PAIK6401 / AIK21341
Pemrograman Berorientasi Objek

13 Modularisasi

Informatika FSM Universitas Diponegoro

PAIK6401 / AIK21341 Pemrograman Berorientasi Objek

13 Modularisasi

Informatika FSM Universitas Diponegoro

Capaian Pembelajaran

CPL03

Mampu menerapkan ilmu dan teknologi informasi dalam proses penyelesaian permasalahan, yang meliputi analisis permasalahan kompleks, pemodelan, pendefinisian kebutuhan, perancangan, implementasi dan evaluasi terhadap sistem, proses, komponen, dan program.

CPL05

Mampu menghasilkan rancangan, mengimplementasikan, dan mengevaluasi solusi berbasis algoritma dengan mempertimbangkan aspek kompleksitas

CPL07

Mampu memilih, mengadaptasi atau membuat, kemudian menerapkan teknik, sumber daya, kakas komputasi moderen dengan tepat pada aktivitas komputasi kompleks, serta memahami keterbatasannya.

Capaian Pembelajaran

CPMK05-2:

Mampu menerapkan konsep teoretis bidang pengetahuan dan keterampilan Ilmu Komputer dalam menyelesaikan permasalahan kompleks dengan pemikiran komputasional untuk pengambilan keputusan.

CPMK10-2:

Mampu menghasilkan rancangan dan Mengimplementasi solusi berbasis algoritma untuk permasalahan kompleks.

Capaian Pembelajaran

Sub CPMK05-2 dan Sub CPMK10-2:

1. Mampu menerapkan (C3) konsep enkapsulasi, kelas, dan algoritma siklus hidup objek dengan mendemonstrasikan (P3) dalam bahasa pemrograman tertentu.
2. Mampu menerapkan (C3) konsep dan konsekuensi pewarisan dengan mengkonstruksi (P4) kelas dalam bahasa pemrograman tertentu.
3. Mampu menganalisis (C4) polimorfisme dan generik dengan mengembangkan (P4) kasus dalam bahasa pemrograman tertentu.
4. Mampu mendesain (C6) koleksi objek persisten dengan mendemonstrasikan (P3) penyelesaian permasalahan kompleks.
5. Mampu memadukan (C6) prinsip rancangan berorientasi objek dengan paradigma lain yang relevan.

Bahan Kajian

1. Objek, Kelas, Enkapsulasi, dan Information Hiding
2. Inheritance, Overloading, Overriding, Kelas Abstrak dan Interface, Eksepsi dan Asersi
3. Polimorfisme dan Generik
4. Koleksi Objek Persisten
5. Desain Berorientasi Objek, Multiparadigma

Modularisasi

- Sistem perangkat lunak didekomposisi menjadi komponen-komponen yang memiliki tanggung jawab khusus dan saling berinteraksi.
- Mekanisme interaksi: pemanggilan suatu operator kepunyaan Objek lain, maka operator tersebut harus terdaftar dalam Objek tersebut.
- Sistem yang modular memudahkan perawatan komponennya.

