Module 2:
"SOLID in Practice"

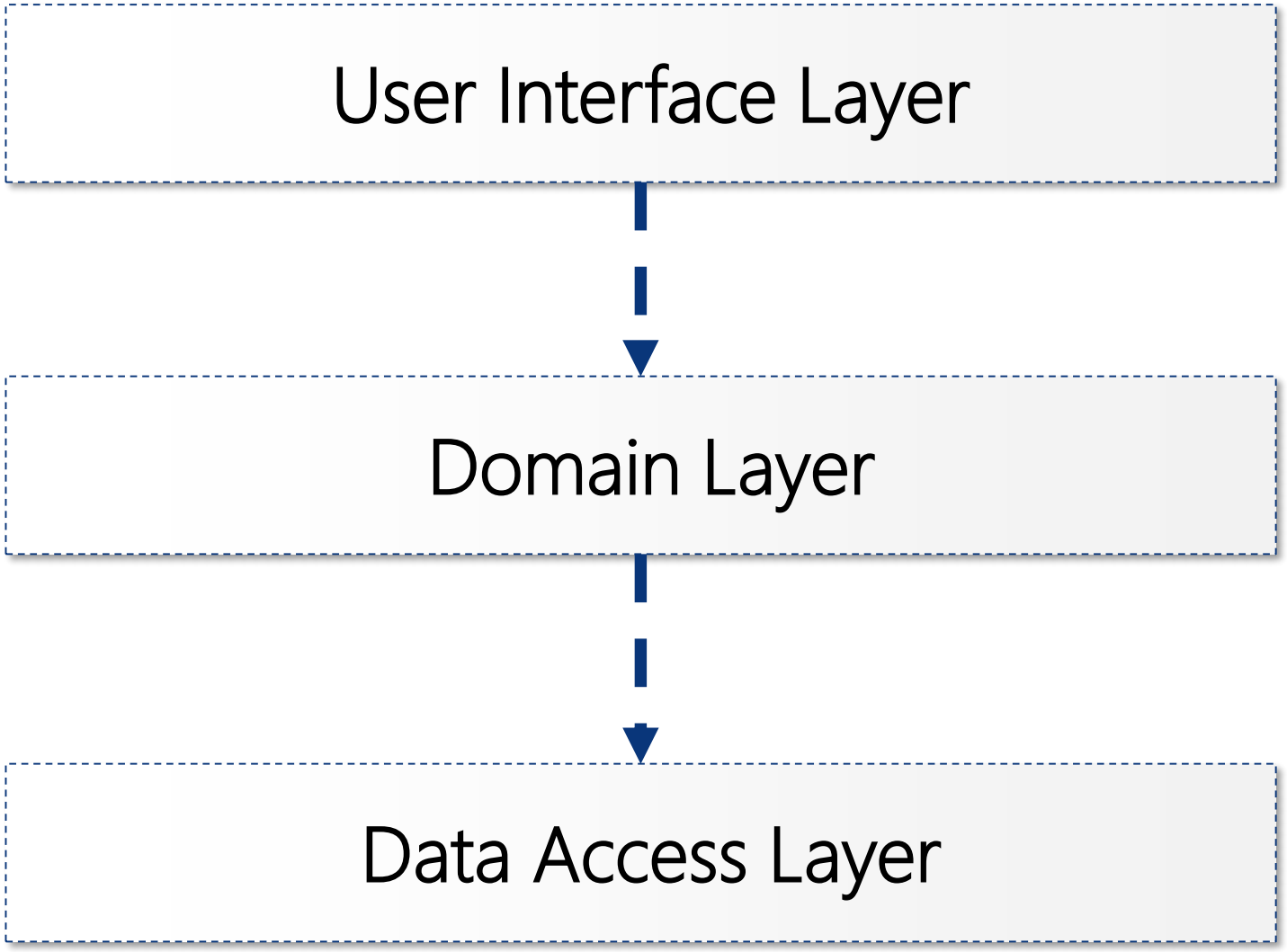
TEKNOLOGISK
INSTITUT

TEKNOLOGISK INSTITUT

Agenda

- **Discussion: Evaluating the Design**
- *Pattern: Repository (with Entity Framework)*
- Workshop A.2: Data Access Layer with Repository
- Discussion: Evaluating the Design Again
- *Automatic Testing*
- Workshop A.3: Modify the Data Access Layer

Beautiful Layered Design?



Discussion: Evaluating the Design

- Can we change the UI Layer from Console to e.g. Web or WPF?
- Can we unit test the Domain Layer?
- Can we change the Data Access Layer?

