Nesneye Yönelik Sistemler İçin Bir Uyum Ölçütü Önerisi: COMIAS

H.Özlem Ertemel[1] Yunus Emre Selçuk[2] Oya Kalıpsız[3]

[1]TUBITAK Marmara Research Center, IT Institute, 41470 Gebze, Kocaeli, Turkey
[2, 3]Yıldız Technical University, Computer Eng. Dept., 34349 Beşiktaş, İstanbul, Turkey

[1]ozlem.ertemel@bte.mam.gov.tr [2] yunus@ce.yildiz.edu.tr
[3]kalipsiz@yildiz.edu.tr

Özetçe

Yazılım kalitesinin ölçümü, karşılaşılan zorluklara rağmen, yazılımın birçok yönden değerlendirilmesi ve iyileştirilmesine olanak tanıması nedeniyle önem arz etmektedir. Bu çalışmada nesneye yönelik yazılımların yaşam döngüsünün her aşamasında kullanılabilecek bir uyum (cohesion) ölçütü olan COMIAS tanıtılmakta ve ilk deneysel geçerlemesine yer verilmektedir.

1. Giriş

Yazılım kalitesi kavramı, halen belirsiz ve farklı kişilere farklı anlamlar ifade eden bir kavramdır [1]. Geliştiricinin bakış açısıyla ya da içsel bakış açısı ile kalite, maliyet ve gecikmenin doğru tahminine, daha kolay teste, daha iyi bakıma giden yoldur [2]. Kullanıcı tarafından ya da dışsal bakış açısı ile bakıldığında ise kullanım kolaylığı, estetik, anlaşılabilirlik vb. kavramlar ifade edebilmektedir.

Ölçme eylemi, somut veya soyut bir varlığın sahip olduğu bir özelliğini, sayısal veya derecelendirilmiş bir veri olarak ifade etmektir. Ölçülecek özelliği ölçme yolu ve biçimine ölçüt, bir ölçüt kullanılarak yapılan ölçme eyleminin sonucunda elde edilen veriye ise ölçüm adı verilir. Ölçme eylemi işimize yarayacak, anlamlı sonuçlar elde etmek için yapılır. Amaç yazılımın kalitesini iyileştirmek olarak belirlendiği zaman, yazılımın kalitesi yazılım yaşam döngüsünün çeşitli aşamalarında ölçülmeli, elde edilen ölçümler karşılaştırılmalı ve değerlendirilmelidir.

Ölçme yapabilmek için önce bir veya birkaç ölçüt belirlenmelidir. Bu aşamada tanımlanması bile zor bir kavram olan yazılım kalitesi ile ilgili üzerinde anlaşmaya varılmış özellikler bulmanın zorluğu ile karşılaşılır. Yazılım yaşam döngüsünün erken aşamalarında ölçüme başlayabilmek için, içsel kalite özelliklerinin kullanılması gerekecektir. Bağlaşım (coupling) [3] ve uyum (cohesion) [4], bu amaçla kullanılan en temel özelliklerdir. Nesneye dayalı programlama (NYP) yaklaşımının doğru kullanımı ile oluşturulan yazılımlarda bağlaşımın düşük ve uyumun yüksek olması beklenir [5]. Literatürde bağlaşım ve uyum için birçok ölçüt bulunmaktadır ancak yazılım ile ilgili ölçütlerin deneysel açıdan doğrulanması kolay olmadığından bu yönde daha çok çalışma yapılmasına gerek vardır [6].

Ölçüm verilerini anlamlı bilgilere dönüştürmek için yapılması gereken yorumlama işlemi zor olabilir. Literatürde "intelligence barrier" [7] olarak geçen bu zorluk, "yorumlama engeli" olarak adlandırılabilir. Bu engel matematiksel ve istatistiksel yöntemlerle aşılmaya çalışılabilir [8], ancak bu yöntemlerin de kolay olduğu söylenemez.

Değinilen bütün zorluklara rağmen, yazılım geliştiricileri ve yöneticileri için yazılımın birçok yönden değerlendirilmesine ve iyileştirilmesine olanak tanımaları nedeniyle, yazılım ölçütleri üzerinde çalışmalar etkin olarak sürmektedir. Bağlaşım ve uyumun ölçülmesi için önerilen ölçüt modelleri incelendiğinde, bağlaşım ölçümü açısından belirli bir olgunluğa erişilmesine rağmen, uyum ölçümünde aynı düzeye ulaşılamadığı değerlendirilmiştir. Bu nedenle COMIAS (Cohesion Method Invocation Attribute Sharing – Uyum Metot Çağrımı Üye Alan Paylaşımı) adı verilen bir uyum ölçütü gerçeklenmiştir. Bu bildiride ağırlıklı olarak COMIAS incelenmektedir.

