Building API server-side architecture for Beginners

GopherCon 2019

2019.07.27 - @hgsgtk

Talk abstract

- A practical approach to build server-side architecture in a Go project

- Especially for teams who do not have Go experience in business

About me

BASE

@hgsgtk

Kazuki Higashiguchi / Backend engineer in Japan🇯🇵

BASE BANK, Inc. / Dev Division / Tech lead

Talk structure

1. Problem of building architecture for beginners
2. Approach to build architecture
3. Summary

Talk structure

1. **Problem of building architecture for beginners**
2. Approach to build architecture
3. Summary

Why I need server-side architecture

1. Keep a design easy to change
   - -> Separate external input/output and business logic

2. Reach common understanding of implementation policies in a team
   - -> To make readable and maintainable code

Reality of building architecture

- No absolute answer for any projects

- We should determine a suitable architecture for the project

To determine a suitable architecture

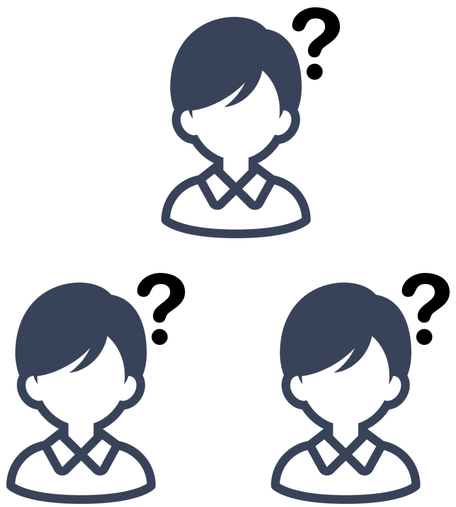
- We should consider various things
  - Service requirements
  - Team member ability
  - Service scale
  - …etc

Problem of building architecture for beginners

- We should consider various things
  - Service requirements
  - Team member ability
  - Service scale
  - …etc

For beginners, “Team member ability” is a factor that can not be ignored

Go beginners have a lot of questions…



- Go basic questions -

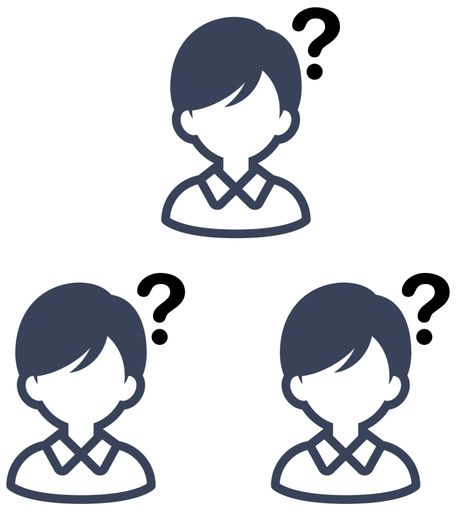
How to implement API server by net/http package

How to write an unit test

How to use interface type

If we determine complicated architecture from the beginning

ex. “Adopt clean architecture!”



- Go basic questions -

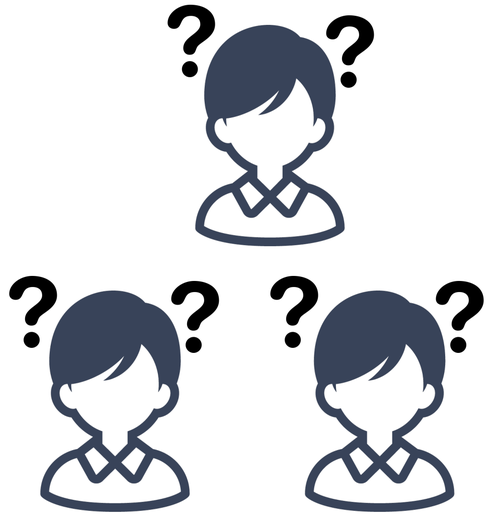
How to implement API server by net/http package

How to write an unit test

How to use interface type

If we determine complicated architecture from the beginning

ex. “Adopt clean architecture!”



- Go basic questions -

- How to implement API server by net/http package
- How to write an unit test
- How to use interface type

- Architecture questions -

- What’s clean architecture?
- How to apply Dependency Inversion Principle in Go?
- What should we write in use case layer?

If we determine complicated architecture from beginning

ex. “Adopt clean architecture!”

- Go basic questions -

How to implement API server by net/http package

How to write an unit testing

How to use interface type

- Architecture questions -

What’s clean architecture?

How to apply Dependency Inversion Principle in Go?

What should we write in use case layer?

There are many questions in team members’ mind

If we determine complicated architecture from beginning

ex. “Adopt Clean Architecture!”

- Go Basic Questions -

How to implement API server by net/http package

How to write unit testing

How to use interface type

- Architecture Questions -

What’s clean architecture?

How to apply Dependency Inversion Principle in Go?

What we should write in use case layer?

