YAZILIM TEST MÜHENDİSLİĞİ

Dr. Öğr. Üyesi Hasan YETİŞ

Yazılım Ölçütleri

- Bir yazılım veya sistemin geliştirilmesinde ölçüt kriterlerine ihtiyaç duyulmaktadır.
- Ölçüt yazılım mühendisliğinde önemli bir rol oynar ve **metrik** olarak da ifade edilmektedir.
- Yazılım ölçütleri, çeşitli kriterler göz önünde bulundurularak sınıflandırılır.
  - ***Statik ölçütler***, yazılım sisteminin çalıştırılmasına gerek kalmadan, yazılım sisteminin kod yapısıyla ilgili belgelerden elde edilebilen ölçütlerdir.
  - **Dinamik ölçütler**, yazılım çalışmasına bağlı olarak elde edilen verilerden oluşmaktadır.

Yazılım Ölçütleri

- Yazılım ölçütlerinin bazılarında yalnızca parametre erişimleri dikkate alınırken, bazı ölçütlerde ise metotların tüm veri erişimleri değerlendirilir.

Şekil 8.3. Yazılım ölçütleri

Yazılım Ölçütleri

- Bir projede takım liderleri çoğunlukla bu metriklerle ilgilenmektedir. Bu gruptaki metriklerden en önemlileri **zaman, çaba ve maliyet** ile ilişkili olanlardır.
- Yazılımların her birindeki süreç, proje ve ürün ölçütlerinin değerlendirilmesindeki parametreler de **birbirinden farklıdır**.

Yazılım Ölçütleri

- Süreç ölçütleri, yazılım yaşam döngüsü ile ilişkili özellikler ölçülür.
- Örneğin; süreç ölçütlerini tanımlarken,
  - **süreç içerisinde harcanan çaba-emek,**
  - **üretilme zamanı,**
  - **geliştirilen sistemde kusur bulma ve kusurun ortadan kaldırılması için gereken süre,**
  - **sorunların çözümünde sistemin yanıt verme süresi**
- gibi parametreleri içerir.