2. Yapısal Uyum Ölçütleri

Uyum, bir parçanın sorumluluklarının birbirleri ile uyumlu olma oranı olarak tanımlanabilir. Parçanın tanımı ise kullanılan yazılım geliştirme yaklaşımı ve sisteme hangi düzeyde bakıldığına göre değişiklik gösterir. Bu bildiride incelenmekte olan COMIAS, NYP yaklaşımı için tasarlanmış bir sınıf düzeyi uyum ölçütüdür.

Yazılım uyum ölçütlerinin kullanım amaçlarına şu örnekler verilebilir:

- Yazılım kalitesinin değerlendirilmesi [9] [10]
- Yazılım kalitesinin ve hata eğilimliliğinin tahmini [11] [12]
- Yazılım modülerleştirilmesi [13] [14]
- Tekrar kullanılabilir olan bileşenlerin belirlenmesi [15] [16]

Uyum ölçütü önerileri, ölçütün hesaplanması için kullanılan yaklaşımlara göre şu şekilde gruplanabilir:

- Yapısal uyum ölçütleri
- Semantik uyum ölçütleri [17]
- Entropi tabanlı uyum ölçütleri [18]
- Dilim tabanlı ölçütler [19]
- Veri madenciliğine dayalı ölçütler [20]
- Bilgi tabanlı sistemler [21] ve dağıtık sistemler [22] gibi özelleştirilmiş yazılım uygulamaları için ölçütler

COMIAS yapısal yaklaşım kullanılarak gerçeklenmiş bir uyum ölçütüdür. Yapısal uyum ölçütleri temel olarak sınıfın üye alanlarının üye metotlar tarafından kullanımını göz önüne almaktadır. Sınıfın metotları arasındaki ilişkiler bu düşünce ile tanımlanır. Literatürde bulunan yapısal uyum ölçütlerinin kimi üye alan kullanımını (LCOM* [23], Coh [24], RCI [25], CC(X) [26], OL_n [27], DC_D, DC_I, LCC_D, LCC_I [28]), kimi üye alan paylaşımını (LCOM1 [29], LCOM2 [30], LCOM3 [31],

LCOM4 [31], Co [31], TCC [32], LCC [32], CBMC [33], ICBMC [27], CCM [34], ECCM [34], OCC [35], PCC [35], ClassCoh [36], WTCoh [36]), kimi üye metotların çağrımını (LCOM4 [31], Co [31], ICH [24], CBMC [33], ICBMC [27], CCM [34], ECCM [34], OCC [35], PCC [35], DC_D, DC_I, LCC_D, LCC_I [28]), kimi parametre tip kesişimini (CAMC [37], NHD [37]), kimi parametre tip kullanımını (RCI [25]) kimi de bunlardan bazılarının bileşimini esas almıştır. Bu düşünce yapısı, Bunge'nin benzerlik tanımına dayanır [4]. Bunge iki varlığın benzerliğini, bu iki varlığın özelliklerinin kesişimi olarak tanımlar. Bir sınıfın metotlarının benzerlik derecesi ise, yapısal uyum ölçütlerinin hesaplanmasının temelini oluşturur.

Literatürde Bunge'nin benzerlik tanımından yola çıkarak önerilen LCOM [30], LCOM* [23] gibi birçok ölçüt bulunmaktadır.

3. Gerçeklenen Ölçüt: COMIAS

COMIAS uyum ölçütü gerçeklenirken aşağıdaki hususlar dikkate alınmıştır:

- Bir sınıf üye alanlar ve üye metotlardan oluşur. Bu iki önemli elemandan birisinin ölçüt tanımında göz ardı edilmesinin, gerçek uyumun yakalanamamasına neden olacağı düşünülmektedir.
- Bir ölçütün normalize edilmiş olması, bir başka deyişle 0 ile 1 arasında değişen bir değer üretmesi tercih edilir. Normalize edilmiş ölçütlerin yorumlanması daha kolay olacaktır. Buna rağmen literatürde normalize olmayan birçok ölçüt bulunmaktadır.
- Çoğu ölçüt tanımında dolaylı metod ilişkileri göz önüne alınmamıştır. Dolaylı bağlantılar göz ardı edildiğinde gerçek uyumun yakalanamayacağı düşünülmektedir.
- İyi tanımlanmış bir ölçüt mutlaka sezgilerle paralel sonuçlar vermeli, matematiksel olarak belirsiz ve tanımsız sonuçlar vermemelidir.

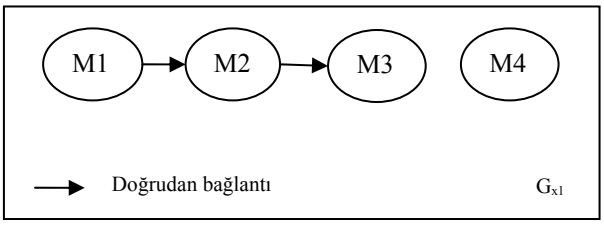
Bir sınıfın uyumu COMIAS ölçütü kullanılarak hesaplanırken üç temel adım yürütülmektedir. Birinci adımda üye metotların arasındaki ilişki metot çağrımı yönünden incelenmekte ve ilk grafik (G_{xA}) elde edilmektedir. İkinci adımda ise üye metotlar, üye alanların kullanımı yönünden incelenmekte ve ikinci grafik (G_{xB}) elde edilmektedir. Üçüncü adımda ise ilk iki adımda elde edilen grafikler tek bir grafikte (G_x) birleştirilmektedir. Bu bölümün geri kalanında söz konusu üç adım ayrıntılı olarak incelenecektir.