When it gets worst,

Confusing… 🤯

Why I need server-side architecture

1. Keep a design easy to change
   - -> Separate external input/outnput and business logic

We are not able to achieve
“common understanding” 🤯

2. Common understanding of implementation policies in a team
   - -> To make readable and maintainable code

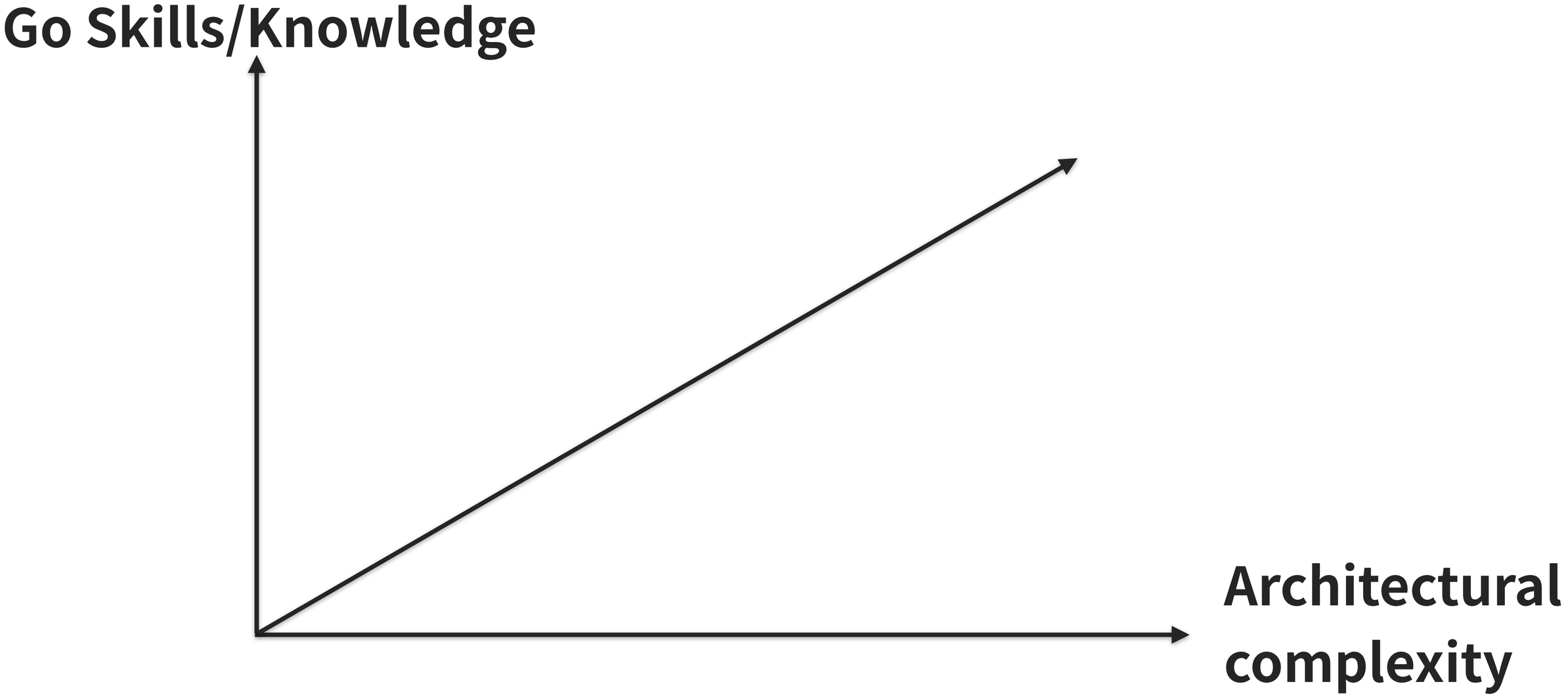
