



# Modularity

## Modülerlik

Course: Software Architectures and Design  
Ders: Yazılım Mimarisi ve Tasarım

### Kaynaklar:

Structured Techniques for Computing; James Martin, Carma McClure, 1985.  
Fundamentals of Software Architecture ; Mark Richards, Neal Ford ,2020



# Modularity

- Yazılım Mimarısine dair kelimelerin %95'i
  - Modülerliğin faydalarına övgülerdir.
- Peki ya buna nasıl varılabileceğine dair ne var?
  - Hemen hiç bir şey veya çok az şey.
    - *Glenford J. Myers (1978)*
- Cohesive bir modülü parçalamak coupling'i artırır, okunurluğu azaltır.
  - *Larry Constantin*



# Modülerleştirme ?

- Kodların gruplandırılması:
  - Class
  - Function
  - Others
- **Modülerleştirme** için sözlükler ne diyor?
  - Bir dizi standart parça veya ayrı birimden daha girift bir yapı inşa edilmesi.
- Mantıksal bir gruplandırma var.
  - Nesne esaslı çalışmalarında “class”
  - Yapısal veya İşlevsel çalışmalarında “function”
- Bir program modülünün “model”i üç parçadan oluşur:
  - **Input**
  - **Process**
  - **Output**



# Giriftlik

- Program giriftliğini (karmaşıklık) etkileyen faktörler:
  - Modül büyüklüğü
  - Programın modülleri arasındaki ilişki
  - Modülün iç unsurları arasındaki ilişki



# Veri İrtibatı

- Modüller arasında veri geçışı
  - Eğer modüller arasında parametre vasıtasıyla ise
    - (“explicit data connection” (**açık veri irtibatı**))
  - Eğer modüller aynı veriyi referans alıyor ise
    - (“implicit data connection” (**örtük veri irtibatı**))
- Var demektir.



# Veri irtibat karmaşıklığı

- Modüller arası irtibatın giritliği kontrol edilmeli ve azaltılmalıdır.
- Bu irtibatın (bağlantının) irdelenmesi için iki temel ölçü var:
  - Coupling
  - Cohesion



# Karmaşıklık kontrolü

- Parçala ve Yönet taklığı
    - Giriftliği kontrol etmek ve
    - Anlaşılabilirliği artırmak için
  - En iyi yoldur.
- 
- **Giriftlik/Karmaşıklık/Complexity**
    - Modüller bağımsız olmadıkça ve
    - Aralarındaki etkileşim sınırlandırılmadıkça
  - kontrol edilemez.



# Bağımsızlık Ölçüsü

- Modüller arasındaki bağımsızlık düzeyini
  - Coupling
- ile ölçuyoruz.
- Bir modülün öğeleri arasındaki ilişkinin gücünü (seviyesini)
  - Cohesion
- ile ölçuyoruz.



# Modül Kalitesinin İki Ölçütü

- Loosely Coupled (gevşek bağ)
  - (Little interaction between modules)
  - **Tersi:** tightly coupled (kuvvetli bağ)
- Highly Cohesive (yüksek sıkılık-türdeşlik)
  - (strong interaction between module elements).



# **“Coupling”ı etkileyen Unsurlar**

- I.** Modüller arasında geçen veri elemanı sayısı
- 2.** Modüller arasında geçen kontrol verisi miktarı
- 3.** Modüller tarafından kullanılan ortak global veri elemanlarının sayısı



# “Coupling”ı etkileyen Unsurlar

- Modüller arasında geçen veri elemanı sayısı
- Veri elemanı sayısı nedir?
  - Bir modül çağrılarında yer alan parametre veya argüman sayısı.
  - Ör: 100 parametre içeren bir “procedure” çağrısı 1 parametre içерenden daha giriftir(karmaşık).



# “Coupling”ı etkileyen Unsurlar

- Modüller arasında geçen kontrol verisi miktarı
- Kontrol verisi nedir ?
  - Program talimatlarının (instruction) ve modül çağrılarının (call) icra (execution) sırasını yönlendirir.
  - Ör: dosya sonu, hareket kodu v.s.