3.1. Metot Çağrımlarının İncelenmesi

Metot çağrımları incelenirken, ilk olarak metotlar arasındaki doğrudan bağlantı ilişkileri incelenir. Eğer bir metot diğerini çağırıyorsa o zaman aralarında çağıran taraftan çağrılan tarafa doğru yönlü bir çizgi oluşturulur. Oluşturulan grafik G_{x1} aşağıdaki şekilde ifade edilebilir:
Yönlü bir grafik $G_{x1}(V, E)$ ve $V = M_x$ ise

$$E = \left\{ < m, n > \in V \times V \,\middle|\, (\text{m calls n}) \right\} \qquad (1)$$

Örnek bir G_{x1} grafiği Şekil 1'de görülebilir.

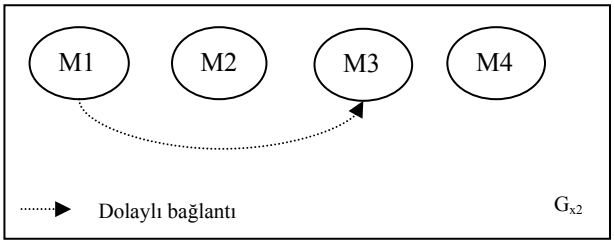


Şekil 1. Örnek bir doğrudan bağlantı grafiği (G_{x1})

Dolaylı bağlantıları bulabilmek için ise G_{x1} grafiğindeki yönlü çizgilerden yararlanılır. Şekil 1 incelendiğinde M1 metodunun M2 metodunu çağırdığı, M2 metodunun da M3 metodunu çağırdığı görülebilir. Dolayısıyla, M1 metodu dolaylı yoldan M3 metodunu çağırmaktadır. M1 metodundan M3 metoduna doğru yönlü bir çizgi oluşturulur. Bu şekilde oluşturulan dolaylı bağlantı grafiği olan G_{x2} aşağıdaki gibi ifade edilebilir:
Yönlü bir grafik $G_{x2}(V, E)$ ve $V = M_x$ ise

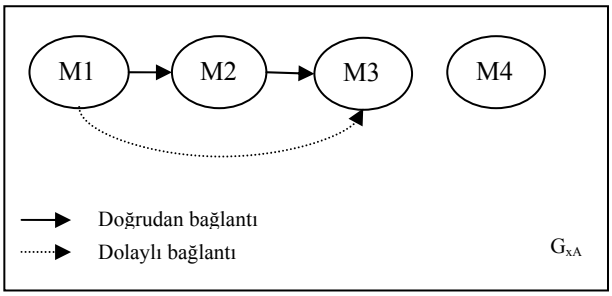
$$E = \left\{ < m, l > \in V \times V \,\middle|\, \begin{matrix} (< m, n > \in G_{x1}) \wedge \\ (< n, l > \in G_{x1}) \end{matrix} \right\} \qquad (2)$$

Örnek G_{x1} grafiğinden elde edilen G_{x2} grafiği Şekil 2'de görülebilir.



Şekil 2. Örnek bir dolaylı bağlantı grafiği (G_{x2})

Metot çağrımlarının incelenmesinin son adımı olarak, önceki adımlarda elde edilmiş olan G_{x1} ve G_{x2} grafikleri tek bir grafikte G_{xA} grafiğinde birleştirilerek, doğrudan ve dolaylı yoldan bağlantılı olan tüm metotlar gösterilmiş olunur. Örnek G_{x1} ve G_{x2} grafiklerinden elde edilen G_{xA} grafiği Şekil 3'te görülebilir.

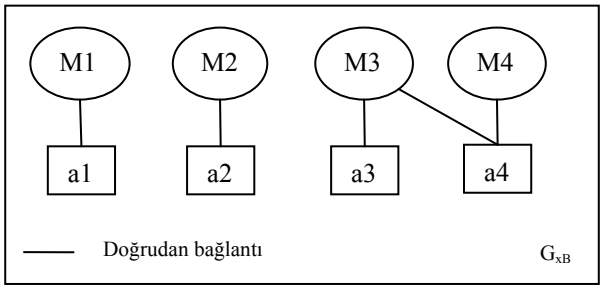


Şekil 3. Örnek bir metot çağrımı grafiği (G_{xA})

3.2. Üye Alan Kullanımlarının İncelenmesi

Üye alanların üye metotlar içerisinden kullanımını görebilmek için metot-alan ilişkisini gösteren bir grafik çizilir. Eğer bir metot bir alanı kullanıyorsa o zaman metot ve

