arttırmanın mümkün olduğu ifade edilmelidir[9].

Geçmişte tamamlanan yazılım geliştirme projelerinin verilerini kullanırken, öncelikle üzerinde çalışılan projeye benzer projelerin seçilmesi, olası uygunsuzlukların saptanmasında daha güvenilir sonuçlar verebilecektir. Projelerin benzerlik derecesini değerlendirirken, projenin teknik içeriği – alanı, konsepti gibi – kadar, kullanılan yazılım mühendisliği teknikleri, ürün geliştirme yaşam döngüsü, ekibin yetkinlikleri de dikkate alınmalıdır.

Kalite güvencede amaç uygunsuzlukların önlenmesi ise, olası uygunsuzlukların en önemli ön göstergelerinden ikisi, yazılım geliştirme faaliyetleri ile ilgili ölçüm sonuçları ile tanımlanan riskler olarak ifade edilebilir. Yazılım geliştirme uygulamalarının tanımlı gereksinimlere uygun olmaları halinde bile, tanımlanmış riskleri ve negatif eğilim gösteren ölçüm sonuçlarını dikkate alarak ileride ortaya çıkması muhtemel bu uygunsuzlukları belirlemek mümkün olabilir.

Önleme odaklı yaklaşımın, diğer yaklaşıma göre bir farkı da, denetim çıktılarında görülebilir. Bu yaklaşımda olası uygunsuzluklar da – uygunsuzluk riski olarak –elde edilir. Olası uygunsuzlukların iki şekilde kullanılması mümkündür:

1. Eğer elde edilen veriler uygunsuzluk gerçekleşmeden önce önlemek için işlem almaya yeterliyse, uygunsuzluğun önlenmesi sağlanabilir.
2. Elde edilen veriler önleme için yeterince temel oluşturmuyorsa, bu veriler ilerleyen aşamalardaki kalite güvence faaliyetlerinin planlamasında kullanılabilir.

Kuşkusuz MilSOFT'un önleme odaklı yaklaşımının en önemli bileşeni, sadece gereksinimleri değil, benzer projelerin verilerini de temel alan denetim yaklaşımıdır. Bu yaklaşımın farkları şunlardır:

- Denetimlerin uygunsuzluğa yol açabilecek noktaları belirleme odaklı olması.
- Kullanılan soru listelerinin, projenin özelliklerine ve risklerine göre özelleşmiş olması.
- Denetimin kapsamının ve zamanlamasının, olası uygunsuzlukları saptayabilmek adına esnek tutularak, denetimden kısa süre önce netleştirilmesi.
- Saptanan olası uygunsuzlukları önlemede kalite güvence personelinin görev alması.

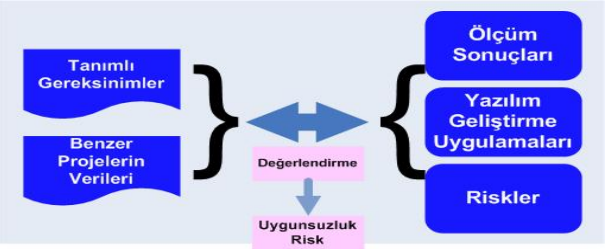
3.2. Bileşenlerin Biraraya Getirilmesi ve Etkileşimler

Altbölüm 3.1'de detayları anlatılan yaklaşımın bileşenlerini aşağıdaki şekilde listelemek mümkündür:

- Temeli oluşturan denetim mekanizması
- Yazılıma yönelik gereksinimler
- Geçmiş projelerin uygunsuzluk verileri
- Yazılım geliştirme faaliyetlerinin çıktıları
- Ölçüm sonuçları ve risk analizleri

Burada listelenen bileşenler, yaklaşımın her yeni yazılım geliştirme projesine uygulanmasında farklılık gösterebilir; denetim mekanizması dahi önceki proje deneyimlerinden faydalanarak güncellenebilir. Geçmiş projelerin uygunsuzluklarına dair veriler ise, her proje sonrası eklenen verilerle yenilenmektedir.

Bu bileşenlerin arasındaki etkileşimler Şekil 4'teki gibi özetlenebilir.



Şekil 4 Önleme Odaklı Kalite Güvencede Denetim

Altbölüm 3.1'de bahsedilen bileşenler, MilSOFT'un yaklaşımında, sırasıyla aşağıdaki adımlar takip edilerek gerçekleştirilmektedir:

1. Yazılım geliştirme boyunca, kritik noktalardaki ölçüm ve risk analizi faaliyetlerinin tanımlanması, veya tanımlı olanların projeye uygulanabilirliğinin analiz edilmesi.
2. Uygulanacak proje yaşam döngüsünün ve ilgili süreçlerin projeye uyarlanması aşamasında müdahil olunması.
3. Geçmiş tecrübeler, gereksinimler ve başlangıçta öngörülen riskler ışığında denetimlerin takvimlendirilmesi.
4. Planlanan denetimlerle, yazılım geliştirme faaliyetlerinin çıktılarının, ölçüm sonuçlarının ve risk tanımlamalarının incelenerek mevcut ve olası uygunsuzlukların saptanması.
5. Mevcut uygunsuzlukların düzeltilmesi; olası uygunsuzlukların eldeki verilerle mümkünse önlenmesi, değilse denetim planlamasına girdi oluşturması.