# “Coupling” Tipleri

- 5 tip:
  1. Data coupling (veri bağlı)
  2. Stamp coupling (damga bağlı)
  3. Control coupling (kontrol bağlı)
  4. Common coupling (genel bağlı)
  5. Content coupling (icerik bağlı)



# Coupling Tipleri

- **Data coupling**

- Two modules are data coupled if they communicate via a variable or array (table) that is passed directly as a parameter between the two modules.
- The data is used in problem-related processing not for program control purposes.
- İki modül birbiriyle doğrudan bir parametre olarak geçen bir **değişken veya dizi (tablo)** aracılığıyla iletişim kurarlarsa, iki modül veri bağlıdır.
- Veriler, program kontrol amaçları için değil **problem ilişkili işlemede** kullanılır.

# Coupling Tipleri

## • Stamp coupling

- Two modules are stamp coupled if they communicate via a **composite data item** (e.g. record).
- The composite data item may contain pieces of data that are not used by a module even though they are passed to it.
- İki modül **bileşik veri ögesi** (örneğin, kayıt) aracılığıyla iletişim kurarlarsa, iki modül damga bağlıdır.
- Bileşik veri ögesi, bir modül tarafından kendisine iletilse bile kullanılmayan veri parçalarını içerebilir.

# Coupling Tipleri

- **Control coupling**
- Two modules are control coupled if data from one **is used to direct the order of instruction execution** in the other (e.g., a flag set in one module and tested in a compare in another module).
- Bir modülden gelen veriler, diğerinin talimat **icra sırasını yönlendirmek** için kullanılıyorsa, iki modül kontrol bağlıdır (örneğin, bir modülde bir “flag” başka bir modülde karşılaştırma için kullanılır).



# Coupling Tipleri

- **Common coupling**
- Two modules are common coupled if they share the same global data areas (e.g., FORTRAN COMMON, PL/I EXTERNAL attribute.)
- İki modül aynı global veri alanlarını paylaşıyorlarsa iki modül ortak bağlıdır.

# Coupling Tipleri

- **Content coupling**
- Two modules are content coupled if one module **refers to or changes** the internals of another module (e.g. a branch or fall through from one module's code into another module).
- Bir modül başka bir module **atıfta bulunuyorsa** veya başka bir modülü **değiştiriyorsa** (örneğin, bir modül kodundan başka bir module'le dallanma veya düşme) iki modül içerik bağlıdır.



# DeCoupling Kılavuzu

- Modülleri bağımsız yapma çalışması.
- *En iyi, program tasarıımı esnasında yapılır.*
- Data coupling'i azaltmak için:
  - Sadece gerekli olan veriyi geçir.
- Stamp coupling'i azaltmak için:
  - Bileşik veri elemanları sadece kullanılacak verileri ihtiva etsin.



# DeCoupling Kılavuzu

- Modülleri bağımsız yapma çalışması.
- Control coupling'i azaltmak için:
  - Mümkin olduğunca sayısını azalt.
- Common coupling'i azaltmak için:
  - Ortak veri alanı kullanmaktan kaçın.
- Content coupling'i hepten kaldır.

# Cohesion

- **Cohesion** measures how strongly the elements within a module are related.
  - Cohesion, bir modül içindeki öğelerin ne kadar güçlü bir bağa sahip olduğunu ölçer.
- *Coincidental* is the weakest and least desirable level; *functional* is the strongest and desirable level.
  - *Coincidental* (*tesadüfi*), en zayıf ve en az arzu edilen seviyedir; *functional* (*işlevsel*) ise en güçlü ve en arzu edilen düzeydir.



# Cohesion

1. Functional
2. Sequential
3. Communicational
4. Procedural
5. Temporal
6. Logical
7. Coincidental

# Cohesion

- **Functional Cohesion**
- Each element in a module is a **necessary and essential** part of **one and only one function**.
- Every part of the module is related to the other, and the module contains everything essential to function.
- **İşlevsel**
- Bir modüldeki her öğe, bir ve yalnızca bir işlevin gerekli ve temel bir parçasıdır.
- Modülün her parçası diğeriley ilişkilidir ve modül, çalışması için gerekli olan her şeyi içerir.
- Example.
- MODULE NETPAY.
  - CALCULATE GROSS.
  - CALCULATE TAXES.
  - CALCULATE NET PAY.