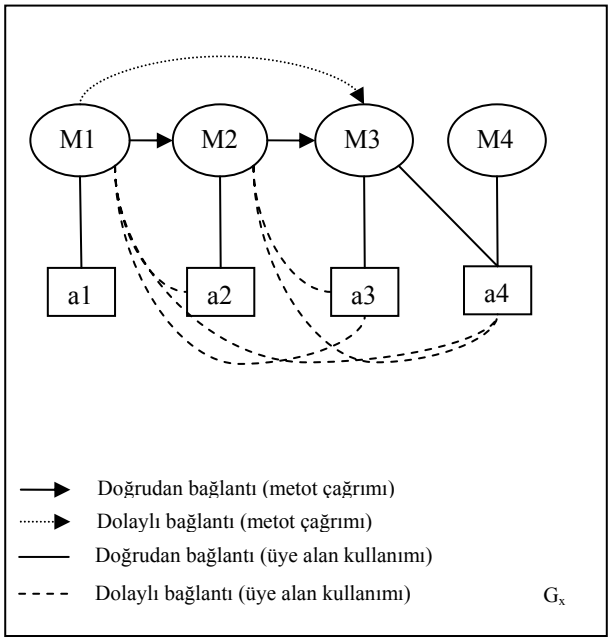
alan arasında bir çizgi oluşturulur. Şekil 4'te bu şekilde elde edilen örnek bir G_{xB} grafiği görülebilir.



Şekil 4. Örnek bir doğrudan üye alan kullanımı grafiği (G_{xB})

3.3. Metot Çağrımı ve Üye Alan Kullanımı

Önceki adımlarda oluşturulan G_{xA} ve G_{xB} grafikleri bu adımda tek bir grafikte birleştirilerek G_x grafiği elde edilir. Dolaylı metot çağrımlarının incelenmesi ve dolaylı üye alan kullanımlarının tespit edilmesi ile G_x grafiğinin son hali elde edilir. Önceki bölümlerde verilen örnek grafikler kullanılarak elde edilen bir G_x grafiği Şekil 5'te görülebilir.



Şekil 5. Örnek bir nihai ilişki grafiği (G_x)

3.4. COMIAS Uyum Ölçütünün Tanımlanması

X bir sınıf, I_x= X sınıfının üye alanlarının kümesi ve M_x = X sınıfının metotlarının kümesi olsun. Yönlü bir grafik $G_x(V, E)$ göz önüne alındığında, iki metot arasındaki metot çağrımı ilişkisi şu şekilde tanımlanacaktır:

$$\vec{Metod}\_cag(M_i, M_k) = \begin{cases} 1 \ , eğer < M_i, M_k > \in G_x \\ \quad ya\ da < M_k, M_i > \in G_x \\ 0 \ , diğer\ türlü \end{cases} \quad (3)$$

İki metot arasındaki üye alan paylaşımı ilişkisi, G_x grafiğine bakılarak şu şekilde bulunur:

$$Attr\_pay(M_i, M_k) = n \left| \begin{matrix} M_i \ ve\ M_k \ metodlarının \\ paylastıkları\ attribute\ sayısı \end{matrix} \right. \quad (4)$$

İki metot arasındaki ilişki, metot çağrımı ve üye alan paylaşımlarının toplamı olarak şu şekilde tanımlanır:

İlişki(M_i, M_k) = M_i metodu ile M_k metodu arasındaki ilişki değeri, $M_i \in M_x$ ve $M_k \in M_x$, $i \neq k$ olarak tanımlansın. Bu durumda;

İlişki(M_i, M_k) = Metod_cag(M_i, M_k) + Attr_pay(M_i, M_k) olarak tanımlanır.

İki metodun ilişkisinden sınıfın toplam ilişki değeri ise şu şekilde bulunur:

$$Toplam\_İlişki(X) = \sum_{i=1}^{m-1} \sum_{k=i+1}^{m} \left[İlişki(M_i, M_k) \right] \quad (5)$$

$$Toplam\_İlişki(X) = \sum_{i=1}^{m-1} \sum_{k=i+1}^{m} \left[\begin{matrix} Metod\_cag(M_i, M_k) \\ +Attr\_pay(M_i, M_k) \end{matrix} \right] \quad (6)$$

İlişki(M_i, M_k) ne kadar yüksekse, iki metot arasındaki ilişki o kadar yüksek değerdedir denilir ve sınıfın toplam ilişki değeri yükselir. Sınıfın toplam ilişki değeri Toplam_İlişki(X) ne kadar yüksekse sınıfın uyumu o kadar yükselir. Toplam ilişki değeri hesaplamasında, metot çağrımı ve üye alan paylaşımı ilişkilerinin ölçümü eşit oranda etkilediği varsayılmaktadır.