Talk structure

1 Problem of building architecture for beginners

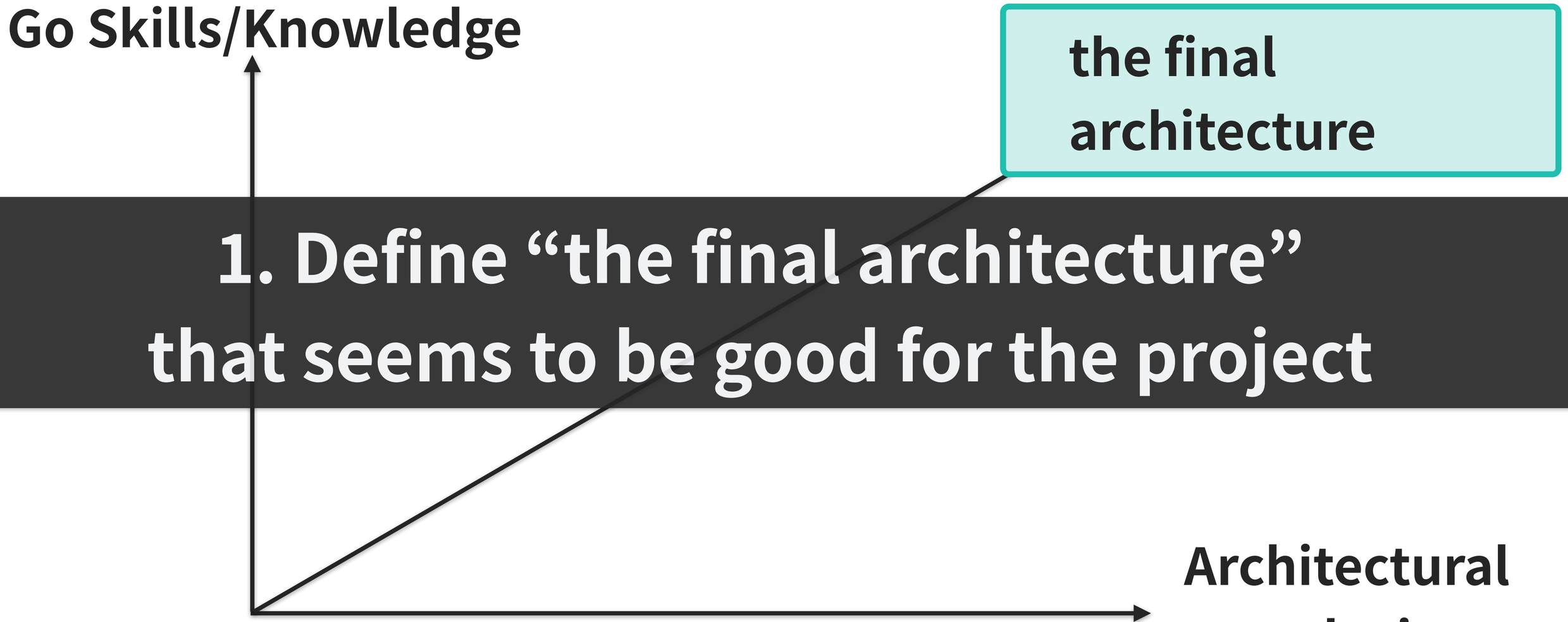
2 **Approach to build architecture**

3 Summary

Approach to build architecture



Approach to build architecture



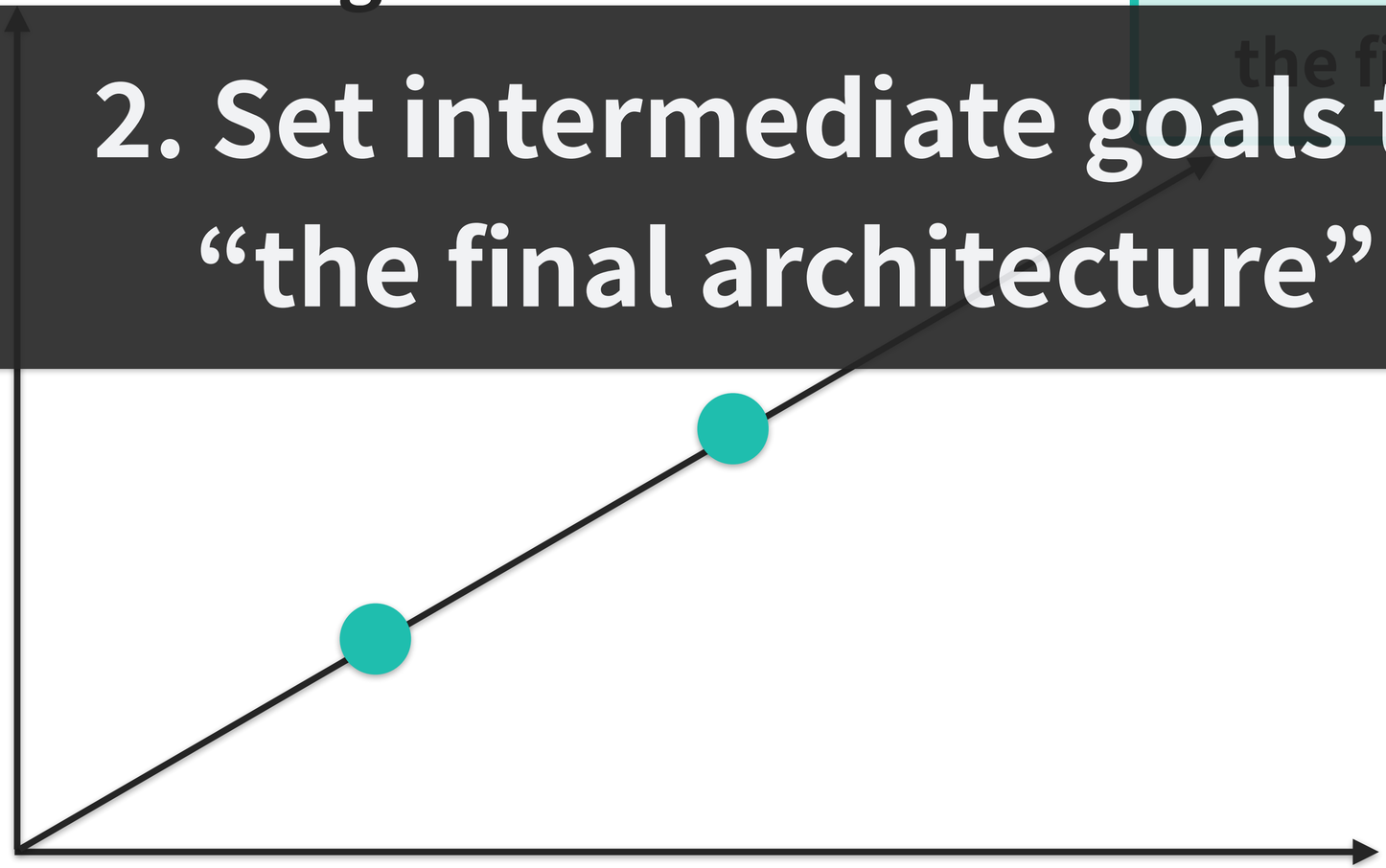
1. Define “the final architecture” that seems to be good for the project