Yaklaşımın tasarlanmasında bazı hususlara dikkat edilmelidir. Örneğin, geçmiş verilerin projede kullanılabilir biçimde işlenmesi ve projeler arası benzerliklerin doğru tanımlanması gerekmektedir. Çok farklı projelerin verileri birarada kullanıldığında, önlemeye yönelik bir denetim planlaması yapmak mümkün olmayabilir; ancak projeleri çok fazla açıdan gruplamak, benzer projelerin sayısını düşürerek yararlı verinin ulaşılabilir olmaktan çıkmasına yol açabilir.

3.3. Uygulamaya Geçiş

Önceki bölümlerde bileşenleri ve aralarındaki etkileşimler verilen yaklaşım uygulamaya geçirilirken, şu sıralama önerilmektedir:

1. Mevcut kalite güvence süreçlerinin analizi ile, önleme amaçlı yaklaşım uygun noktalara yerleştirilmelidir.
2. Kalite güvence personeli, olası uygunsuzlukların analizi için gerekli yetkinliğe ve bakış açısına kavuşturulmalıdır.
3. Yazılım geliştirme faaliyetleri ile ilgili tanımlı ölçümler ve risk analiz yaklaşımları incelenerek kalite güvencede kullanıma yönelik strateji oluşturulmalıdır.
4. Geçmiş yazılım geliştirme projelerindeki uygunsuzluklar analiz edilerek bir sonraki dönem kalite güvence planlamasına temel oluşturacak veriler belirlenmelidir.
5. Yazılım geliştirme ekipleri bilgilendirilerek yaklaşımın katkıları ve onlardan beklenenler anlatılmalıdır.

Burada önemli hususlardan biri, kalite güvence personelinin yetkinliklerinin yaklaşımı destekler biçimde geliştirilmesidir. Ölçüm sonuçları ve risk analizi çıktılarının etkin değerlendirilmesi, olası uygunsuzlukların saptanmasında kritik olduğu için, ilgili personelin bu konularda belirli bir bilgi seviyesinde olmaları beklenmektedir.

Bir başka husus ise, yazılım geliştirme ekiplerinin de yaklaşımın getirdikleri ve kendilerinden beklenenler konusunda bilgi sahibi olmalarıdır. Denetimler ile değerlendirilen uygulama çıktıları, ölçüm sonuçları ve risk analizleri bu ekip tarafından ortaya konduğu için, bu verilerin projeyi mümkün olduğu kadar doğru yansıtması, kalite güvencenin etkin ve verimli çalışması için gereklidir. Bu yüzden, yazılım geliştirme ekibinin de sürece tam katılımı sağlanmalıdır.

4. Önleme Odaklı Kalite Güvencenin Sonuçlarının Değerlendirilmesi

MilSOFT'un uygulamaya koyduğu yaklaşımda amaç, uygunsuzlukların oluşmadan tespiti ve önlenmesi için etkin ve verimli bir sistem kurulmasıdır. Bu nedenle, yeni sistemin etkinliğinin ve verimliliğinin takibi amacıyla kriterler tanımlanmıştır.

Yaklaşımın etkinliğini değerlendiren kriter Tablo 1 'dedir.

	Değer	Sonuç
Başarılı Önleme Sayısı / Denetim Sayısı	< Hedef	Başarısız
	>= Hedef	Başarılı

Tablo 1 Kalite Güvence Etkinlik Başarı Kriteri

Yaklaşımının verimliliğini değerlendiren kriter ise Tablo 2'dedir.

	Değer	Sonuç
Başarılı Önleme Sayısı / Denetim Eforu	< Hedef	Başarısız
	>= Hedef	Başarılı

Tablo 2 Kalite Güvence Verimlilik Başarı Kriteri

Buradaki hedefler, MilSOFT'un geçmiş tecrübeleri ve gelecek dönem organizasyonel hedefleri dikkate alınarak belirlenmiştir. Her organizasyonun, hedeflerini bu doğrultuda belirlemesi beklenmelidir.

Önleme odaklı kalite güvencenin başarısının değerlendirilmesinde, başarılı önleme sayısının belirlenmesi için, her bir işlemin kayıt altına alınıp takip edilmesi gereklidir. Bu kapsamda, olası uygunsuzluğun gerçekten önlenip önlenmediği izlenmelidir; bu izleme, işlemin alınması aşamasında uygunsuzluğun ortaya çıkması beklenen nokta belli ise o noktaya kadar, değilse kalite güvence ve yazılım geliştirme personeli tarafından kararlaştırılacak bir süreyle yapılmalıdır.

5. Yaklaşımla İlgili Riskler

Önleme odaklı kalite güvence yaklaşımının taşıdığı riskleri aşağıdaki başlıklar altında toplamak mümkündür.