COMIAS'ın normalize bir ölçüt olması için bir sınıftaki maksimum ilişki değerini bulup bu sayıya yukarıdaki Toplam_İlişki(X) değerini oranlamak gerekmektedir. Bir sınıftaki maksimum ilişki değeri Max_İlişki(X), maksimum metot çağrımı ilişkisi ve maksimum üye alan paylaşımı ilişkisinin toplamı ile bulunur.

$$Max\_İlişki(X) = Max(Metod\_cag) + Max(Attr\_pay) \quad (7)$$

Bir sınıftaki maksimum metot çağrımı ilişkisi şu şekilde tanımlansın:

Max (Metot_cag) = Her (M_i, M_k) ilişkisinde bir çağrım vardır demektir ve bu da mümkün olan tüm (M_i, M_k) ilişki sayısı ile bulunur:

$M_i \in M_x$ ve $M_k \in M_x$, $i \neq k$. (8)

X sınıfının m tane metodu varsa mümkün olan tüm (M_i, M_k) ilişki sayısı m.(m-1)/2 olacaktır. Her ilişkide bir çağrım olması durumu, maksimum metot çağrımı durumunu verecektir. Maksimum metot çağrımı şu şekilde ifade edilir:

$$Max(Metod\_cag) = \frac{m.(m-1)}{2} \times 1 \quad (9)$$

Tablo 1'de 4 adet metodu olan bir sınıf için bu durum örneklenerek açıklanmıştır. 4 adet metodu olan bir sınıf için tüm ilişkiler ve maksimum metot çağrımı ilişkisi bu tabloda verilmiştir; her metot çifti için aralarında bir çağrım vardır.

Tablo 1. Metotlar arasında maksimum çağrım ilişkisi örneği

	Metot çağrım durumu
(M_1, M_2)	1
(M_1, M_3)	1
(M_1, M_4)	1
(M_2, M_3)	1
(M_2, M_4)	1
(M_3, M_4)	1
Max (Metod_cag)	6 = 6 *1

Aynı şekilde bir sınıftaki maksimum üye alan paylaşımı ilişkisi şu şekilde tanımlanır:

Max (Attr_pay) = Her (M_i, M_k) ilişkisinde sınıfta bulunan tüm üye alanlar paylaşılıyor demektir:

$$M_i \in M_x \text{ ve } M_k \in M_x \text{ , } i \neq k. \quad (10)$$

X sınıfının a tane üye alanı varsa, maksimum üye alan paylaşımı, üye alan sayısının mümkün olan tüm (M_i, M_k) ilişki sayısı m.(m-1)/2 ile çarpımı kadardır. Maksimum üye alan paylaşımı şu şekilde ifade edilir:

$$Max(Attr\_pay) = \frac{m.(m-1)}{2} \times a \quad (11)$$

Tablo 2'de 4 adet metodu ve 4 adet üye alanı olan bir sınıf için maksimum üye alan paylaşımı durumu örneklenerek açıklanmıştır. Her metot çifti var olan tüm üye alanları paylaşıyor demektir.

Tablo 2. Metotlar arasında üye alan paylaşımı ilişkisi örneği

	Üye alan paylaşım durumu
(M_1, M_2)	4
(M_1, M_3)	4
(M_1, M_4)	4
(M_2, M_3)	4
(M_2, M_4)	4
(M_3, M_4)	4
Max (Attr_pay) =	24 = 6 * 4

Ölçütü normalize edebilmek için gereken Max_İlişki(X) değeri aşağıdaki gibi bulunur:

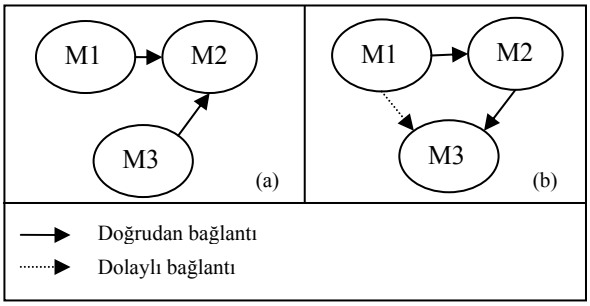
$$Max\_İlişki(X) = \left(\frac{m.(m-1)}{2} \times 1\right) + \left(\frac{m.(m-1)}{2} \times a\right) \quad (12)$$

Yukarıdaki tanımlamalardan yola çıkarak, önerilen ölçüt aşağıdaki gibi formüle edilir:

$$COMIAS = \frac{Total\_İlişki(X)}{Max\_İlişki(X)} \quad (13)$$

3.5. COMIAS Uyum Ölçütünün Örneklenmesi

Bu bölümde Şekil 6'da verilmiş bir örnek üzerinden COMIAS ölçütünün hesaplanması gösterilecektir. Sezgisel olarak bakıldığında Şekil 6.b'de verilmiş olan grafiğin uyumunun, Şekil 6.a'da verilmiş olana göre daha iyi çıkması beklenir. M1 ile M3 arasındaki dolaylı bağlantı hesaba katılmazsa bu sonuç görülemeyecektir.