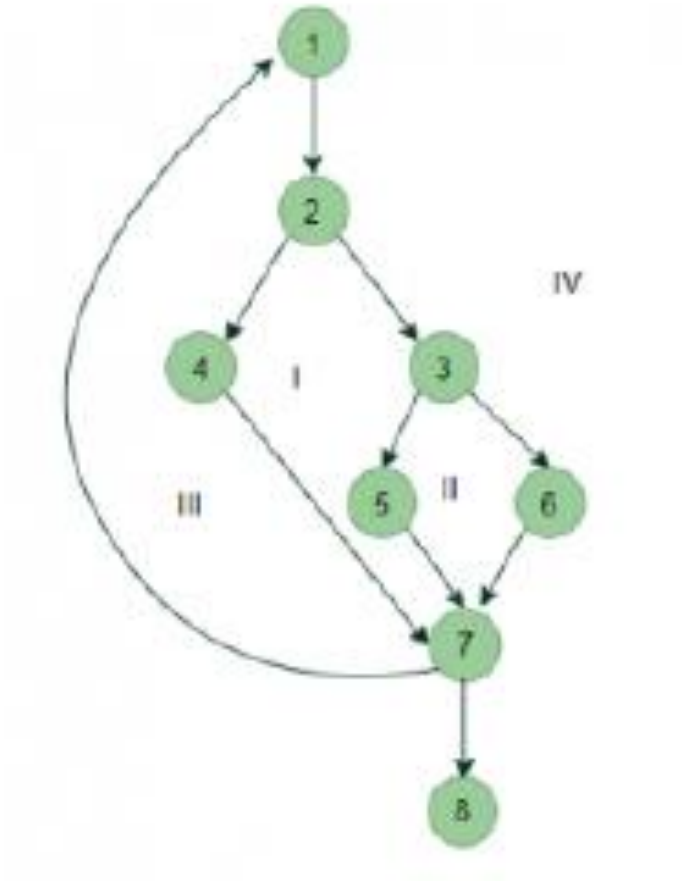
Kaynak Kod için Ölçütler

Ölçüt adı	**Açıklama**
Kod satırları *(LOC-Lines Of Code)*	Kod bloğundaki satır sayıları.
Kaynak kod satırları *(SLOC-Source Lines of Code)*	Kod bloklarındaki boşluk satırları hariç ama yorum satırları da dahil olan tüm kod satırlarının sayısıdır.
Koddaki fiziksel Satırlar *(PLOC-Physical Lines Of Code)*	Kod bloklarındaki açıklama satırları ve boş satırlar dışında kalan tüm fiziksel satırların sayısıdır. Yorum ve boşluk içermeyen satırlar *(Non-Comment Non-Blank-NCNB)* olarak da bilinir.
Koddaki mantıksal satırlar *(LLOC-Logical Lines of Code)*	Kod bloklarındaki `if, while, for, switch-case, do-while` gibi mantıksal ifadelerin bulunduğu satırların sayısıdır.
Çalıştırılabilir yordam sayısı *(EXEC-Executable Statements)*	Kod bloklarında bulunan içinde hesaplama adımları bulunduran, fonksiyon, altyordam, altprogram, metot gibi kod parçacıklarının toplam sayısıdır.
Yorum oranı *(CP-Comment Percentage)*	Program içerisinde bulunan yorum satırlarının, programda bulunan yorum ve boşluk içermeyen toplam satır sayısına oranıdır. Yorum oranının yüksek olması, programın anlaşılabilir ve okunabilirliğini arttırır.
Döngüsel karmaşıklık *(CC-Cyclomatic Complexity)*	Koddaki koşullu gidiş yollarının doğrusal dallanmaları şeklinde ifade edilir.

Kaynak Kod için Ölçütler

- *Döngüsel karmaşıklık (Cyclomatic Complexity - CC):* Döngüsel karmaşıklık ölçütü, koşullu ifade ve yapıların sayısı ile hesaplanır. Kodda bulunan kontrol blokları ve döngüsel yapılar koşullu dallanmaya sebebiyet veren ölçütler olarak değerlendirilir.
- Döngüsel karmaşıklığı yüksek olan modüllerin, döngüsel karmaşıklığı daha düşük olan modüllere göre hataya eğilimli olma olasılığı daha yüksektir.
- Döngüsel karmaşıklık, yazılımın mantıksal karmaşıklığının bir ölçüsüdür. Bu ölçü, bağımsız yolların sayısını tanımlamak için kullanılır.

Kaynak Kod için Ölçütler



Şekil 8.5. Çizgesel diyagram örneği

$$CC = (KenarSayisi) - (DugumSayisi) + 2$$

- Formülde bulunan düğümlerden her biri mantıksal bir dalı, kenarlardan her biri düğümler arasındaki bağlantıyı temsil eder.
- Yazılım Mühendisliği Enstitüsü (Software Engineering Institute - SEI) tarafından kabul gören döngüsel karmaşıklık ve risk değerlendirmesi

Döngüsel Karmaşıklık	Risk Değerlendirmesi
1-10	Düşük riske sahip karmaşık olmayan modül
11-20	Orta riske sahip az karmaşık modül
21-50	Yüksek derecede riske sahip olan karmaşık modül
50-∞	Çok yüksek derece riskli test edilemez modül

Nesne tabanlı tasarım ölçütleri

- Program yapısı, modül bağımsızlığı, ilişkisiz modüller, veritabanı boyutu, veritabanı bölümlendirme, modül giriş-çıkış karakteristiği gibi önemli parametreler ölçülür.
- Sınıf, bir nesne tabanlı sistemin temel birimidir.
- Bu nedenle, nesne tabanlı tasarım kalitesinin değerlendirilmesinde, **tek bir sınıf, sınıf hiyerarşisi ve sınıf işbirlikleri** için ölçüler ve metrikler çok değerlidir.