Anti-Pattern: Entourage

- *When A depends upon B, and you group B and C in the same assembly, then if C depends upon D, in effect, you have equipped A with a dependency upon D.*

- Outline
  - If you keep the interfaces and implementations in the same assembly, you essentially inherit dependencies' dependencies.
  - Entourage means ask for one assembly and it gives all it assemblies.
  - Nuget packages are potentially evil!

- See:
  "Adaptive Code" (2nd Edition)
  Gary McLean Hall (2017)

Agenda

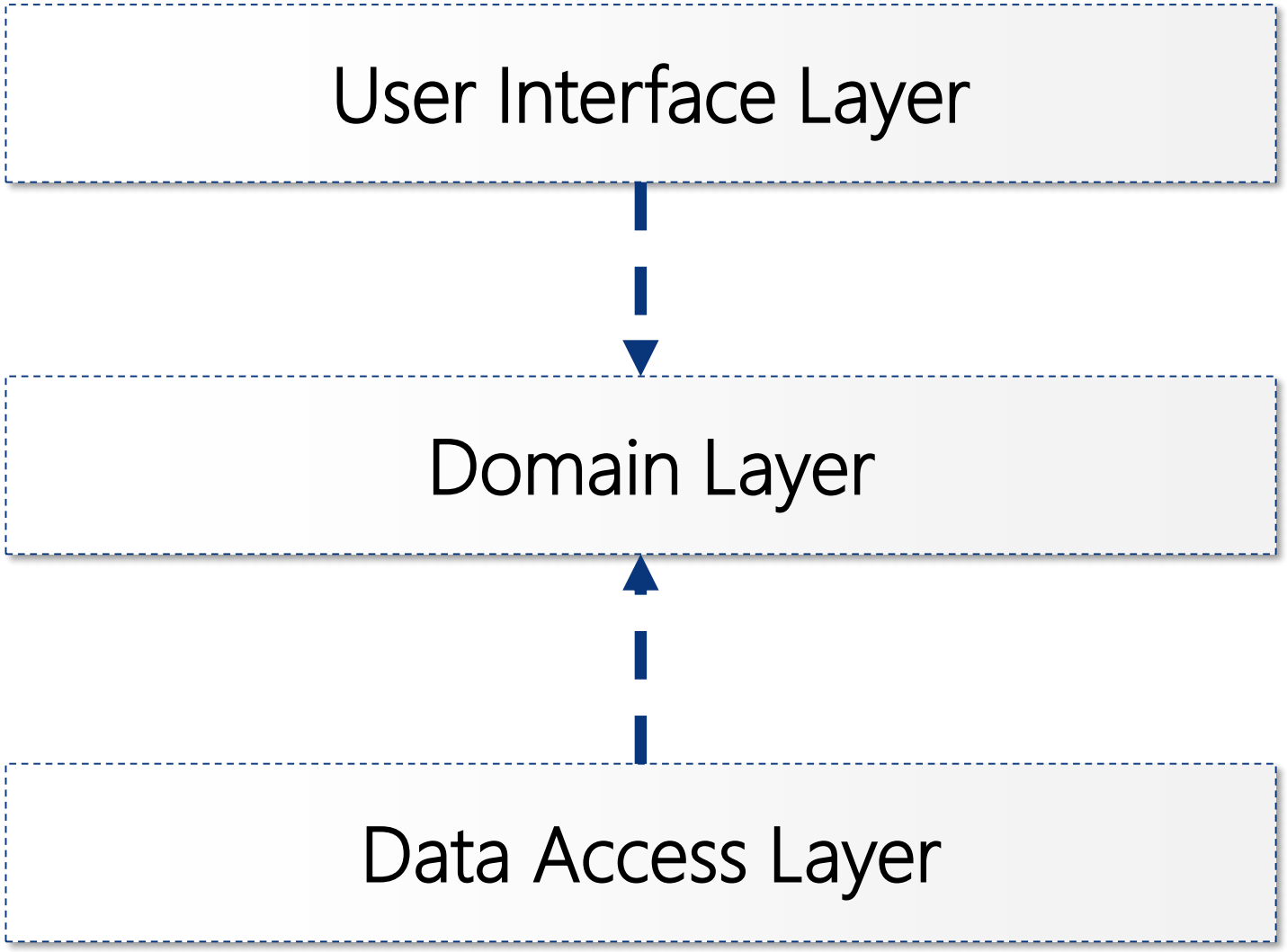
- Discussion: Evaluating the Design
- ***Pattern: Repository (with Entity Framework)***
- Workshop A.2: Data Access Layer with Repository
- Discussion: Evaluating the Design Again
- *Automatic Testing*
- Workshop A.3: Modify the Data Access Layer