Şekil 6. Örnek bir sistemin dolaylı ilişkiler hesaplanmadan (a) ve hesaplanarak (b) elde edilen nihai ilişki grafikleri.

Dolaylı bağlantılar hesaba katılmazsa Şekil 6.a'da verilen grafik için COMIAS ölçütü şu şekilde hesaplanır:

Tablo 3. Şekil 6.a örneği için metotlar arasında üye alan paylaşımı ilişkisi

	Metot çağrımı	Üye alan paylaşımı
(M_1, M_2)	1	0
(M_1, M_3)	0	0
(M_2, M_3)	1	0

$$COMIAS = \frac{\left[(1+0)+(0+0)+(1+0)\right]}{\left(\frac{3\times(3-1)}{2}\times 1\right)+\left(\frac{3\times(3-1)}{2}\times 0\right)} = \frac{2}{3} \quad (14)$$

Dolaylı bağlantıların hesaba katılması ile elde edilen Şekil 6.b için ise COMIAS ölçütü aşağıdaki gibi hesaplanır:

Tablo 4. Şekil 6.b örneği için metotlar arasında üye alan paylaşımı ilişkisi

	Metot çağrımı	Üye alan paylaşımı
(M_1, M_2)	1	0
(M_1, M_3)	1	0
(M_2, M_3)	1	0

$$COMIAS = \frac{\left[(1+0)+(1+0)+(1+0)\right]}{\left(\frac{3\times(3-1)}{2}\times 1\right)+\left(\frac{3\times(3-1)}{2}\times 0\right)} = 1 \quad (15)$$

Bu şekilde Şekil 6'da verilen örnek sistemde dolaylı bağlantıların da dikkate alındığı takdirde uyumun daha yüksek çıktığı görülmektedir. Şekil 6.b'de olası tüm bağlantılar kurulu olduğu için uyumun da maksimum çıkması beklenir ki sonuç da bu şekilde elde edilmiştir. Bu örnek aynı zamanda sınıf

bileşenleri arasındaki etkileşim arttıkça uyumun da beklendiği gibi yükseldiğini göstermektedir.

4. Önerilen Ölçütün Deneysel Geçerlemesi

COMIAS uyum ölçütünün deneysel olarak geçerlenmesi amacıyla Zemberek adı verilen açık kaynak kodlu projenin çeşitli sürümleri kullanılmıştır. Zemberek, açık kaynak kodlu Türkçe doğal dil işleme kütüphanesi ve OpenOffice eklentisidir. Zemberek'in Java'da geliştirilmiş olması, kalite açısından aralarında karşılaştırma yapılabilecek farklı sürümlerinin bulunması ve nesne tabanlı olması, bu programın seçilmesinde etkili olmuştur.

COMIAS tarafından üretilen sonuçların gerçek dünya verileriyle karşılaştırılabilmesi için kullanıcı görüşlerine başvurulmuştur. Bu amaçla Zemberek programı geliştirici grubundan olan Ahmet Afşın AKIN ile görüşülmüştür ve şu bilgiler elde edilmiştir:

- Tspell (Zemberek programının eski adı) 0.3 daha çok bir deneme sürümüdür. Sistem sadece belirli işlemleri yüksek sayılmayacak bir başarı ile yapabilmektedir.
- Zemberek 0.6.3 ve 0.6.4 arasında aslında çok fazla fark bulunmamaktadır. Çünkü asıl zıplama Zemberek 0.5 ile Zemberek 0.6 arasında gerçekleşmiştir.
- Zemberek 2.2.1 ile kod olgunluğa ulaşmıştır ve çalışmalar daha çok alt yapıda bazı düzenlemeler ile hata giderimi seklinde olmuştur.

Bu bilgilere dayanarak, Zemberek programı için sürüm sayıları arttıkça genel olarak programın kalitesinin arttığı söylenebilir. COMIAS ölçütünün de bu görüşe paralel sonuçlar vermesi beklenmektedir. Nitekim Tablo 5'de verilen ölçüm sonuçlarının beklentileri karşıladığı görülebilir.

Tablo 5. Zemberek programının ilerleyen sürümleri için COMIAS ölçütü ile elde edilen sonuçlar

Sürüm	0.3	0.6.3	0.6.4	2.2.1
Ölçüm	0,133	0,303	0,310	0,337

5. Sonuçlar

Günümüz insanının çevresinde gelişen ve gittikçe artan yazılım sistemlerinde ortaya çıkabilecek bir yazılım hatası, insan hayatının yok oluşu, mali kayıp veya zaman gecikmesi gibi kötü sonuçlar doğurabilecek güce sahiptir. Bu nedenle, bugünün yazılım sistemleri hatasız ve tutarlı çalışabilen sistemler olarak çalışabilmek zorundadırlar. Artan yazılım kalitesi talebi, "kalite" karakteristiğini ürünler arasında önemli bir ayırıcı faktör olarak karşımıza çıkarmaktadır.