Nesne tabanlı tasarım ölçütleri

- **Chidamber ve Kemerer (CK)**, en yaygın olarak başvurulan nesne tabanlı yazılım metriklerini önermişlerdir

Ölçüt adı	Açıklama
Sınıf Başına Ağırlıklı Metot Sayısı *(WMC-Weighted Methods per Class)*	Bir sınıftaki metotların karmaşıklık derecesinin veya sayısının toplamı olarak belirlenir. WMC, sınıfın geliştirilmesi ve bakımı için harcanacak sürenin tahminini belirlemede yardımcı olur. Sınıf başına ağırlıklı metot sayısı ile nesne yönelimli bir yazılımın anlaşılırlık, yeniden kullanılabilirlik ve dayanıklılık gibi ölçütleri değerlendirilir. Bundan dolayı, WMC makul olduğu kadar düşük tutulmalıdır.
Kalıtım ağacının derinliği *(DIT-Depth of Inheritance Tree)*	Bu metrik, düğümden ağacının köküne kadar olan maksimum uzunluktur. Derin bir sınıf hiyerarşisi, daha fazla tasarım karmaşıklığına yol açar. Olumlu tarafı, büyük DIT değerleri, birçok yöntemin yeniden kullanılabileceğini belirtir. Herhangi bir sınıftan türetilmeyen değerler için DIT, 0 olarak belirlenir. Çoklu kalıtımın söz konusu olduğu sistemlerde ölçüt için köke en uzak olan kalıtım dikkate alınır. Bu ölçütten yararlanarak yazılımın verimliliği, yeniden kullanılabilirliği, anlaşılırlığı, test edilebilirliği gibi kavramların oranları belirlenir.

Nesne tabanlı tasarım ölçütleri

Alt Sınıf Sayısı *(NOC-Number Of Children)*	Alt Sınıf Sayısı (Number Of Children-NOC), bir sınıftan doğrudan türetilen toplam alt sınıfların sayısıdır. Çocukların sayısı arttıkça yeniden kullanım artar. Ancak alt sınıf sayısı yüksek olan sistemler, daha karmaşık, düşük bakılabilirlik oranına ve daha çok hataya sahip, yüksek zaman-çaba-maliyet ile test edilebilirlik özelliklerine sahiptir.
Nesne Sınıfları Arasındaki Bağımlılık *(CBO-Coupling Between Object Classes)*	Bir sınıfın bağımlı olduğu sınıfların sayısı toplamı olarak belirlenir. Genel olarak, her sınıf için CBO değerleri makul olduğu kadar düşük tutulmalıdır. CBO, yeniden kullanılabilirlik ve verimliliğin ölçülmesinde kullanılır. Bir sınıfın az bağımlılığa sahip olması, farklı uygulamalarda daha kolay yeniden kullanılabilirliğe sahip olması demektir. Yüksek bağımlılık oranı, değişim duyarlılığını arttırır. Bu durumda yazılımın bakımı daha zor olur. Bağımlılığın fazla olması testlerin daha dikkatli bir şekilde yapılması gerektiğini, dolayısıyla test maliyetlerinin de aynı oranda artacağını gösterir.

Nesne tabanlı tasarım ölçütleri

Sınıfın Tetiklediği Metot Sayısı *(RFC-Response For a Class)*	RFC, bir sınıfta bulunan nesnenin metotlarının çağrılması ile bu nesnenin tetikleyebileceği tüm metotların sayısı ile belirlenir. RFC arttıkça, test dizisi büyüdüğü için gereken test çabaları da artar. Ayrıca, RFC arttıkça, genel tasarım karmaşıklığının da artar. RFC, test edilebilirlik, bakım ve test zaman-maliyet değerleri hakkında fikir verir. Bu nedenle RFC değerinin düşük olması, sistemin test edilebilirlik oranının yüksek, test için harcanan zaman-maliyet değerlerinin düşük, sistem dayanıklılığının yüksek ve bakımın kolay olduğunu gösterir.
Metot İçi Uyumsuzluk *(LCOM-Lack of Cohesion in Methods)*	LCOM, metriği ile metotların birbirlerine olan benzerliği ölçülür. LCOM yüksekse, yöntemler nitelikler aracılığıyla birbirine bağlanabilir. Bu, sınıf tasarımının karmaşıklığını artırır. LCOM için yüksek bir değerin haklı gösterilebileceği durumlar olsa da, uyumun yüksek tutulması arzu edilir; yani, LCOM'un düşük tutulması en iyi durumdur. LCOM değeri ne kadar yüksekse, o kadar çok durum test edilmelidir. LCOM yardımı ile verimlilik ve yeniden kullanılabilirlik derecesi hakkında bilgi edinilir.

