

The slide features a background with abstract, overlapping circular shapes in shades of blue, teal, and red. The title 'HUBUNGAN ANTAR CLASS' is centered in a white, sans-serif font. Below the title, the author's name 'Ama Fariza, S.Kom, M.Kom' is displayed in a smaller white font. At the bottom of the slide, there is a dark red horizontal bar containing two small white text elements: 'The document name can go here' on the left and 'Company Proprietary and Confidential' on the right.

HUBUNGAN ANTAR CLASS

Ama Fariza, S.Kom, M.Kom

The document name can go here

Company Proprietary and Confidential

The slide has a light gray background with a subtle, abstract pattern of overlapping shapes. The title 'MATERI' is positioned in the upper left in a bold, black, sans-serif font. The number '2' is located in the upper right corner. A bulleted list is centered on the slide, containing four items: 'Asosiasi', 'Class asosiasi', 'Akhir dari asosiasi', and 'Dependency'. At the bottom left, the word 'COMPANY' is written in a bold, gray, sans-serif font. A dark red horizontal bar is at the very bottom of the slide.

MATERI

2

- Asosiasi
- Class asosiasi
- Akhir dari asosiasi
- Dependency

COMPANY

ASOSIASI

3

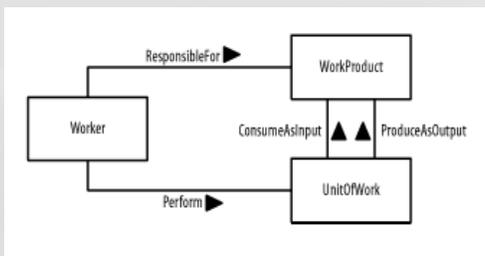
- Asosiasi menyatakan jenis hubungan dan relasi antar class
- Sebagai contoh: **Project Management System** melibatkan beberapa hubungan yaitu **manage, lead, execute, input** dan **output** antara **projects, managers, teams, work products, requirements** dan **sistems**

COMPANY

Asosiasi Biner

4

- Asosiasi biner menghubungkan dua class
- Contoh, satu relasi biner dalam **project management system** adalah antara **worker** dan **units of work**, dan relasi biner lain adalah antara **worker** dan **work products**
- Dalam diagram class, asosiasi biner ditandai dengan garis yang menghubungkan 2 class yang berhubungan dan bisa diberi label nama, dibaca dari kiri ke kanan dan dari atas ke bawah

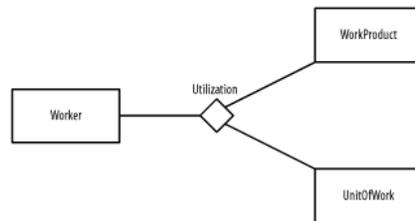


- **Worker** bertanggung jawab untuk **work products** dan membentuk **units of work**
- **Units of work** menggunakan **work products** sebagai input dan menghasilkan **work products** sebagai output.
- Asosiasi biner diberi nama menggunakan kata kerja

Asosiasi N-ary

5

- Asosiasi n-ary menghubungkan 3 class atau lebih
- Contoh: pada **project management sistem**, seorang worker melibatkan semua **worker**, **units of work** dan **work products**
- Asosiasi n-ary menggunakan tanda diamond dengan garis yang terhubung pada semua class dan dapat diberi label nama



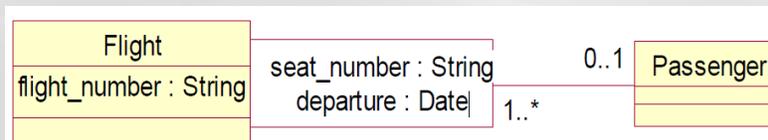
- Asosiasi utilization melibatkan **worker**, **units of work** dan **work products**.
- Nama asosiasi menggunakan kata kerja, tetapi tidak harus, contohnya **utilization** dinyatakan dengan kata benda, karena dilihat dari perspektif kita dimana kita ingin mengerti utilitas pekerja relatif terhadap class lain.
- **Worker** bertanggung jawab untuk **work product** dan **unit of work** nya

COMPANY

Asosiasi Bersyarat (Qualified Association)