- Yaklaşımın anlaşılması ile ilgili riskler
- Uygulamayla ilgili riskler
- Yaklaşımın maliyetiyle ilgili riskler

Yaklaşımın anlaşılması ile ilgili riskler, beklentilerin ilk aşamada çok yüksek tutulması ve, yazılım geliştirme ve proje yönetimi ekipleri tarafından önlenmesi mümkün uygunsuzlukların kalite güvence tarafından tespit edilebileceği varsayımıyla sorgulanmamasından kaynaklanmaktadır. Oysa kalite güvence, destek fonksiyonu olarak, ancak etkin yazılım geliştirme ve proje yönetimi ile sonuç verebilir.

Uygulamayla ilgili riskler, ya kalite güvence personelinin yetkinliklerinin yaklaşıma uyumsuzluğundan, veya yazılım geliştirme personelinin yaklaşımı destekler çalışmamasından doğmaktadır.

Yaklaşımın maliyeti ile ilgili riskler de söz konusudur. Tanımlanan mekanizma, olası uygunsuzlukları yakalamak adına projenin tüm verilerini değerlendirmeyi öngörüyorsa, verimsizlik riski ortaya çıkmaktadır. Bu yüzden, değerlendirilecek veriler, tecrübeler ve yazılımın özellikleri gözönüne alınarak seçilmelidir.

6. Tartışma

Yazılım geliştirme doğası gereği bilgi birikiminin etkin biçimde kullanılmasına dayandığından, ortaya çıkardığı ürün de teknik olarak karmaşıktır ve bu konuda bilgisi sınırlı bireylerin ürünü değerlendirmesi ancak sınırlı seviyede mümkün olabilir. Bu nedenle kalite güvencede amaçlanan, asıl işi yazılım geliştirme olmayan bireylerin bu faaliyeti değerlendirmesine olanak sağlayacak yazılım kalite güvencesi metodlarına odaklanmak olmalıdır[10].

Bu doğrultuda, yazılım kalite güvencesinin, sadece etkin değil, verimli de çalışabilmesi beklenmelidir. Böyle bir beklenti beraberinde, kalite güvencenin mümkün olduğu kadar az maliyetle, mümkün olduğu kadar büyük tasarruf sağlaması gerekliliğini getirmektedir. Bunu sağlamanın bir yolu, uygunsuzlukların oluşmadan önlenmesidir.

Organizasyonel hedeflerle örtüşen, kurum kültürü ve süreçleriyle uyumlu, önleme odaklı bir kalite güvence, yazılım geliştirme ekibinin öngöremediği sorunları önceden belirleyip önlenmesine yardımcı olabilir. Bu da, yazılım geliştirmenin başarı olasılığını yükseltecektir.

7. Sonuçlar

Yazılım kalite güvencesi, özellikle zaman baskısı bulunan ortamlarda, sadece mevcut uygunsuzlukları ortaya çıkartarak düzeltilmelerini sağlayacak şekilde yapılandırılmışsa, uygunsuzlukların düzeltilmeleri çok maliyetli olabileceği gibi, kimi zaman mümkün olamayabilir. Böyle bir durumda, geçmiş tercübelere dayanarak oluşturulan önleme odaklı bir kalite güvence sistemi, düzeltme odaklı kalite güvenceye göre daha etkin ve verimli sonuçlar verebilir, uygunsuzlukların oluşmadan önlenmesi vasıtasıyla maliyetlerin düşmesine ve yazılım geliştirme takvimi ile ilgili risklerin önlenmesine katkıda bulunabilir.

Uluslararası rekabetin giderek arttığı yazılım geliştirme sektöründe, Türk yazılım endüstrisinin rekabet edebilirliğini arttırmak için etkin yöntemlerden birisi de, etkin ve verimli bir kalite güvence mekanizması oluşturmaktır. Bu doğrultuda MilSOFT, modeller, standartlar ve kendi tecrübelerini temel alan ve yukarıda tarif edilen sistemi oluşturmuştur. Bu sistem esnekliği nedeniyle tüm yazılım organizasyonlarının adapte edebileceği bir yaklaşım içermektedir.

8. Kaynakça

[1] Software Engineering Body of Knowledge, "*Chapter 11: Software Quality." 2.1 Software Quality Assurance,* 2008.

[2] CMMI Product Development Team. *CMMI for Development, Version 1.2 (CMU/SEI-2006-TR-008, ESC-TR-2006-008),* Software Engineering Institute, Carnegie Mellon University, August 2006.

[3] International Organization for Standardization. *ISO 9001, Quality Management Systems—Requirements*, 2000.

[4] NATO. *Yazılım İçin Ömür Devri Boyunca Birleştirilmiş NATO Kalite Gereksinimleri, Baskı 1*, 2001.

[5] Maraia, V. *How Many Lines of Code in Windows?*. Knowing.NET, December, 2005.

[6] Meyer, B. *Software: Product or Service?,* Software Development, 2001.

[7] Rakitin, S. R., *Balancing Time to Market and Quality*, Software Quality Professional, 1999.

[8] Baker, E.B., *Which Way, SQA?*, IEEE Software, January / February 2001.

[9] Tavassoli, D., *Strategic QA: Steps to Effective Software Quality Assurance*, IBM, October 2008.

[10] Humphrey, W. *Managing the Software Process*. Addison-Wesley Professional, 1990.