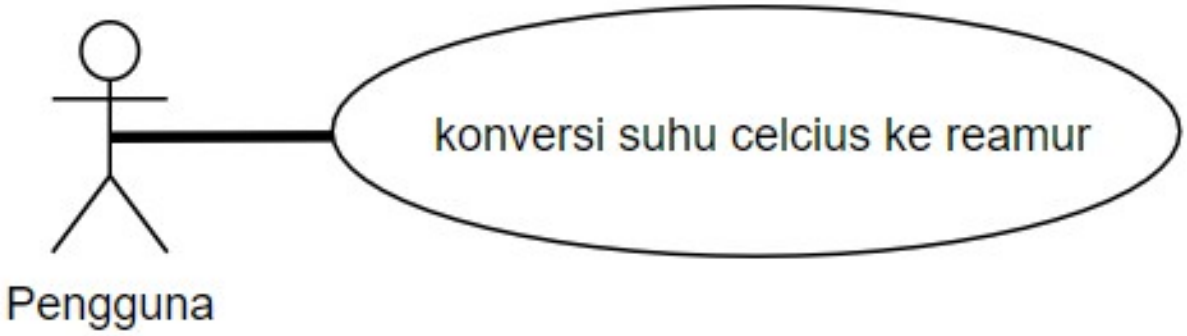
Contoh Modularisasi MVC

- Model-View-Controller
- Komponen Model bertanggung jawab atas penyediaan data, baik kalkulasi maupun dari memori sekunder
- Komponen View bertanggung jawab atas interaksi dengan pengguna, baik manusia maupun perangkat lunak atau keras lain.
- Komponen Controller bertanggung jawab mengendalikan komponen lain, baik Model, View, maupun Controller lain.

Contoh Kasus MVC

- Sebuah program CelciusToReamur menerima input angka derajat Celcius dari papan kunci kemudian menghitung kesetaraan derajat Reamur dan menampilkannya ke layar.
- Rumus: suhu Reamur = (4/5) * suhu Celcius

Analisis Kasus MVC



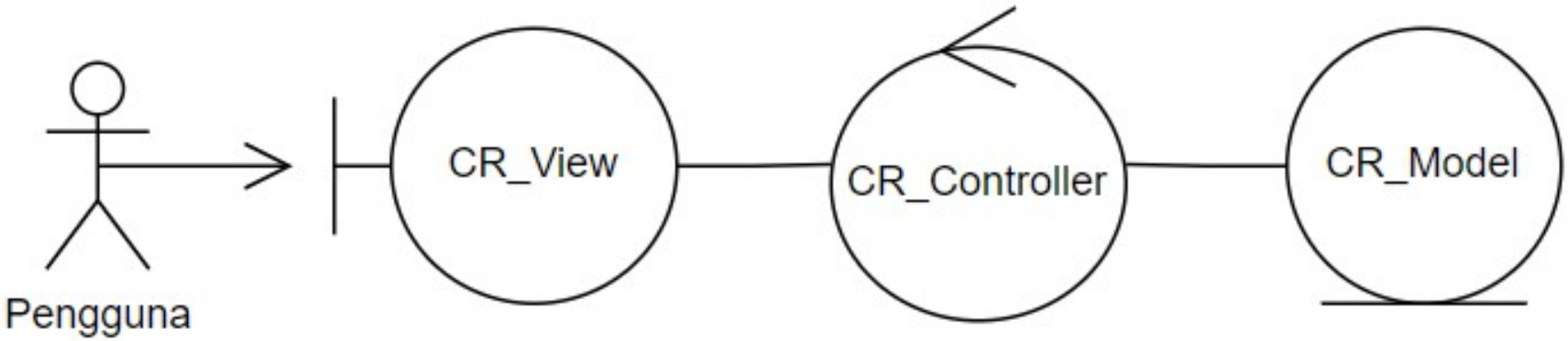
Tugas Pengguna:

1. memasukkan angka suhu Celcius
2. mengklik tombol enter
3. melihat hasil konversi suhu Reamur