6

- Asosiasi bersyarat mempunyai kualifikasi atribut pada salah satu ujung asosiasi biner (asosiasi yang memenuhi syarat dua ujungnya sangat jarang)
- Kualifikasi berisi satu atau lebih atribut yang disajikan dalam bentuk indeks kunci untuk melintasi asosiasi dari class yang memenuhi syarat ke class target pada sisi lainnya

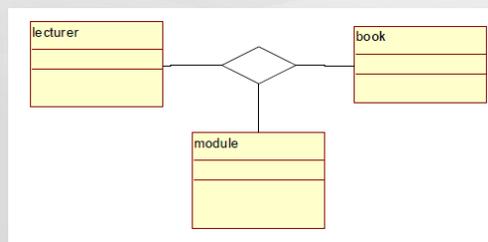


COMPANY

Asosiasi Ternary

7

- Asosiasi yang melibatkan tepat 3 class disebut Asosiasi Ternary.
- Contoh: Terdapat asosiasi antara **Lecturer**, **Book** dan **Module**, artinya terdapat asosiasi yang jelas antara dosen, buku dan modul tertentu
- Misalnya, Dosen Joan menggunakan buku "UML and Spirituality" pada modul "modern theology", artinya kita menentukan kumpulan 3 nilai tersebut (L,B,M), bukan asosiasi yang melibatkan 1 class.

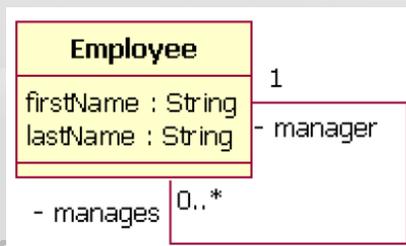


COMPANY

Asosiasi Refleksif (Reflexive associations)

8

- Sebuah class juga dapat diasosiasikan dengan dirinya sendiri, menggunakan asosiasi refleksif.
- Contoh: class **Employee** dapat dihubungkan ke dirinya sendiri menggunakan aturan **manager/manages**
- Jika class diasosiasikan ke dirinya sendiri, bukan berarti anggota class juga relatif ke dirinya sendiri, tetapi anggota class berhubungan dengan anggota lain dari class yang sama



Misalnya anggota dari class **Employee** dapat menjadi **manager** bagi anggota **Employee** lain. Karena aturan relasi "**manages**" mempunyai multiplicity 0..*, seorang **Employee** mungkin tidak mempunyai **Employee** lain untuk di "**manage**"

