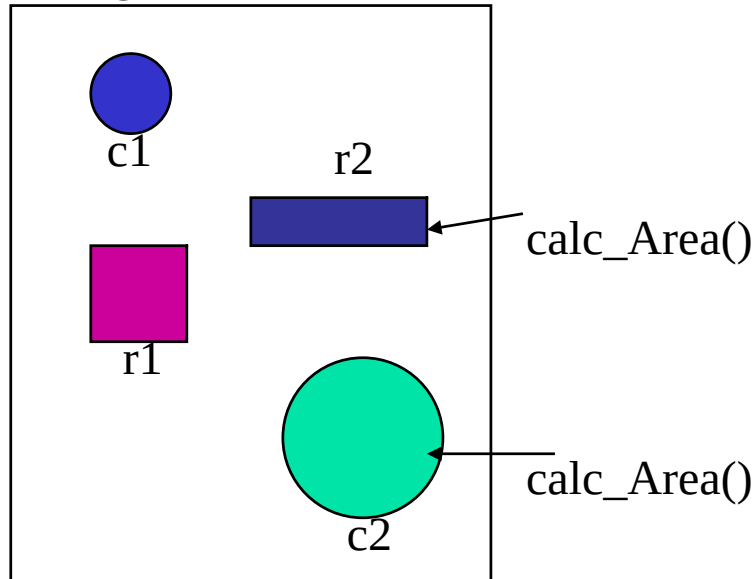


プログラムの拡張

ある領域にあるRectangleとCircleの面積の総和

Region



```
public class Region {  
    Rectangle[] r = new Rectangle[10];  
    Circle[] c=new Circle[10];
```

```
    public double sum_of_Area() {
```

```
        return sum ;
```

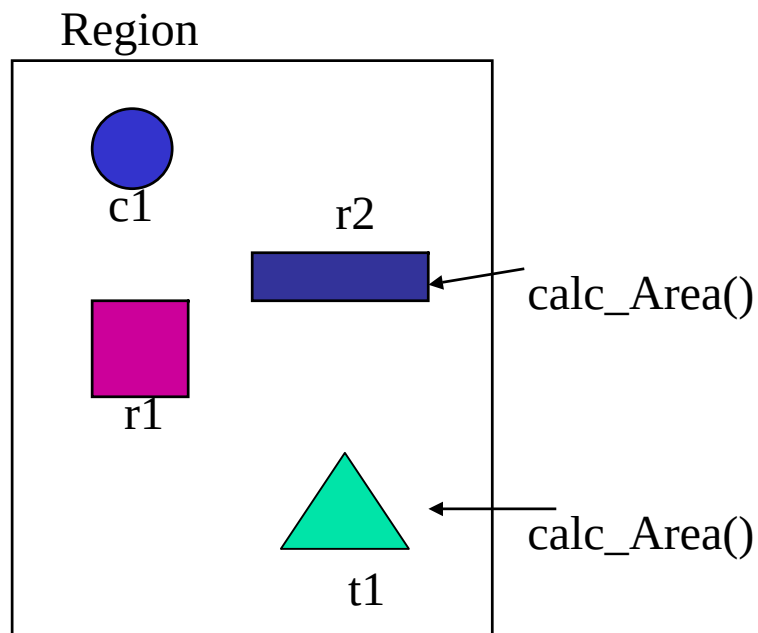
```
    }
```

```
}
```

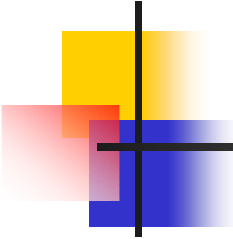
問題点



- 他の種類(例えば, 三角形など)の図形を追加したとき, 配列宣言と総和計算を追加する必要有



```
public class Region {  
    Rectangle[] r = new Rectangle[10];  
    Circle[] c=new Circle[10];  
    Triangle t = new Triangle[10];  
  
    public double sum_of_Area() {  
  
  
  
        return sum ;  
    }  
}
```