Approach to build architecture

Go Skills/Knowledge

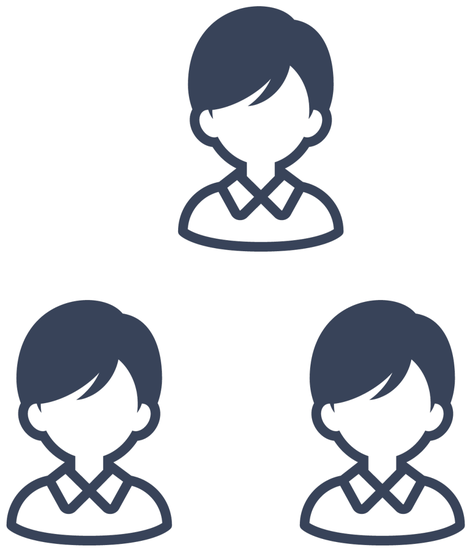
the final architecture

2. Set intermediate goals to “the final architecture”



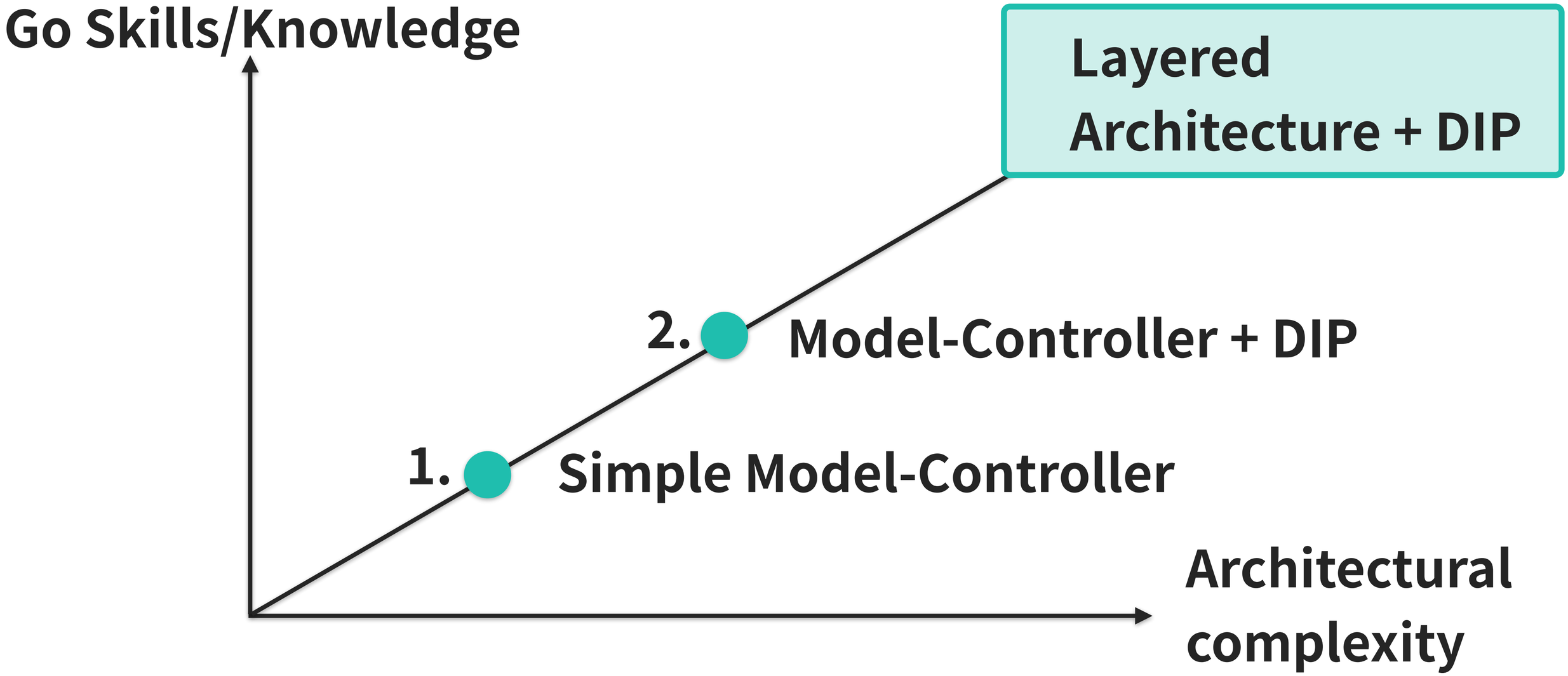
Architectural complexity

Example of my team

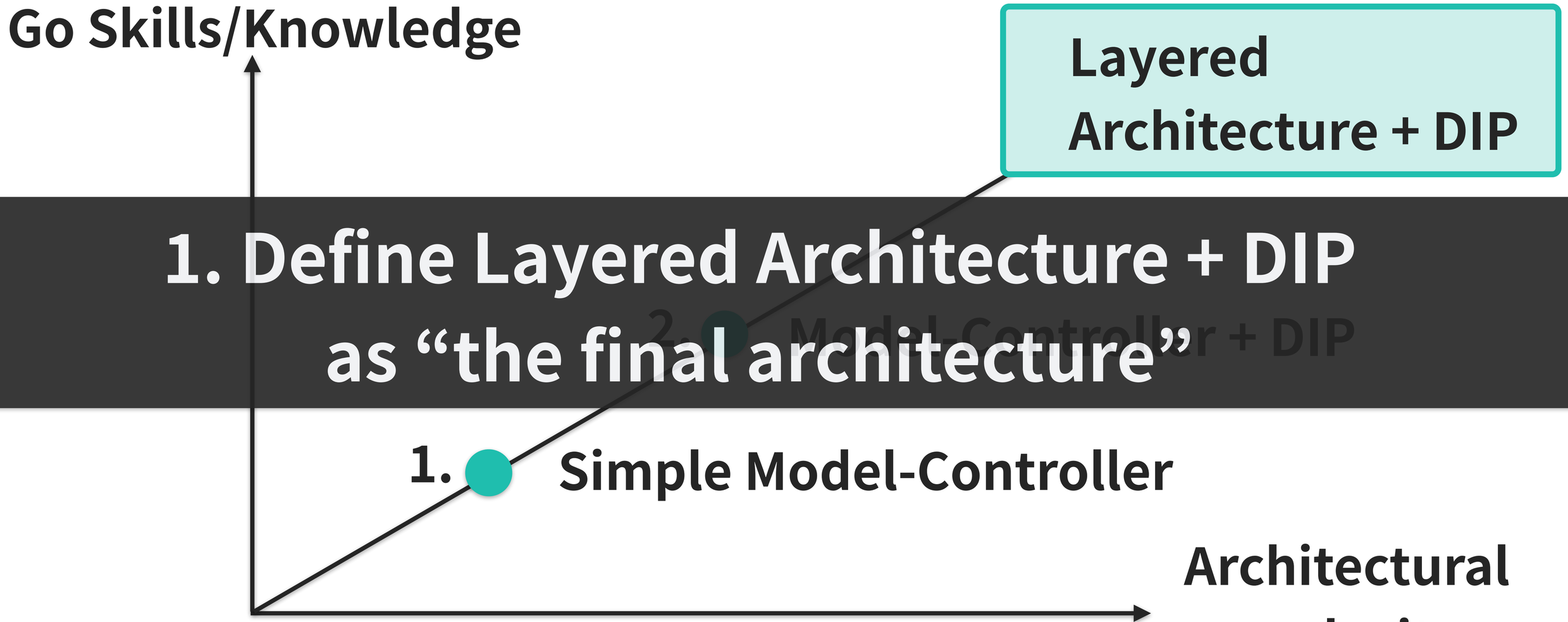


- The team have few Go experience in business
- The Team has 3 backend engineers
  - usually use PHP in work