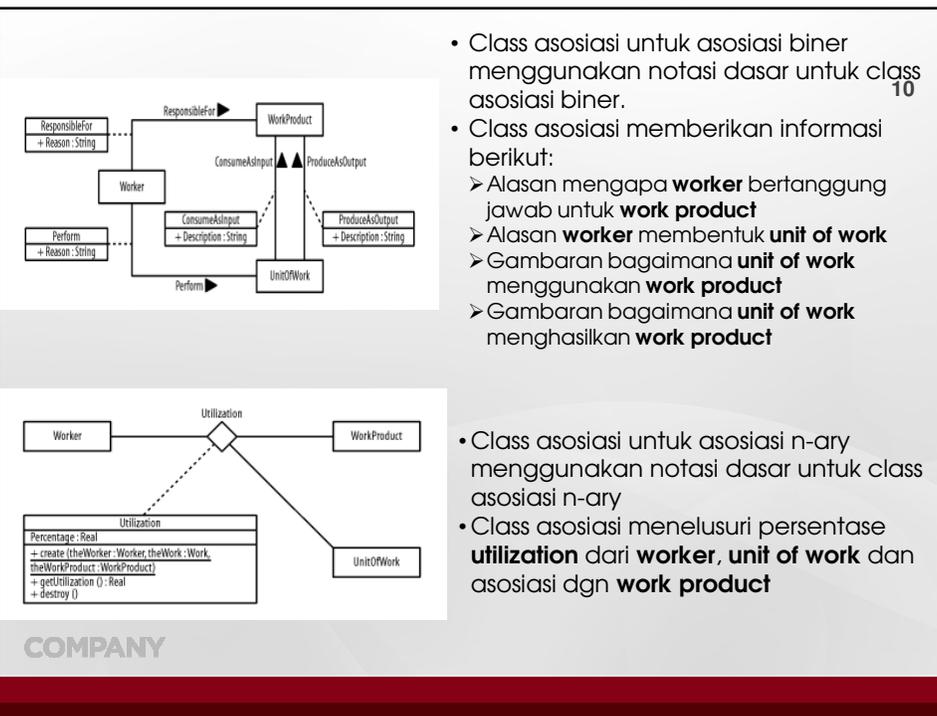
COMPANY

CLASS ASOSIASI

9

- Class asosiasi dapat diaplikasikan menggunakan asosiasi biner dan n-ary
- Sebagaimana class menyatakan karakteristik obyek, termasuk fitur struktural dan perilaku, class asosiasi dapat digunakan untuk menyatakan karakteristik link, baik fitur struktural dan perilaku
- Jenis class ini digunakan bila diperlukan memelihara informasi mengenai hubungan tersebut
- Class asosiasi pada diagram class menggunakan garis putus-putus sebagai asosiasi pada biner atau n-ary. Nama class asosiasi harus sesuai dengan nama asosiasi.

COMPANY



- Class asosiasi untuk asosiasi biner menggunakan notasi dasar untuk class asosiasi biner.
- Class asosiasi memberikan informasi berikut:

- Alasan mengapa **worker** bertanggung jawab untuk **work product**
- Alasan **worker** membentuk **unit of work**
- Gambaran bagaimana **unit of work** menggunakan **work product**
- Gambaran bagaimana **unit of work** menghasilkan **work product**

- Class asosiasi untuk asosiasi n-ary menggunakan notasi dasar untuk class asosiasi n-ary
- Class asosiasi menelusuri persentase **utilization** dari **worker**, **unit of work** dan asosiasi dgn **work product**

COMPANY

AKHIR ASOSIASI

11

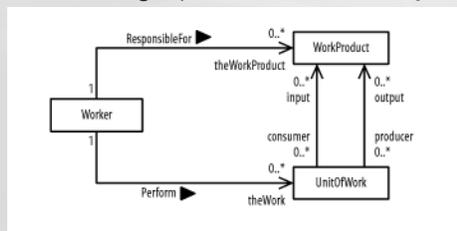
- Akhir asosiasi adalah titik akhir dari garis yang digambar untuk suatu asosiasi dan dihubungkan dengan asosiasi pada sebuah class
- Akhir asosiasi melibatkan beberapa hal untuk menyatakan lebih detail tentang bagaimana class berhubungan ke class lain atau class dalam asosiasi yaitu:
 - Rolename (nama pemeran)
 - Panah navigasi
 - Spesifikasi multiplicity
 - Simbul agregasi atau komposisi
 - kualifikasi

COMPANY

Rolename (Nama Pemeran)

12

- Rolename bersifat opsional dan menyatakan aturan dalam class relatif ke class dalam suatu asosiasi, bagaimana class lain "melihat" class tsb atau apa "wajah" proyek class terhadap class lain dalam relasi. Rolename ditampilkan pada akhir asosiasi dari class.
- Contoh:
 - **work product** dilihat sebagai **input** oleh **unit of work** dimana **unit of work** dilihat sebagai konsumen oleh **work product**
 - **work product** dilihat sebagai **output** oleh **unit of work** dimana **unit of work** dilihat sebagai produsen oleh **work product**

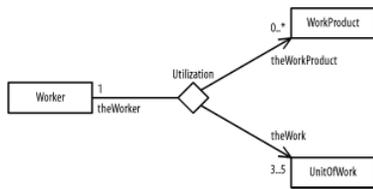


COMPANY

Navigasi

13

- Navigasi bersifat opsional dan menyatakan apakah class direferensi dari class lain dalam asosiasi
- Menggunakan panah yang menuju ke akhir asosiasi menunjuk ke class. Jika tidak ada panah, asosiasi diasumsikan untuk dua arah dan semua class terlibat dalam asosiasi mungkin mereferensi class lainnya