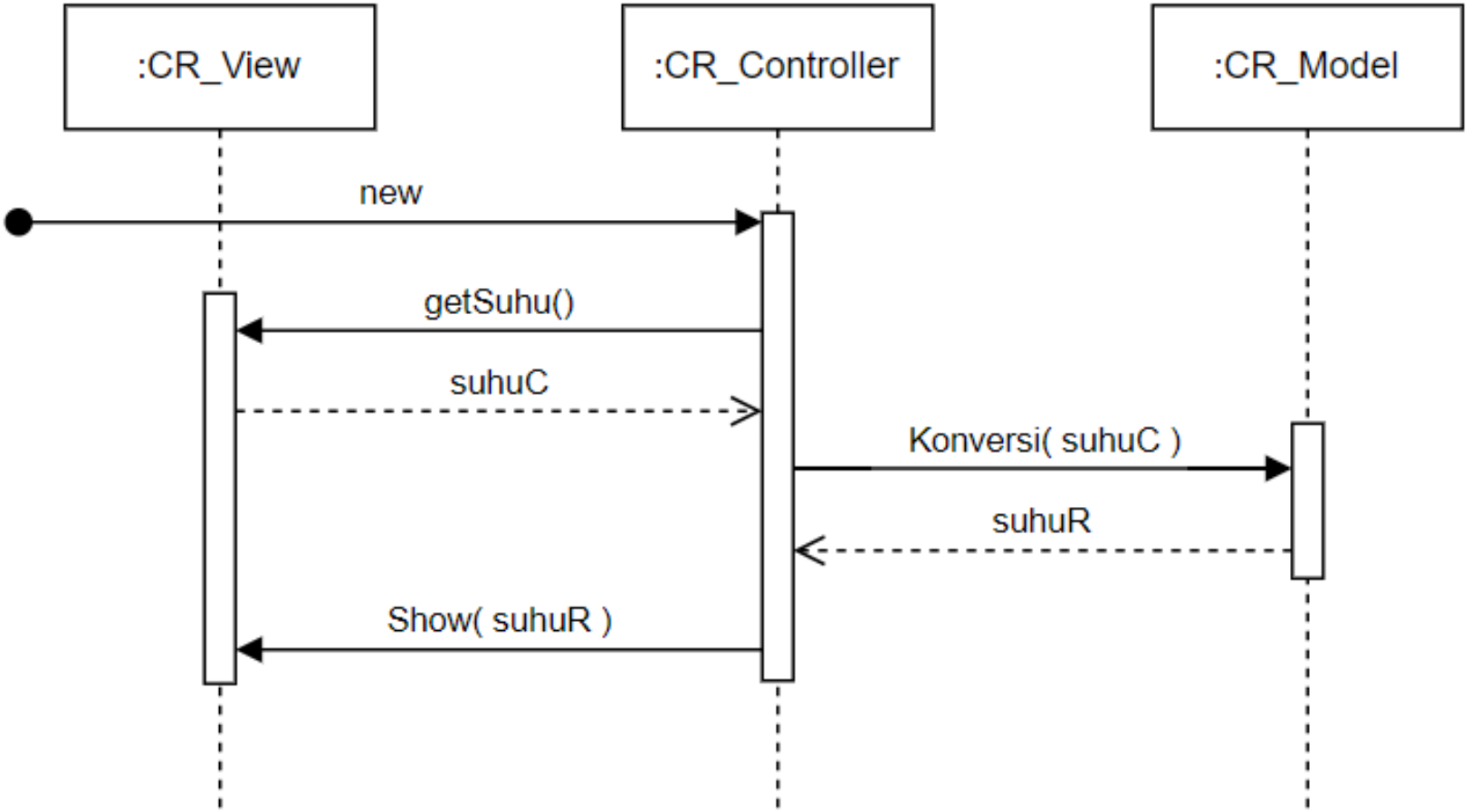
Analisis Kasus MVC

- Program didekomposisi menjadi 3 komponen:
- Komponen CR_Model bertugas menghitung suhu Reamur
- Komponen CR_View bertugas menerima input suhu Celcius, dan menampilkan kesetaraan suhu Reamur ke layar
- Komponen CR_Controller bertugas mengatur interaksi komponen View dan Model.