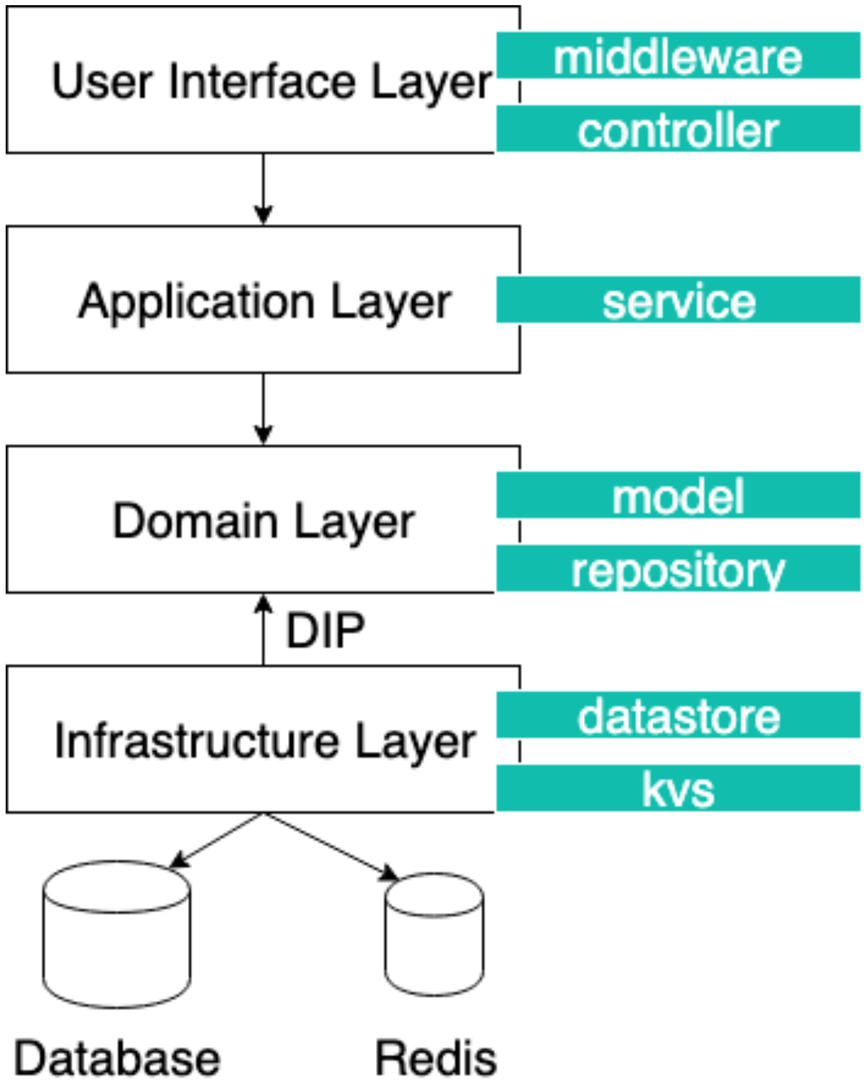
Example of my team



Example of my team

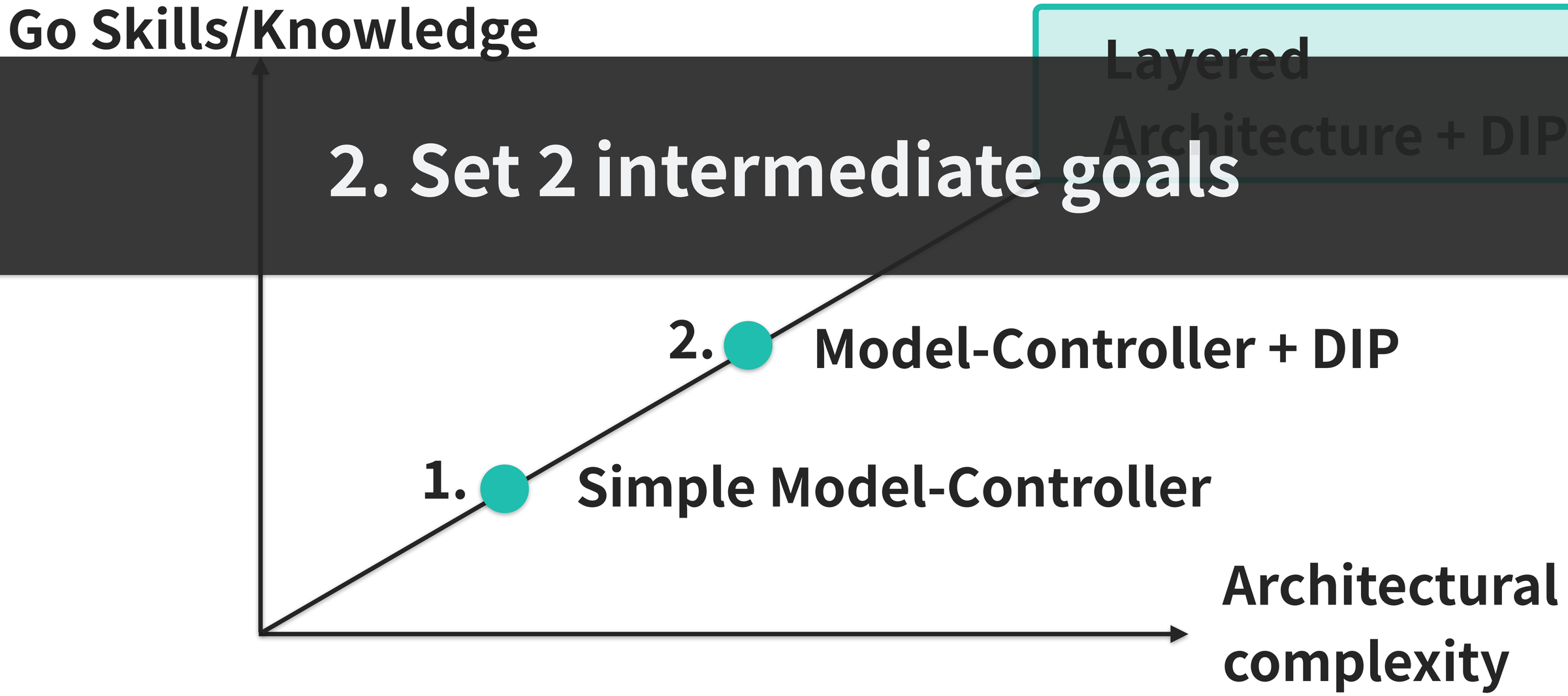


Layered Architecture + DIP

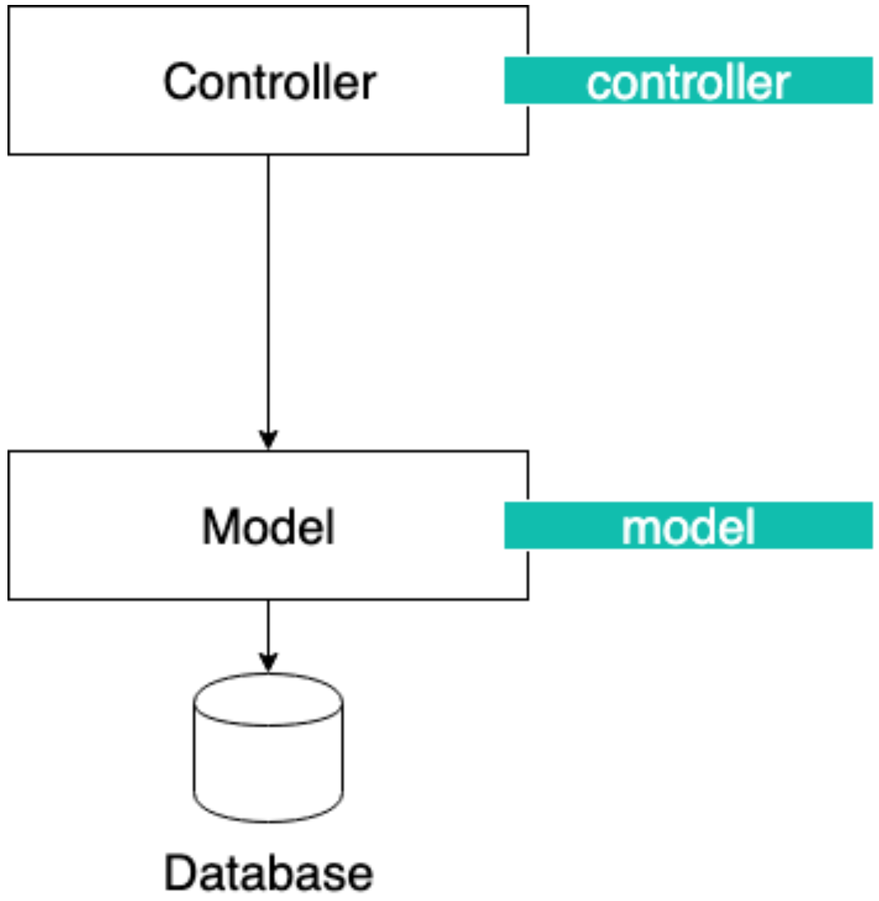


- Refer to Layered Architecture
- Apply DIP (Dependency Inversion Principle) to isolate domain logic from infrastructure implementations(ex. database handling)

Example of my team



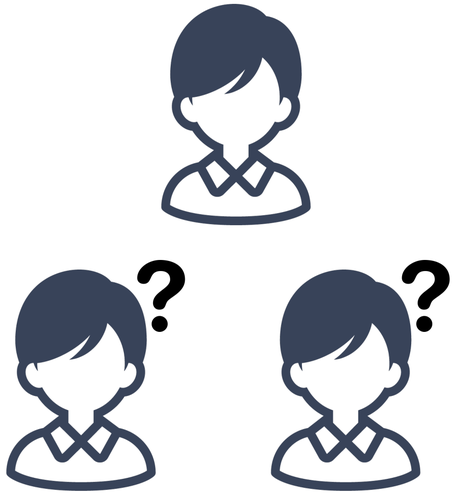
1st goal: Simple Model-Controller



- Simple design only with controller and model