Yazılım uyumu, yazılımın içsel kalite özellikleri arasında bağlaşım ile birlikte önemli bir yere sahiptir ve bir modülün elemanlarının birbirine aitliğinin derecesi olarak tanımlanabilir. İyi tasarlanmış bir sistemde uyumun yüksek ve bağlaşımın düşük olması beklenir. Uyum eksikliği, sınıfların iki veya daha fazla alt sınıfa ayrılmaları gerekliliğini gösterir. Düşük uyum karmaşıklığı arttırarak geliştirme sürecindeki hataların olma olasılığını arttırır. Literatürde önerilen uyum ölçütlerinden hiçbiri bir standart olarak yaygın kabul görmemiştir [38]. Bu duruma yol açan etkenlerden biri de önerilen ölçütler ile ilgili daha fazla deneysel gerçekleme çalışmasına gereksinim duyulmasıdır [6].

COMIAS ölçütü, normalize oluşu, dolaylı ilişkileri de göz önünde bulundurması ve sezgilerle paralel sonuçlar vermesi özellikleri ile ümit vericidir. Ancak COMIAS ölçütü kalıtım ile aktarılan metotları dikkate almamaktadır. Bu nedenlerle COMIAS ölçütünün deneysel geçerlemesi için çalışmaların sürdürülmesi ve kalıtım ilişkisinin de dikkate alındığı yeni bir uyum ölçütü üzerinde çalışılması planlanmaktadır. Ayrıca, ölçüt tanımında metot çağrımı ve üye alan paylaşımı ilişkilerinin ölçümü eşit oranda etkilediği varsayılmaktadır. Bu etkilerin en uygun oranını belirlemek üzere yeni bir çalışma da yapılabilir.

6. Kaynakça

[1] Kitchenham, B. and Pfleeger, S.L., "Software Quality: the Elusive Target", *IEEE Software, vol. 13, no. 1, pp. 12-21,* January 1996.

[2] Baldassari, B., Robach, C. and du Bosquet, L., "Early metrics for Object Oriented Designs", *IEEE*

[3] Stevens, W., Myers, G., and Constantine, L., "Structured Design," *IBM Systems J., vol. 13, no. 2, pp. 115-139*, 1974.

[4] Bunge, M., Treatise on Basic Philosophy: Ontology I *: The Furniture of the World. Boston: Riedel*, 1977.

[5] Lieberherr, K., Holland, I., and Riel, A., "Object oriented programming: An objective sense of style,'' in Third Annu. *ACM Conf: Object Oriented Prog. Syst., Lung. and Applicat. (OOPSLA), 1988, pp. 323-334*.

[6] Briand, L. C., Daly, J. W., and Wüst, J. K., "A Unified Framework for Coupling Measurement in Object-Oriented Systems." *IEEE Transactins on Software Engineering, vol. 25, no. 1,* January/February 1999.

[7] Kriz, J., Facts and Artifacts in Social Science: *An Epistemological and Methodological Analysis of Empirical Social Science Techniques*. McGraw Hill, 1988.

[8] Ebert, C., Dumke, R., "*Software Measurement: Establish - Extract - Evaluate – Execute*", Springer, 2007.

[9] Bansiya, J. and Davis, C. G., "A hierarchical model for object-oriented design quality assessment", *IEEE Transactions on Software Engineering, vol. 28, no. 1*, January 2002, pp. 4-17.

[10] Briand, L. C., Wüst, J., Daly, J. W., and Porter, V. D., "Exploring the relationship between design measures and software quality in object-oriented systems", *Journal of Systems and Software, vol. 51, no. 3, May 2000, pp. 245-273.*

[11] El-Emam, K., "Object-Oriented Metrics: A Review of Theory and Practice", *in Advances in software engineering, Springer-Verlag, New York, 2002, pp. 23-50.*

[12] Quah, T.-S. and Thwin, M. M. T., "Application of neural networks for software quality prediction using object-oriented metrics", *in Proceedings of International Conference on Software Maintenance*, September 22-26 2003, pp. 116-125.

[13] Brito e Abreu, F. and Goulao, M., "Coupling and cohesion as modularization drivers: are we being overpersuaded?" *in Proceedings of 5th European*

Conference on Software Maintenance and Reengineering, 2001, pp. 47-57.

[14] Maletic, J. I. and Marcus, A., "Supporting Program Comprehension Using Semantic and Structural Information", *in Proceedings of 23rd International Conference on Software Engineering, Toronto, Canada*, May 12-19 2001, pp. 103-112.

[15] Etzkorn, L. H. and Davis, C. G., "Automatically Identifying Reusable OO Legacy Code", *IEEE Computer, vol. 30, no. 10, October 1997, pp. 66-72.*

[16] Lee, J. K., Jung, S. J., Kim, S. D., Jang, W. H., and Ham, D. H., "Component identification method with coupling and cohesion*", in Proc. of 8th Asia-Pacific Software Engineering Conference (APSEC'01)*, Dec. 2001, pp. 79-86.