解決法

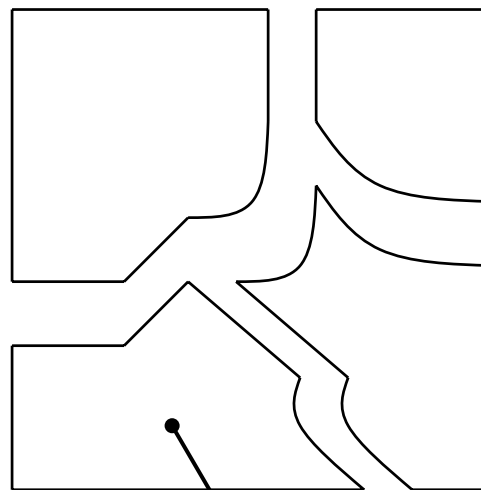
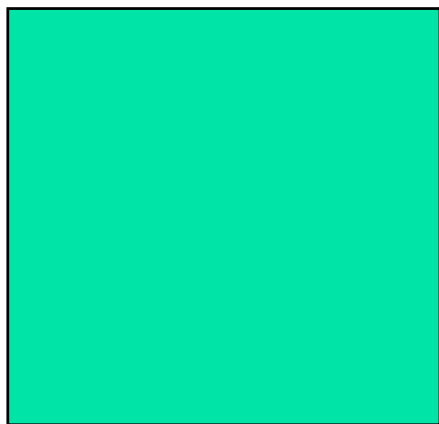
■ 図形クラスの

を作る

```
public class Figure {  
    ...  
}  
  
public class Rectangle  
    {  
    ...  
}  
....
```

```
public class Region {  
  
    public double sum_of_Area() {  
        double sum = 0 ;  
        /* 計算 */  
        return sum ;  
    }  
}
```

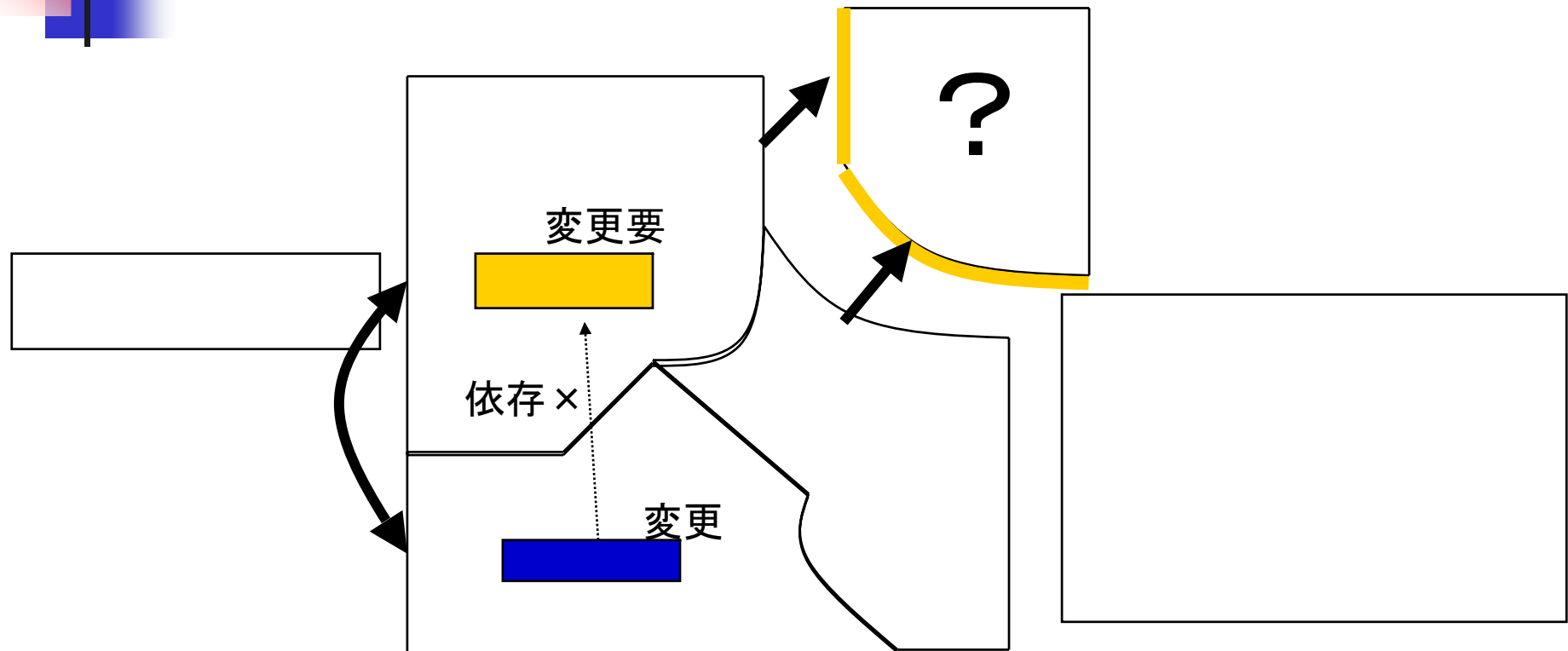
設計(1)