- Model includes implementations such as handling a database
- To get used to Go API development

At 1st goal, team will acquire…

Got it!



- Go Basic Questions -

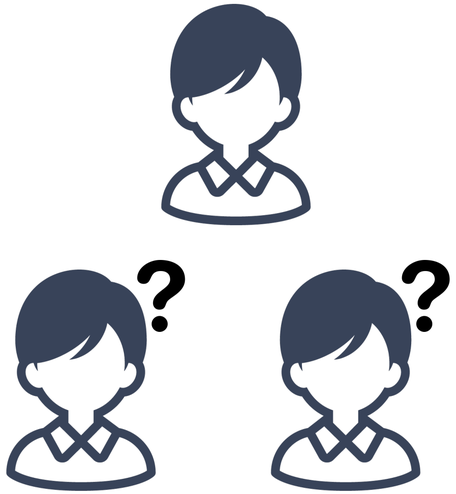
How to implement API server by net/http package

How to write an unit test

How to use interface type

At 1st goal, team will acquire…

Got it!



- Go Basic Questions -

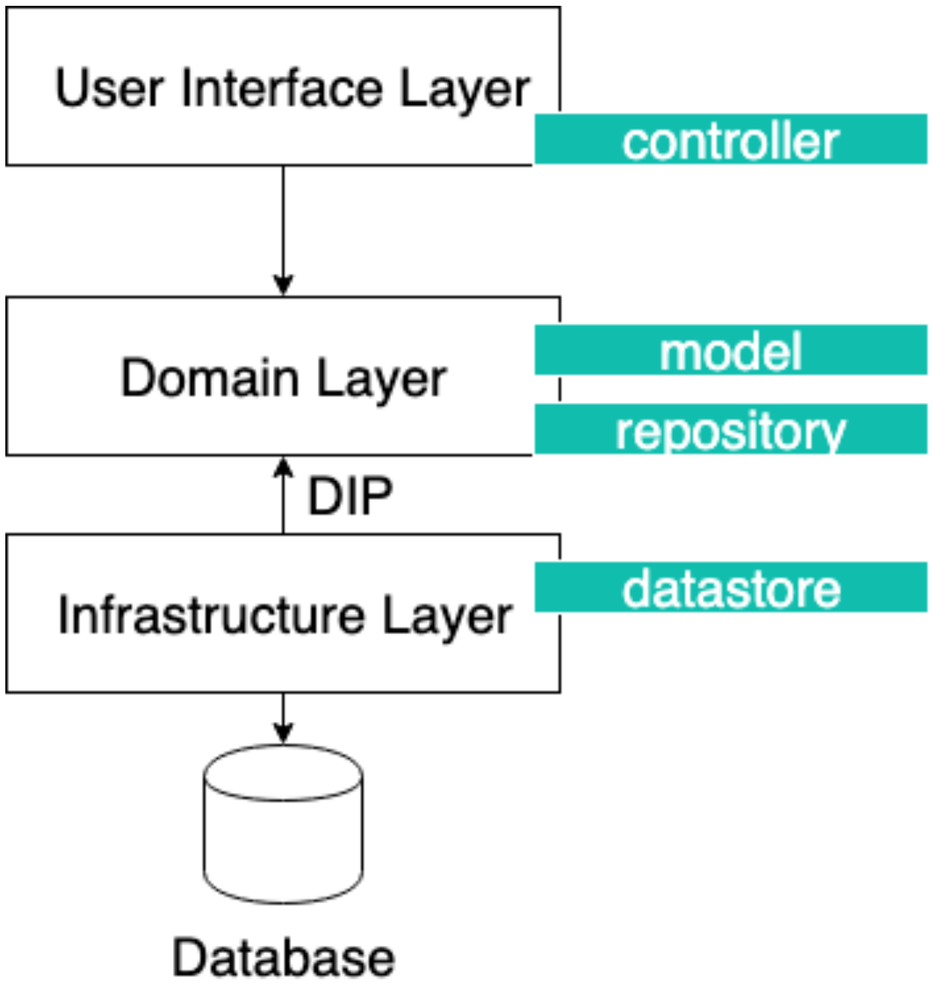
How to implement API server by net/http package