# Cohesion

- **Sequential Cohesion**
- The elements of a module are related by performing different parts of a **sequence** of operations where the **output of one** operation is the **input to the next**.
- **Dayalı sıralı**
- Bir modülün öğeleri, bir işlemin çıkışının diğerinin girdisi olduğu işlemler dizisinin farklı bölümlerinin ifasıyla ilişkilidir.
- Example.
- MODULE *INVENTORYUPDATE*
- GET INVENTORY INPUT.
- CONVERT INVENTORY INPUT.
- UPDATE INVENTORY MASTER.

# Cohesion

- **Communicational Cohesion**
- The elements of a module all **operate the same data**.
- **İletişimsel**
- Bir modülün bütün unsurları aynı verileri kullanır.
- Example.
- MODULE PROCESS TRANSACTION
- READ TRANSACTION.
- EDIT TRANSACTION.
- PROCESS TRANSACTION.

# Cohesion

- **Procedural Cohesion**

- The elements of a module are **all part of a procedure** — a **certain sequence of steps** that have to be done a certain order.

- **Sıralı**

- Bir modülün öğelerinin tümü bir prosedürün parçasıdır - belirli bir sırada atılması gereken belirli bir adımlar dizisi.
- Ex.
- **MODULE LOOP.**
- **DO UNTIL NO MORE ORDERS.**
- **READ ORDER**
- **PROCESS ORDER**
- **END DO**

# Cohesion

- **Temporal Cohesion**

- The elements of a module are **related by time** but **need not occur in a certain order or operate on the same data.**
- The elements of a module are related based on **timing dependencies**. For example, many systems have a list of seemingly unrelated things that must be initialized at system startup; these different tasks are temporally cohesive.

- **Geçici**

- Bir modülün öğeleri zaman bakımından ilişkilidir, ancak belirli bir sırada meydana gelmesi veya aynı veriler üzerinde çalışması gerekmek.
- Bir modülün öğeleri zamanlama bağımlılıklarına dayalı bir ilişkilidir. Örneğin, birçok sistem, sistem açılırken başlatılması gereken görünüşte ilgisiz şeylerin bir listesine sahiptir; bu farklı görevler geçici olarak sıkı ve türdeştir.
- Example.
- INITIALIZE.
- SET COUNTERS.
- ZERO OUT TABLES.
- OPEN FILES.



# Cohesion

- **Logical Cohesion**
- The elements of a module **are all oriented toward** performing a **certain class of operations**.
- The data within modules is **related logically but not functionally**. For example, consider a module that converts information from text, serialized objects, or streams. Operations are related, but the functions are quite different. A common example of this type of cohesion exists in virtually every Java project in the form of the `StringUtils` package: a group of static methods that operate on `String` but are otherwise unrelated.
- Example.
- MODULE *EDIT*
- EDIT TRANSACTION,
- EDIT MASTER.
- EDIT TERMINAL INPUT.

# Cohesion

- **Logical Cohesion**

- The elements of a module **are all oriented toward** performing a certain class of operations.
- The data within modules is **related logically** but **not functionally**. For example, consider a module that converts information from text, serialized objects, or streams. Operations are related, but the functions are quite different.

- **Mantıksal**

- Bir modülün unsurlarının tümü, **belirli bir eylem sınıfını** gerçekleştirmeye yöneliktir.
- Modüller içindeki veriler **mantıksal olarak ilişkilidir, işlevsel olarak değil**.
- Örneğin, bilgileri metinden, serileştirilmiş nesnelerden veya akışlardan dönüştüren bir modül düşünün. Eylemler birbiriyle ilişkilidir, ancak işlevler oldukça farklıdır.