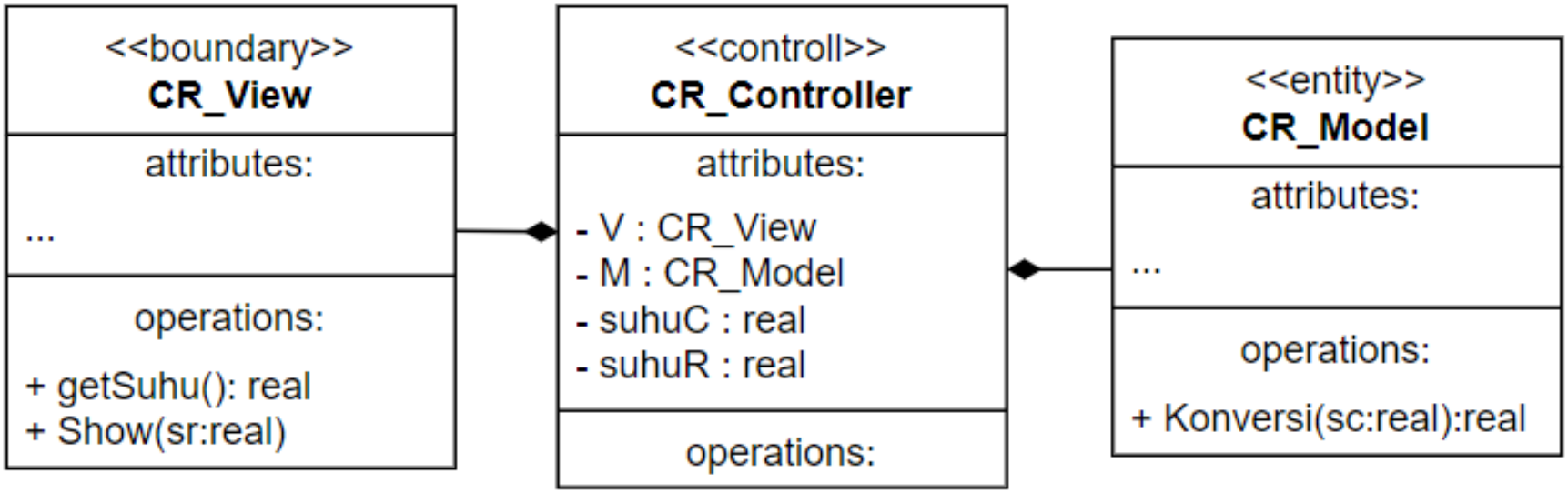
Analisis Kasus MVC

- 

Desain Sekuens



Desain Kelas



Implementasi CR_Model

```
class CR_Model has
   function Konversi( sc: real ) -> real
   {mengembalikan kesetaraan suhu Celcius ke Reamur}
        -> sc * 0.8 ;
```

Implementasi CR_View

```
class CR_View has
   function getSuhu() -> real
   {mengembalikan angka suhu Celcius}
        cc : real
        input cc
        -> cc
   procedure Show( sr : real  )
   {menampilkan angka suhu reamur ke layar}
        output sr
```

Implementasi CR_Controller

```
class CR_Controller has
  V : CR_View   {objek tampilan}
  M : CR_Model  {objek data}
  suhuC : real  {suhu Celcius}
  suhuR : real  {suhu Reamur}
  constructor CR_Controller()
      M <- new CR_Model()
      V <- new CR_View()
      suhuC <- V.getSuhu()
      suhuR <- M.Konversi( suhuC )
      V.Show( suhuR )
```

Program Utama

-

```
class Main has
   constructor Main
      kamus
       C : CR_Controller
      algoritma
       C <- new CR_Controller()
```

Referensi

1. Panji Wisnu Wirawan, Indra Waspada, Satriyo Adhy. 2018. Buku Ajar Pemrograman Berorientasi Objek.
2. Herbert Schildt. 2019. Java The Complete Reference, 11th edition.
3. Vaskaran Sarcar. 2020. Interactive Object-Oriented Programming in Java.
4. Robert C. Martin. 2013. Agile Software Development, Principles, Patterns, and Practices, 1st edition.
5. Peter Sestoft. 2017. Programming Language Concepts, 2nd edition.