How to write an unit test

How to use interface type

Acquire Go Basic from simple design code

2nd goal: Model-Controller + DIP

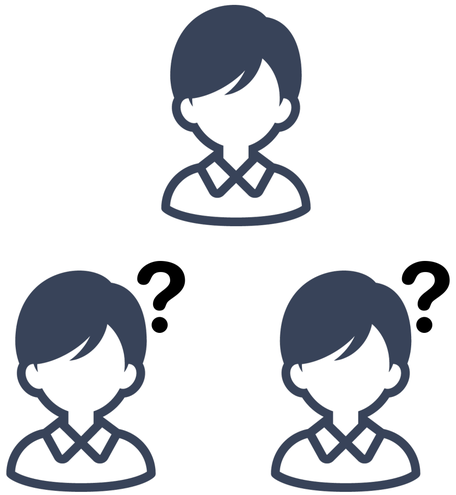


- Separate infrastructure implementations from model
- Prepare repository package and move database handling implementation to datastore package. (apply DIP)
- Get used to how to use interface in Go

At 2nd goal, team will acquire…

Final goal: “Layered Architecture + DIP”

Got it!



- Go Basic Questions -

How to implement API server by net/http package

How to write an unit test

How to use interface type

- Architecture Questions -

What’s layered architecture?

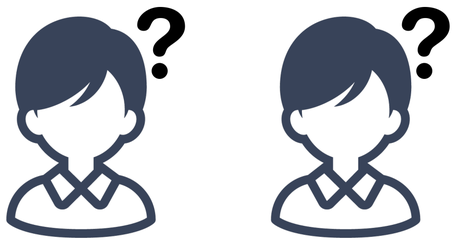
How to apply Dependency Inversion Principle in Go?

What should we write in application layer?

At 2nd goal, team will acquire…

Final goal: “Layered Architecture + DIP”

Acquire Basic of Go and Archictecture from more complicated design



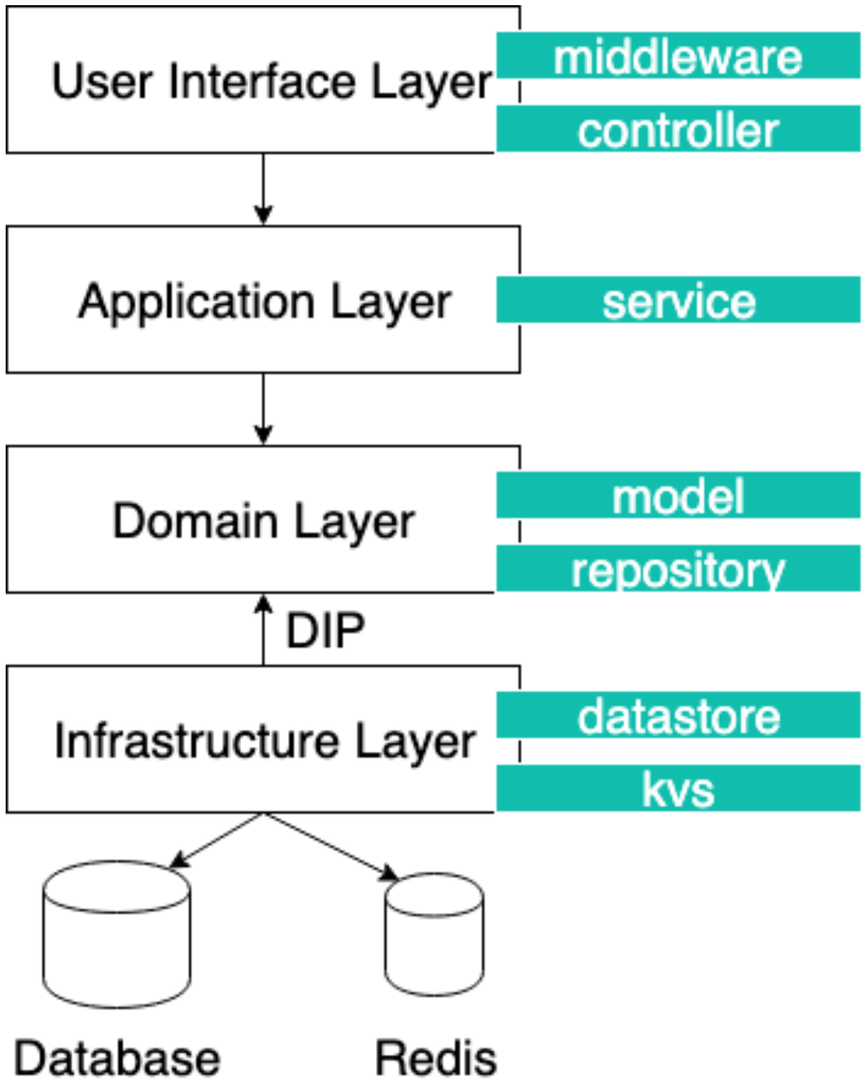
How to write an unit test

How to use interface type

How to apply Dependency Inversion Principle in Go?