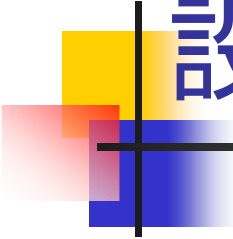


部分問題に分割
モジュール化

モジュール

設計(2)

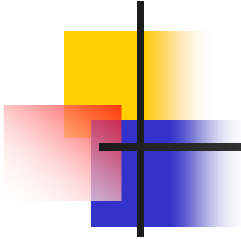




設計(3)

- 要求を実現するにはどのようなモジュール必要か. それらをどう組み合わせるか.
- モジュール: ソフトウェア中のまとまりを持った部分, 部品. (関数, サブルーチンなど)
- 得られるもの





モジュールへの分解

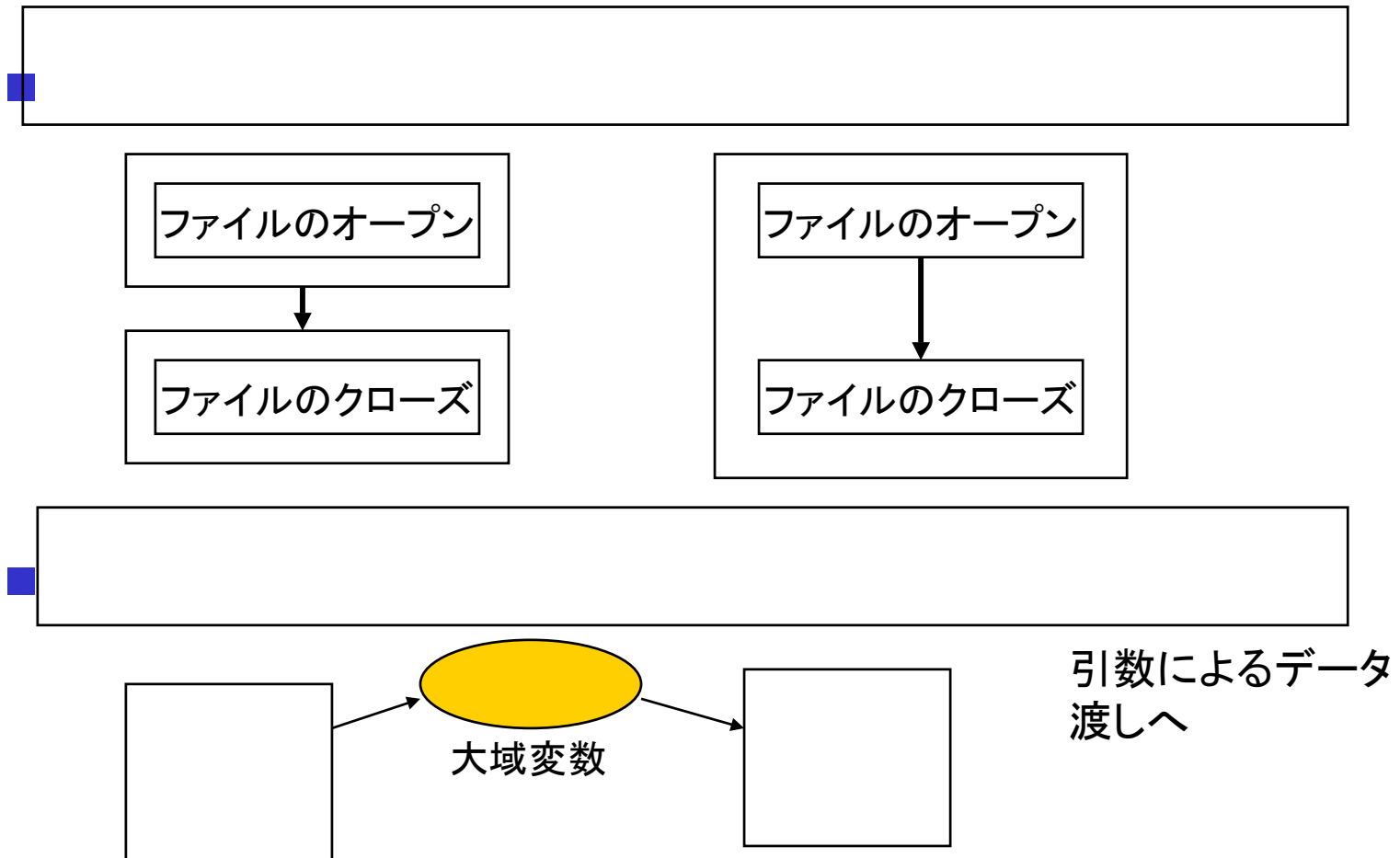
大規模システムの設計には、モジュール分解が不可欠

ではどうやって分解する？

隠蔽性
独立性 } 高める

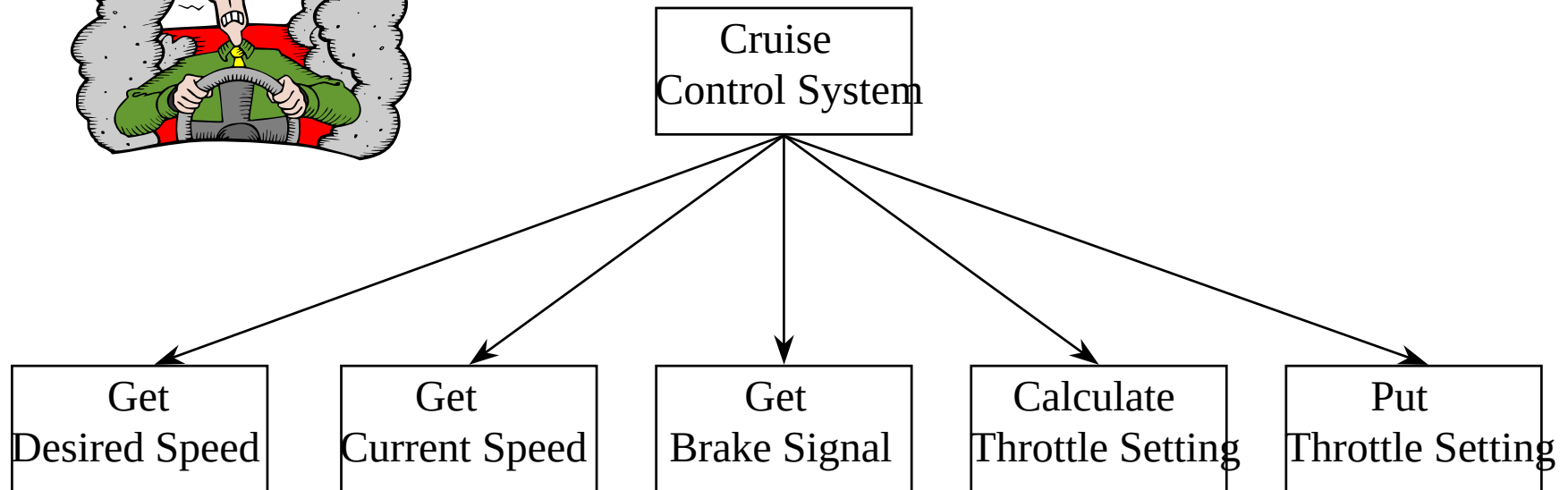
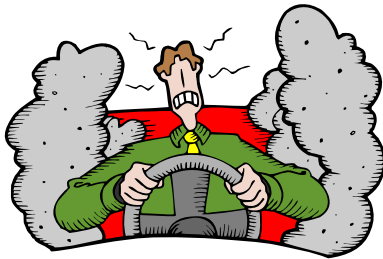