Nesne tabanlı tasarım ölçütleri

- *Brito e Abreu MOOD ölçüt kümesi*, nesne yönelimli yöntemin yapısal noktalarını esas alır.

Ölçüt adı	Açıklama
Metot Gizleme Faktörü *(MHF-Method Hiding Factor)*	Metot Gizleme Faktörü(Method Hiding Factor-MHF), sistemde tanımlı olan bütün sınıflardaki çağrılabilen metotların, var olan bütün metotlara oranıdır. Kalıtımla gelen metotlar hesaplamaya dahil edilmez. Bu ölçüt, sınıf tanımının görünürlüğü hakkında yorum yapma şansı tanır.
Nitelik Gizleme Faktörü *(AHF-Attribute Hiding Factor)*	Kalıtımla gelen niteliklerin dahil edilmemesi ve sistemde tanımlı olan tüm sınıflardaki görünür niteliklerin, tüm niteliklere oranı olarak hesaplanır. MHF metriğinde olduğu gibi AHF metriğinin hesaplanmasında da kalıtım ile gelen metotlar hesaplamaya dahil edilmez ve MHF gibi sınıfların görünürlüğü hakkında bilgi sahibi olmak için bu ölçütten yararlanılabilir. Bu ölçütle sınıf tanımının görünürlüğü ölçülür.

Nesne tabanlı tasarım ölçütleri

Metot Kalıtım Faktörü *(MIF-Method Inheritance Factor)*	Var olan bütün sınıflardaki kalıtım ile gelen metot sayısının, sınıflarda bulunan toplam metot sayısına oranı olarak tanımlanır.
Nitelik Kalıtım Faktörü *(AI-Attribute Inheritance Factor)*	Sistemde tanımlı olan tüm sınıflardaki kalıtım yardımıyla gelen niteliklerin, sistemde var olan bütün niteliklere oranı ile belirlenir.
Çok Biçimlilik Faktörü *(PF-Polymorphism Factor)*	Belirlenen bir sınıf için sistemdeki farklı çok şekilli olan durumların, maksimum olası diğer çok şekilli durumlara oranı olarak belirlenir.
Bağımlılık Faktörü *(CF-Coupling Factor)*	Sistemde sınıflar arasındaki bağımlılık sayısının toplamının, oluşabilecek maksimum bağımlılık sayısına oranı olarak belirlenir. Oranların belirlenmesinde kalıtım ile gelen bağımlılıklar dikkate alınmaz.

Nesne tabanlı tasarım ölçütleri

- Bansiya ve Davis tarafından irdelenen QMOOD(Quality Model for Object Oriented Design) ölçütlerinde

Ölçüt adı	Açıklama
Ortalama Ata Sayısı *(ANA-Average Number of Ancestors)*	Tüm sınıflarda bulunan ve Chidamber-Kemerer ölçütleriyle hesaplanan DIT değerlerinin ortalaması olarak hesaplanır. Yazılımda soyutlamanın kullanımı hakkında bilgi verir.
Metotlar Arası Uyumluluk *(CAM-Cohesion Among Methods)*	Bansiya tarafından tam olarak tanımı verilmemiş olsa da literatür tanımı olarak metotların imzalarına bakarak ölçülen uyumluluğun hesaplanması olarak ifade edilebilir.
Sınıf Arayüz Boyutu *(CIS-Class Interface Size)*	Sınıfta bulunan erişime açık (public) metotların sayısıdır. Bu sayı sayesinde yazılım sisteminin mesajlaşma özelliği hakkında yorum yapılabilir.
Veri Erişim Metriği *(DAM-Data Access Metric)*	Yazılım sisteminin sınıflarında bulunan özel erişimli (private) ve korumalı (protected) olarak tanımlanan niteliklerin, var olan tüm niteliklere oranı olarak belirlenir. Bu ölçüt ile yazılımın kapsülleme özelliği hakkında bilgi sahibi olunabilir.