What should we write in application layer?

Final goal: Layered Architecture + DIP

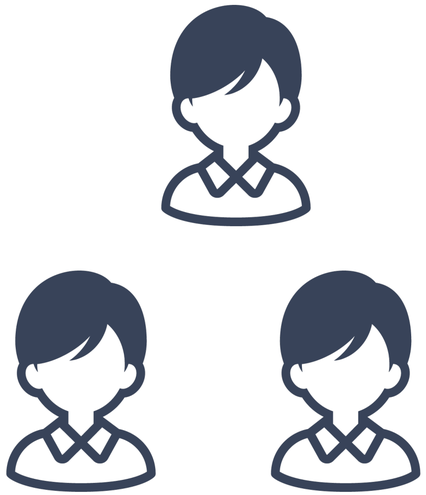


- Introduce service package corresponding to “service application layer” in DDD as processing becomes more complicated

At final goal, team will reach…

Final goal: “Layered Architecture + DIP”

Got it!



- Go Basic Questions -

How to implement API server by net/http package

How to write an unit test

How to use interface type

- Architecture Questions -

What’s layered architecture?

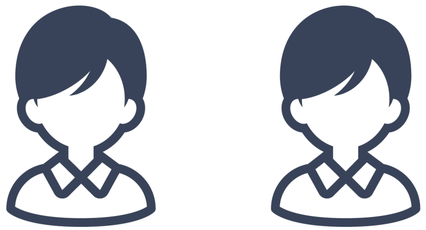
How to apply Dependency Inversion Principle in Go?

What should we write in application layer?

At final goal, team will reach…

Final goal: “Layered Architecture + DIP”

Reach common understanding of implementation policies



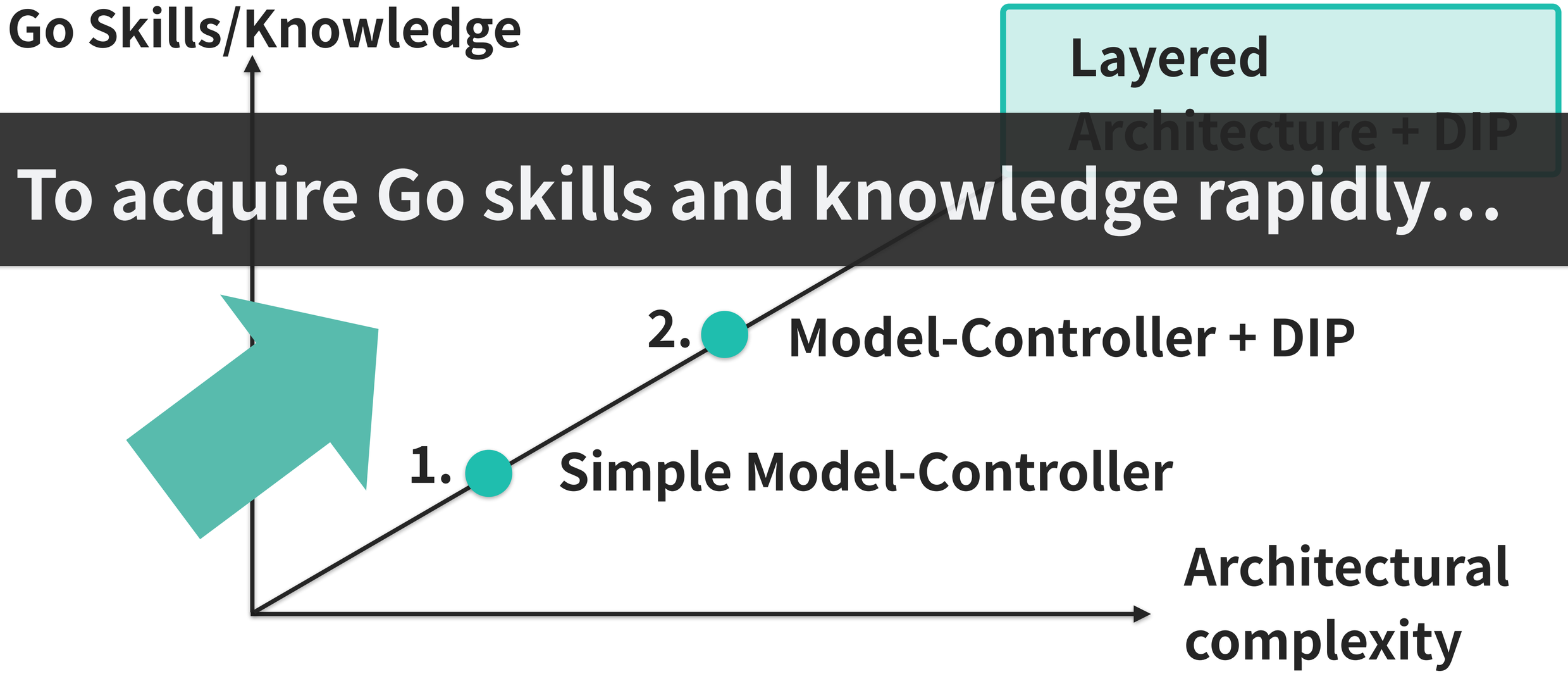
How to write an unit test

How to use interface type

How to apply Dependency Inversion Principle in Go?

What should we write in application layer?

Approach to build architecture



To acquire Go skills and knowledge rapidly…

1. Use standard package as much as possible
   - To learn Go language itself
   - ex. use net/http package to serve HTTP
2. Write a test
   - To get feedback on the code design (such as testability)
   - For quick refactoring

Talk structure

1 Problem of building architecture for beginners

2 Approach to build architecture

3 **Summary**

Summary

- In my approach, an architecture grew with team members' ability growth
- Set a final goal and intermediate goals
- To grow rapidly, use standard package as much as possible and write a test

Feel free to ask me any Q 👍

GopherCon 2019

2019.07.27 - @hgsgtk