[17] Etzkom L., Delugach H., "Towards a Semantic Metrics Suite for Object-Oriented Design", *IEEE TOOLS (34) 2000: 71-80.*

[18] Allen, E. B., Khoshgoftaar, T. M., and Chen, Y., "Measuring coupling and cohesion of software modules: an information-theory approach", *in Proceedings of 7th International Software Metrics Symposium (METRICS'01)*, April 4-6 2001, pp. 124-134.

[19] Meyers, T. M. and Binkley, D., "Slice-based cohesion metrics and software intervention*", in Proceedings of 11th Working Conference on Reverse Engineering* (WCRE'04), Nov. 8-12 2004, pp. 256-265.

[20] Montes de Oca, C. and Carver, D. L., "Identification of data cohesive subsystems using data mining techniques", *in Proceedings of International Conference on Software Maintenance (ICSM'98)*, November 1998, pp. p. 16-23.

[21] Kramer, S. and Kaindl, H., "Coupling and cohesion metrics for knowledge-based systems using frames and rules", *ACM Trans. on Soft. Engineering and Methodology (TOSEM), vol. 13, no. 3, July 2004, pp. 332-358.*

[22] Cho, E. S., Kim, C. J., Kim, D. D., and Rhew, S. Y., "Static and dynamic metrics for effective object clustering*", in Proceedings of Asia Pacific International Conference on Software Engineering, 1998, pp. 78 - 85.*

[23] Henderson-Sellers, B. *Object-Oriented Metrics Measures of Complexity*, Prentice-Hall, 1996.

[24] Briand, L.C., Daly, J.C. ve Wüst J. "A unified Framework for Cohesion Measurement in Object-Oriented Systems", *Empirical Software Eng.; An Int'l J., vol.3, no. 1, pp. 65-117, 1998.*

[25] Briand, L.C., Daly, J.C. ve Wüst J. "A unified Framework for Cohesion Measurement in Object-Oriented Systems", *Technical Report ISERN-97-05. Kaiserlautern:Germany: Fraunhofer Institute for Experimental Software Engineering, 1997.*

[26] Bonja, C. ve Kidanmariam, E., "Metrics for Class Cohesion and Similarity between Methods", *ACM SE'06, March 10-12, 2006, Melbourne, Florida, USA.*

[27] Zhou, Y., Xu, B., Zhao, J., and Yang, H., "ICBMC: An Improved Cohesion Measure for Classes", *in Proceedings of International Conference on Software Maintenance, October 3-6 2002, pp. 44-53.*

[28] Badri, L. ve Badri, M. "A Proposal of a New Class Cohesion Criterion: An Empirical Study". *Journal of Object Technology, 3(4), Apr. 2004. Special issue: TOOLS USA 2003.*

[29] Chidamber, S. R. ve Kemerer, C. F., "Towards a Metrics Suite for Object Oriented Design", *in Proceedings of OOPSLA'91, 1991, pp. 197-211.*

[30] Chidamber, S. R. and Kemerer, C. F., "A Metrics Suite for Object Oriented Design", *IEEE Transactions on Software Engineering, vol. 20, no. 6, 1994, pp. 476-493.*

[31] Hitz, M. ve Montazeri, B., "Measuring Coupling and Cohesion in Object-Oriented Systems", *Proceedings of International Symposium on Applied Corporate Computing. (Monterrey, Mexico, 1995).*

[32] Bieman, J. ve Kang, B-K., "Cohesion and Reuse in An Object-Oriented System", *in Proceedings of ACM Symposium on Software Reusability (SSR'95), April 1995, pp. 259-262.*

[33] Chae, H. S., Kwon, Y. R. ve Bae, D. H., "A Cohesion Measure for Object-Oriented Classes" *Softw. Pract. Exper., Vol. 30, No. 12. (2000), pp. 1405-1431.*

[34] Jarallah, A., Wasiq, M. ve Ahmed, M. A., "Principle and Metrics for Cohesion-Based Object-Oriented Component Assessment". *Confidential Draft Copy, 2001.*

[35] Aman H., Yamasaki K., Yamada H. ve Noda M., "A Proposal of Class Cohesion Metrics Using Sizes of Cohesive Parts", *Knowledge-based Software Engineering, T. welzer et al.(Eds.), pp102-107, IOS Press, Sept. 2002.*

[36] Gui, G. ve Scott, P.D., "Coupling and Cohesion Measures for Evaluation of Component Reusability", *MSR'06, May 22-23, 2006, Shanghai, China.*

[37] Counsell, S., ve Swift, S., "The Interpretation and Utility of Threee Cohesion Metrics for Object-Oriented Design", *ACM Transactions on Software Engineering and Methodology, Vol. 15, No. 2, April 2006, Pages 123–149.*

[38] Poshyvanyk D., Marcus A., The Conceptual Chesion of Classes. *21st IEEE International Conference on Software Maintenance* (ICSM'05), 2005, pp. 133-142.