Nesne tabanlı tasarım ölçütleri

Doğrudan Sınıf Bağımlılığı *(DCC-Direct Class Coupling)*	Bir sınıfın parametre olarak kabul edildiği sınıfların toplam sayısı ile bu sınıf nitelik olarak içeren sınıf sayılarının toplamı olarak değer alır. Sınıflar arası bağımlılık bu ölçüt ile belirlenir.
Kümeleme Ölçüsü *(MOA-Measure Of Aggregation)*	Tanımlanan sınıf bildirimlerinin, tanımlı olan temel veri tiplerine oranı olarak belirlenir.
İşlevsel Soyutlama Ölçüsü *(MFA-Measure of Functional Abstraction)*	Sistemdeki tüm sınıflarda bulunan türetilmiş metot sayıları toplamının, tüm metot sayıları toplamına oranıdır. Yazılım sisteminin kalıtım özelliği bu ölçüt ile değerlendirilir.
Metot Sayısı *(NOM-Number of Methods)*	Sınıfta tanımlı olan metot sayısının toplamıdır. Sınıflarda bulunan metot sayısı, sınıf karmaşıklığının da göstergesidir.

Web tabanlı tasarım ölçütleri – Arayüz Ölçüt

Önerilen Metrik	Açıklama
Sayfa düzeni	Arayüz içindeki varlıkların görüş alanına uygun konumları.
Düzen karmaşıklığı	Bir arabirim için tanımlanan farklı bölgelerin tanımlanması.
Düzen bölgesi karmaşıklığı	Bölge başına eklenen ortalama farklı bağlantı sayısı.
Tanıma karmaşıklığı	Kullanıcının gezinme veya veri girişi kararı vermeden önce bakması gereken ortalama farklı öğe sayısı.
Tanıma süresi	Bir kullanıcının belirli bir işlemi yaparken uygun eylemi seçmesi için geçen ortalama süre.
Klavye bilgi girişi	Belirli bir işlev için gereken ortalama tuş vuruşu sayısı.
Fare klikleri	İşlev başına ortalama fare seçme sayısı.
Seçim karmaşıklığı	Sayfa başına seçilebilecek ortalama bağlantı sayısı
İçerik edinme süresi	Web sayfası başına düşen ortalama metin kelime sayısı.
Bellek yükü	Belirli bir hedefe ulaşmak için kullanıcının hatırlaması gereken farklı veri öğelerinin ortalama sayısı

Web tabanlı tasarım ölçütleri – Estetik Ölçüt

Önerilen Metrik	Açıklama
Kelime sayısı	Bir sayfada görünen toplam kelime sayısı.
Gövde metni yüzdesi	Gövde ve görüntü metni arasındaki kelimelerin yüzdesi.
Vurgulanan gövde metni yüzdesi	Gövde metninin kalın, büyük harfle yazılmış vurgulanan kısmı.
Metin konumlandırma sayısı	Soldan hizalı metin konumunda değişikliklerin sayısı.
Metin kümesi sayısı	Renkle vurgulanan metin alanları, kenarlıklı bölgeler, kurallar veya listeler.
Bağlantı sayısı	Bir sayfadaki toplam bağlantı sayısı.
Sayfa boyutu	Sayfanın yanı sıra öğeler, grafikler ve stil sayfaları için toplam bayt
Grafik yüzdesi	Grafikler için olan sayfa baytlarının yüzdesi
Grafik sayısı	komut dosyalarında, uygulamalarda ve nesnelerde belirtilen grafikler hariç bir sayfadaki toplam grafik ve resim sayısı.
Renk sayısı	Kullanılan toplam renkler.
Yazı tipi sayısı	Kullanılan toplam farklı yazı tipleri.

Kaynak Kod için Ölçütler

- Yazılımın kalitesinin değerlendirilmesinde sürdürülebilirlik önemli bir etkendir.
- Sürdürülebilirlik için kodun sürekli gelişime açık olması ve bakımının kolay olması gerekmektedir.
- Dolayısıyla kodun okunabilirliği ve anlaşılabilirliğinin yüksek olması kalite oranını da arttırır.
- Yazılım kod kalitesinin belirlenmesi için çeşitli ölçütler önerilmiştir.
- Önerilen ölçütlerden en temel olanı geleneksel McCabe kalite ölçütleridir.