Agenda

- Discussion: Evaluating the Design
- *Pattern: Repository (with Entity Framework)*
- **Workshop A.2: Data Access Layer with Repository**
- Discussion: Evaluating the Design Again
- *Automatic Testing*
- Workshop A.3: Modify the Data Access Layer

TEKNOLOGISK
INSTITUT

Better SOLID Design



TEKNOLOGISK
INSTITUT

Workshop A.2:
Data Access Layer with Repository

Agenda

- Discussion: Evaluating the Design
- *Pattern: Repository (with Entity Framework)*
- Workshop A.2: Data Access Layer with Repository
- **Discussion: Evaluating the Design Again**
- *Automatic Testing*
- Workshop A.3: Modify the Data Access Layer

Discussion: Evaluating the Design

- Can we change the UI Layer from Console to e.g. Web or WPF?
- Can we unit test the Domain Layer?
- Can we change the Data Access Layer?

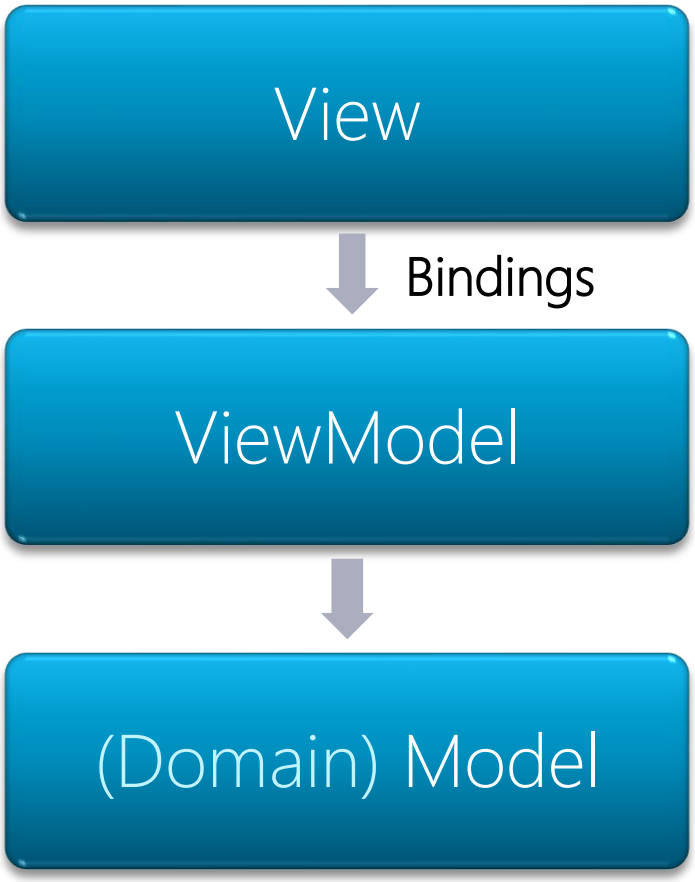
Pattern: Stairway

- *Let your implementation packages depend upon packages that exclusively contain interfaces (or interface-like classes). Moreover, your packages should not depend other implementation packages.*

- Outline
  - This is essentially the "module" part of DIP
  - Avoids the Entourage anti-pattern
  - May not be practically manageable

- See:
  "Adaptive Code" (2nd Edition)
  Gary McLean Hall (2017)

Pattern: Model-View-ViewModel



- Separation between presentation and application logic
- The ViewModel is an abstraction of the View
- Depends heavily on **data binding** and **command binding**

Agenda

- Discussion: Evaluating the Design
- *Pattern: Repository (with Entity Framework)*
- Workshop A.2: Data Access Layer with Repository
- Discussion: Evaluating the Design Again
- ***Automatic Testing***
- Workshop A.3: Change the Data Access Layer

Agenda

- Discussion: Evaluating the Design
- *Pattern: Repository (with Entity Framework)*
- Workshop A.2: Data Access Layer with Repository
- Discussion: Evaluating the Design Again
- *Automatic Testing*
- **Workshop A.3: Change the Data Access Layer**

TEKNOLOGISK
INSTITUT

Workshop A.3:
Change the Data Access Layer

Summary

- Discussion: Evaluating the Design
- *Pattern: Repository (with Entity Framework)*
- Workshop A.2: Data Access Layer with Repository
- Discussion: Evaluating the Design Again
- *Automatic Testing*
- Workshop A.3: Change the Data Access Layer


WINCUBATE
Jesper Gulmann Henriksen
PhD, MCT, MCSD, MCPD
Phone : +45 22 12 36 31
Email : jgh@wincubate.net
WWW : http://www.wincubate.net
Hasselvangen 243
8355 Solbjerg
Denmark