COMPANY

- Contoh: **worker**, ditentukan **work product** dan **units of work**. Panah yang menunjuk ke **work product** dan **units of work**. **Unit of work** menentukan **input** dan **output work product**; tetapi pada **work product**, tidak dapat mengidentifikasi **worker** yang bertanggungjawab atau **units of work** yang mereferensi sebagai **input** atau **output**.
- Navigasi diaplikasikan ke asosiasi n-ary. Kita dapat mereferensi **work product** dan **units of work** dari **worker** untuk menentukan **utilization** nya, tetapi **work product** atau **unit of work** tidak dapat menentukan **utilization** dari **worker**

Multiplicity # 1

14

- Multiplicity (opsional) menyatakan berapa obyek dari class yang mungkin berhubungan ke class lain dalam asosiasi
- Multiplicity menggunakan
 - Interval integer
 - Nilai integer literal
- Interval menggunakan tanda *batas bawah* .. *batas atas* yang menyatakan jangkauan. Contoh 1 berarti satu, 1..5 berarti satu ke lima, 1,4 berarti satu atau empat, 0..* dan * berarti satu atau lebih (atau many), dan 0..1 dan 0,1 berarti nol atau satu
- Tidak ada multiplicity default untuk akhir asosiasi.
- Multiplicity harus dispesifikasikan, contohnya:
 - Satu **worker** bertanggung jawab untuk nol atau lebih **work product**
 - Satu **work product** bertanggung jawab tepat satu **worker**
 - Satu **worker** membentuk satu atau lebih **unit of work**
 - Satu **unit of work** dibentuk tepat satu **worker**
 - Satu **unit of work** sebagai **input** consumer nol atau lebih **work product** dan sebagai **output** produser nol atau lebih **work product**
 - Satu **work product** mungkin dikonsumsi sebagai **input** dengan nol atau lebih **units of work** dan diproduksi sebagai **output** dengan nol atau lebih **units of work**

COMPAN

Multiplicity #2

15

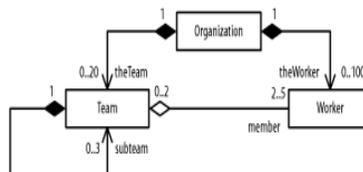
- Untuk menentukan multiplicity dari class, buatlah pertanyaan berapa banyak obyek yang mungkin berhubungan pada satu obyek dari class → jawaban menentukan multiplicity pada akhir asosiasi
- Contoh: Pada satu obyek **worker**, berapa banyak **work product** yang menjadi tanggung jawabnya? Jawaban: nol atau lebih
- Cara lain menentukan multiplicity dengan menanyakan berapa banyak obyek dari class yang berhubungan ke satu obyek dari class pada akhir asosiasi, atau ke satu obyek dari setiap class ke akhir asosiasi n-ary lain → jawaban menentukan multiplicity untuk class
- Contoh:
 - berapa banyak **work product** yang menjadi tanggung jawab **worker**? Jawaban: nol atau lebih
 - Berapa banyak **work product** berhubungan dgn satu **worker** dan satu **unit of work** untuk menentukan **utilization** dari **worker**? Jawaban: nol atau lebih.

COMPANY

Aggregasi

16

- Agregasi adalah “**whole-part relationship**” antara agregasi, keseluruhan dan sebagian
- Relasi ini biasanya disebut ‘**has-a relationship**’, karena keseluruhan mempunyai beberapa bagian
- Contoh: **worker** membuat sebuah **team**, dapat dikatakan satu **team** mempunyai **worker**
- Agregasi hanya digunakan pada asosiasi biner.
- Terdapat relasi circular untuk memungkinkan terbentuknya sub tim. Relasi circular disebut relasi refleksif, karena menghubungkan 2 obyek dari class yang sama