モジュールの独立性



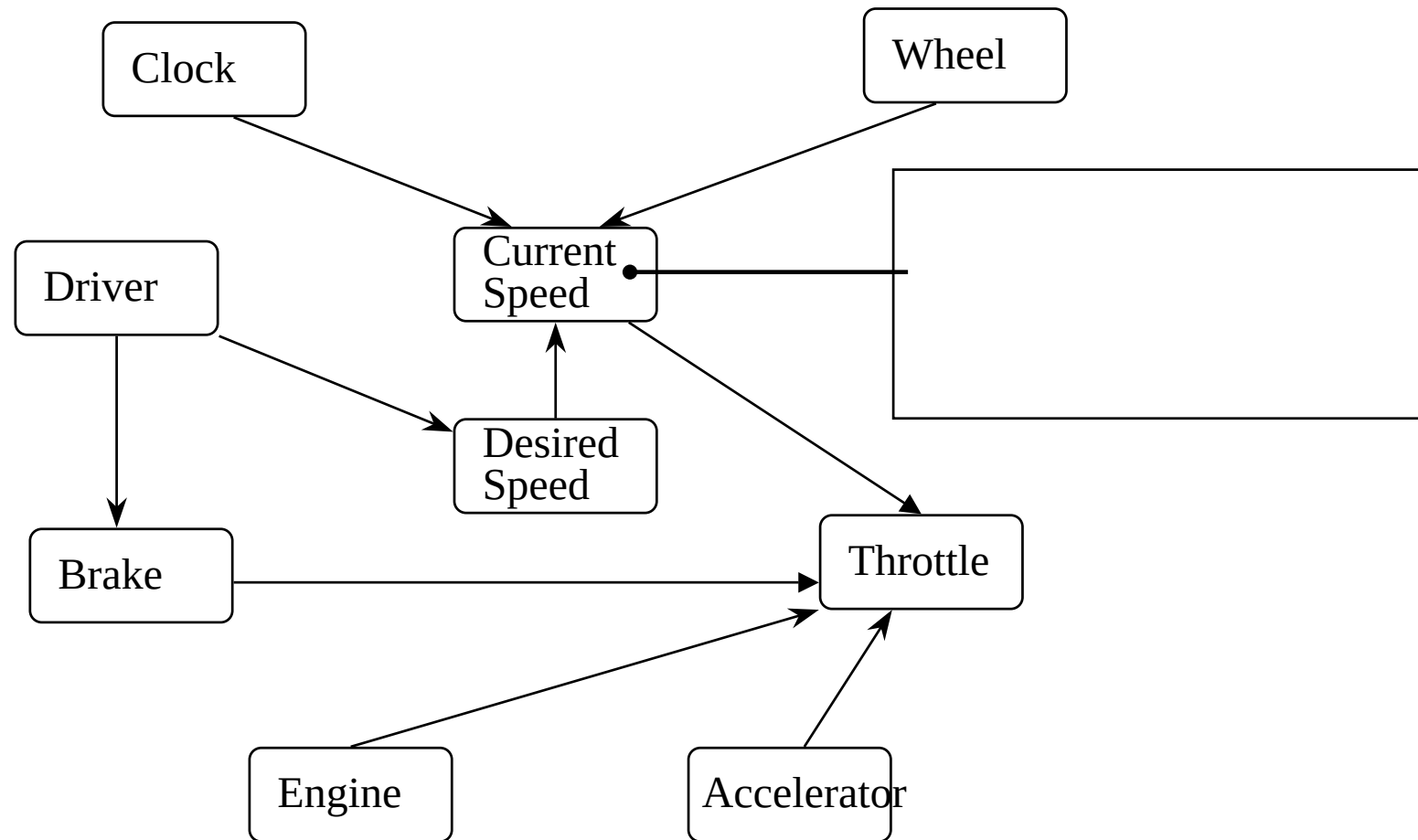
機能分解の例

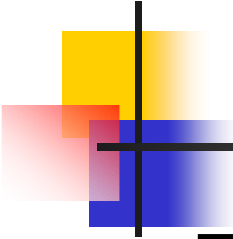
設定したスピードに保つようにスロットルを制御する



オブジェクト指向分解の例

オブジェクト(もの)が独自の計算能力を持っていると考える





オブジェクト(1)

オブジェクトとは？

- 固有の属性(内部状態)を持つ
- 他のオブジェクトと関係を持つ
- uniqueな名前を持つ
- ある共通の性質を持つものをまとめて1つのグループとすることができる：



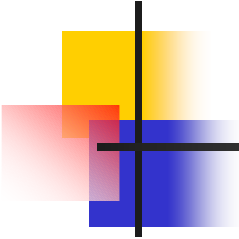


オブジェクト(2)

オブジェクトとは？

- 固有の操作を持ち、この操作のみが許される：
カプセル化，情報隠蔽
 - 属性値の参照
 - 属性値の変更
 - オブジェクトの生成・消滅
- 操作名が同じでも異なるクラスのオブジェクトに適用されれば，効果は異なる

- メッセージ通信による計算



クラス階層

- スーパークラス, サブクラス
 - サブクラスの元はスーパークラスの元
 - スーパークラスが持っている性質はサブクラスに引き継がれる

-
- 属性や操作を共通に持つことができる
 - 差分(異なる部分)を記述すればよい