# Cohesion

- **Logical Cohesion**

- Bir modülün unsurlarının tümü, belirli bir eylem sınıfını gerçekleştirmeye yöneliktir.
- Modüller içindeki veriler mantıksal olarak ilişkilidir, işlevsel olarak değil.
- Örneğin, bilgileri metinden, serileştirilmiş nesnelerden veya akışlardan dönüştüren bir modül düşünün. Eylemler birbiriyle ilişkilidir, ancak işlevler oldukça farklıdır.

- **Example.**

- **MODULE EDIT**

- **EDIT TRANSACTION,**
- **EDIT MASTER.**
- **EDIT TERMINAL INPUT.**



# Cohesion

- **Coincidental**
- The elements of a module are essentially **unrelated** by any common function, procedure, data, or anything.
- Elements in a module are not related other than being **in the same source file**; this represents the most negative form of cohesion.
- Example.
- MODULE X.
  - READ TRANSACTION.
  - PROCESS PRINT ERROR.
  - READ OLD MASTER.
  - ZERO COUNTERS.

# Cohesion

- **Coincidental Cohesion**

- The elements of a module are essentially **unrelated** by any common function, procedure, data, or anything.
- Elements in a module are not related other than being **in the same source file**; this represents the most negative form of cohesion.

- **Tesadüfi**

- Bir modülün öğeleri, temelde herhangi bir ortak işlev, prosedür, veri veya herhangi başka bir şeyle ilişkili değildir.
- Bir modüldeki öğelerin, aynı kaynak dosyada olmaktan başka bir ilişkisi yoktur;
- bu, cohesion'ın en olumsuz biçimini temsil eder.
- Example.
- MODULE X.
  - READ TRANSACTION.
  - PROCESS PRINT ERROR.
  - READ OLD MASTER.
  - ZERO COUNTERS.



# Cohesion

1. Functional(işlevsel) Tek işlev
2. Sequential Dayalı sıralı
3. Communicational(iletişimsel) Aynı veri
4. Procedural Adım adım sıralı
5. Temporal(geçici) Aynı zaman
6. Logical(mantıksal) Mantıksal ilişki var,  
işlevsel yok
7. Coincidental(tesadüfi) Ortak unsur yok



# Cohesion

- Seviyeler arasındaki ölçek sürekli, kesikli değildir;
- Çünkü herhangi iki seviyeyi ayıran keskin bir çizgi çizemeyiz.
- Ayrıca, cohesion niceliksel değil niteliksel bir ölçü olduğundan ölçek doğrusaldır.



# “Cohesion” türünün tespiti

- I. If the only way to describe the module's function is using a compound sentence, its level of cohesion is most likely sequential, communicational, or logical.
- I. Modülün işlevini tarif etmenin tek yolu bileşik bir cümle kullanmaksa, modül en büyük ihtimalle
  - “sequential”, “communicational”, veya “logical”dır.



# “Cohesion” türünün tespiti

- 2. If the description of a module's function contains time-oriented words like *first*, *next*, *after*, or *for all*, the module probably has temporal or procedural cohesion.
- 2. Bir modülün işlevinin tarifi, “ilk”, “sonraki,” “sonra” veya “tümü” gibi zamana yönelik kelimeler içeriyorsa, modül muhtemelen
  - “temporal” veya “procedural cohesion”a sahiptir.

# “Cohesion” türünün tespiti

- 3. If the module's function is described as performing some operation for a class of items (e.,g., all types of transaction), the module probably has logical cohesion.
- 3. Modülün işlevi, bir öğe sınıfı için (ör., tüm işlem türleri) bazı işlemleri gerçekleştirmek olarak tarif edilirse, modül muhtemelen
  - “logical cohesion”a sahiptir.



# “Cohesion” türünün tespiti

- 4. If the module's function can be described as "initialization", "clean up", or "housekeeping" the module has temporal cohesion.
- 4. Modülün işlevi "başlatma", "temizleme" veya "yeniden düzenleme" olarak tarif edilebiliyorsa, modülün
  - “temporal cohesion”ı vardır.