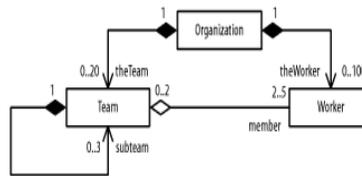


COMPANY

Komposisi

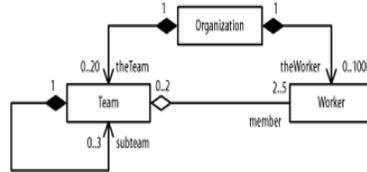
17

- Komposisi, juga disebut agregasi komposit, adalah **'whole-part relationship'** antara komposit (keseluruhan) dan sebagian, dimana sebagian hanya menjadi milik satu keseluruhan dan keseluruhan bertanggung jawab untuk membuat dan menghapus sebagian jika dibuat dan dihapus
- Relasi ini disebut **'contains-a relationship'**, karena keseluruhan berisi sebagian.
- Contoh: organisasi berisi **team** dan **worker**, dan jika **organization** ada, **team** dan **worker** juga ada. Individu tertentu yang merepresentasikan **worker** juga masih ada, tetapi mereka tidak lagi menjadi **worker** dari **organization**, karena organisasi tidak ada lagi.



COMPANY

18



- **Organization** bisa terdiri dari 0 s/d 20 **team** dan 0 s/d 1000 **worker**. Setiap **team** mempunyai 2 s/d 5 **worker** dan setiap **worker** mempunyai anggota 0 s/d 2 **team**. Sebagai tambahan, sebuah **team** terdiri dari 0 s/d 3 **subteam**.
- Untuk menentukan menggunakan agregasi atau komposisi gunakan pertanyaan
 - Jika class berhubungan satu dengan lainnya gunakan asosiasi
 - Jika satu class adalah bagian dari class lain, yang merupakan keseluruhan, gunakan agregasi, sebaliknya gunakan asosiasi
 - Jika sebagian adalah milik dari hanya satu keseluruhan dan keseluruhan bertanggung jawab untuk membuat dan menghapus sebagian nya, gunakan komposisi, sebaliknya gunakan agregasi
- Contoh: **worker** adalah bagian dari **team** dan **organization**, **team** adalah bagian dari **organization** dan **subteam** adalah bagian dari **team**

COMPANY

19

- **Team** dan **worker** adalah milik satu **organization** dan **organization** bertanggung jawab untuk menentukan (membuat dan menghapus) **team** dan **worker** nya.
- **Subteam** harus menjadi milik satu **team** saja dan **team** bertanggung jawab untuk menentukan (membuat atau menghapus) **subteam**
- Apabila hal ini tidak jelas gunakan asosiasi tanpa agregasi dan komposisi

- Komposisi juga dapat berupa class nesting, dimana multiplicity dari class ditunjukkan pada pojok kanan atas dari nama class dan rolenama ditunjukkan di depan nama class
- Membedakan rolenama dari nama class menggunakan tanda ':'
- Contoh:
 - ✓ nesting antara **team** dan **worker** dalam organization untuk mengkomunikasikan informasi yang sama
 - ✓ Nesting dari **subteam** dalam **team** untuk mengkomunikasikan informasi

DEPENDENCY

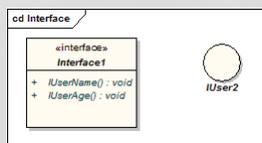
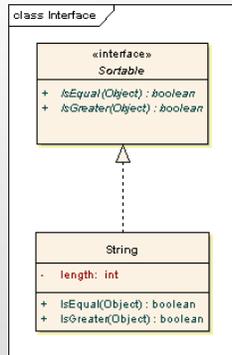
- Terkadang relasi antara dua class sangat '**weak**'. Class tersebut tidak diimplementasi dengan variabel, tetapi sebagai argumen fungsi.
- Sebagai contoh, fungsi **Draw** dari class **Shape**, mengambil argumen dari **DrawingContext**. **Shape** bergantung pada **DrawingContext**. Dalam C++ hal ini sama dengan #include

20

COMPANY

Interface

21



- Interface adalah perilaku khusus yang mengimplementasikan kontrak
- Dengan merealisasikan interface, class dijamin untuk mendukung perilaku yang dibutuhkan, artinya mengijinkan sistem untuk memperlakukan elemen yang tidak berhubungan dengan cara yang sama melalui interface
- Interface dapat digambarkan dengan cara yang sama dgn class, dengan operasi tertentu
- Juga dapat digambarkan sebagai lingkaran tanpa operasi yang ditulis eksplisit. Bila menggunakan lingkaran, garis realisasi digambar tanpa panah

COMPANY