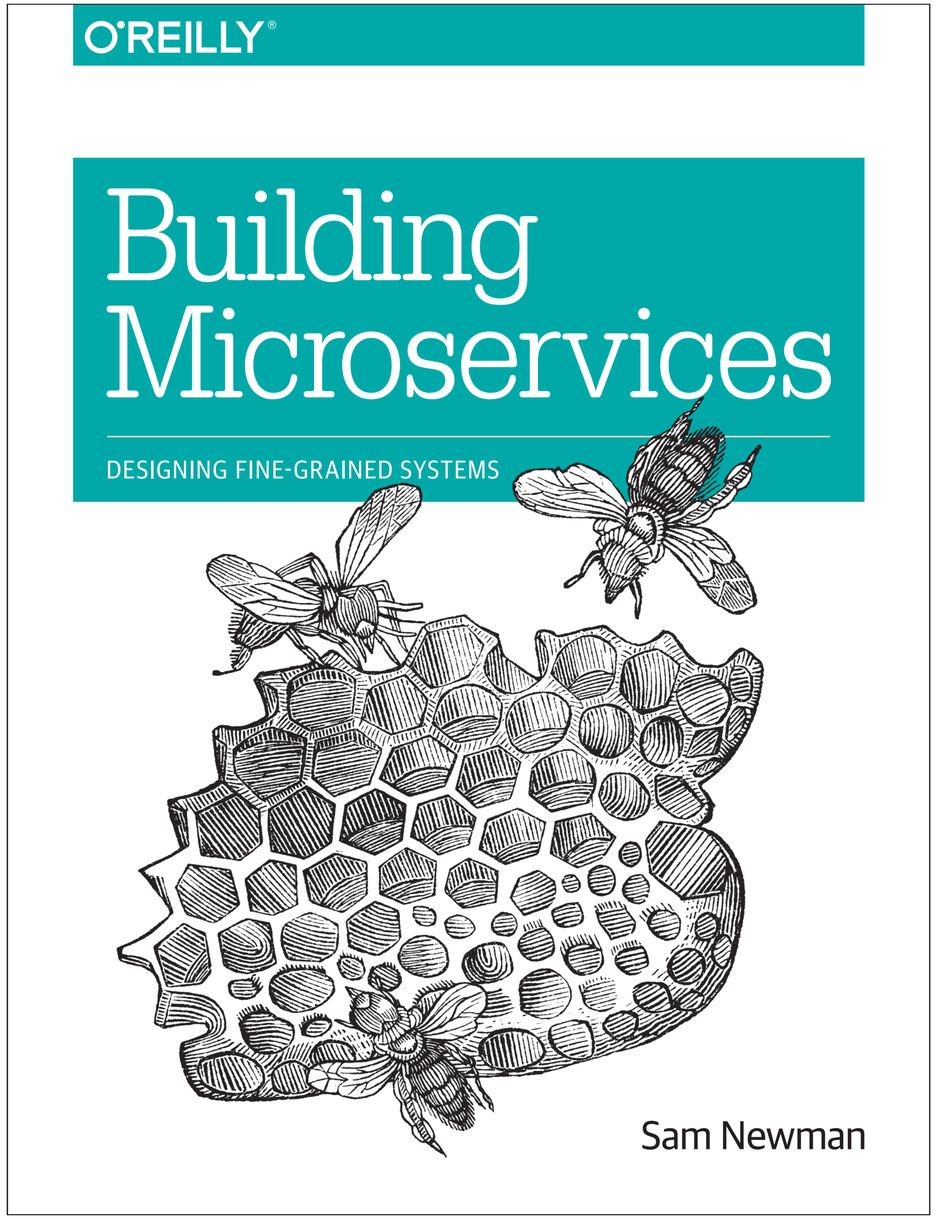
HIDING THE LEAD

COUPLING, COHESION AND MICROSERVICES

Sam Newman

https://www.flickr.com/photos/spongebabyalwaysfull/8793058944/



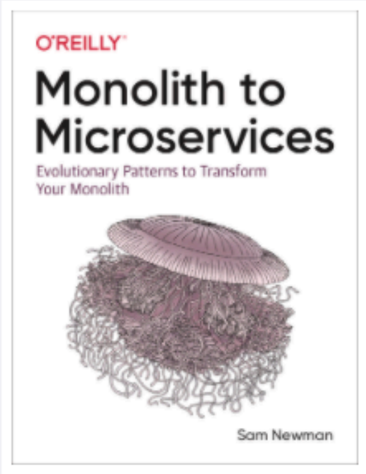
Sam Newman
& Associates

NEW BOOK!

Monolith To Microservices.

Monolith To Microservices is a new book on system decomposition from O'Reilly

How do you detangle a monolithic system and migrate it to a microservices architecture? How do you do it while maintaining business-as-usual? As a companion to Building Microservices, this new book details multiple approaches for helping you transition from existing monolthic systems to microservice architectures. This book is ideal if you're looking to evolve your current systems, rather than just rewriting everything from scratch.



How To Get The Book.

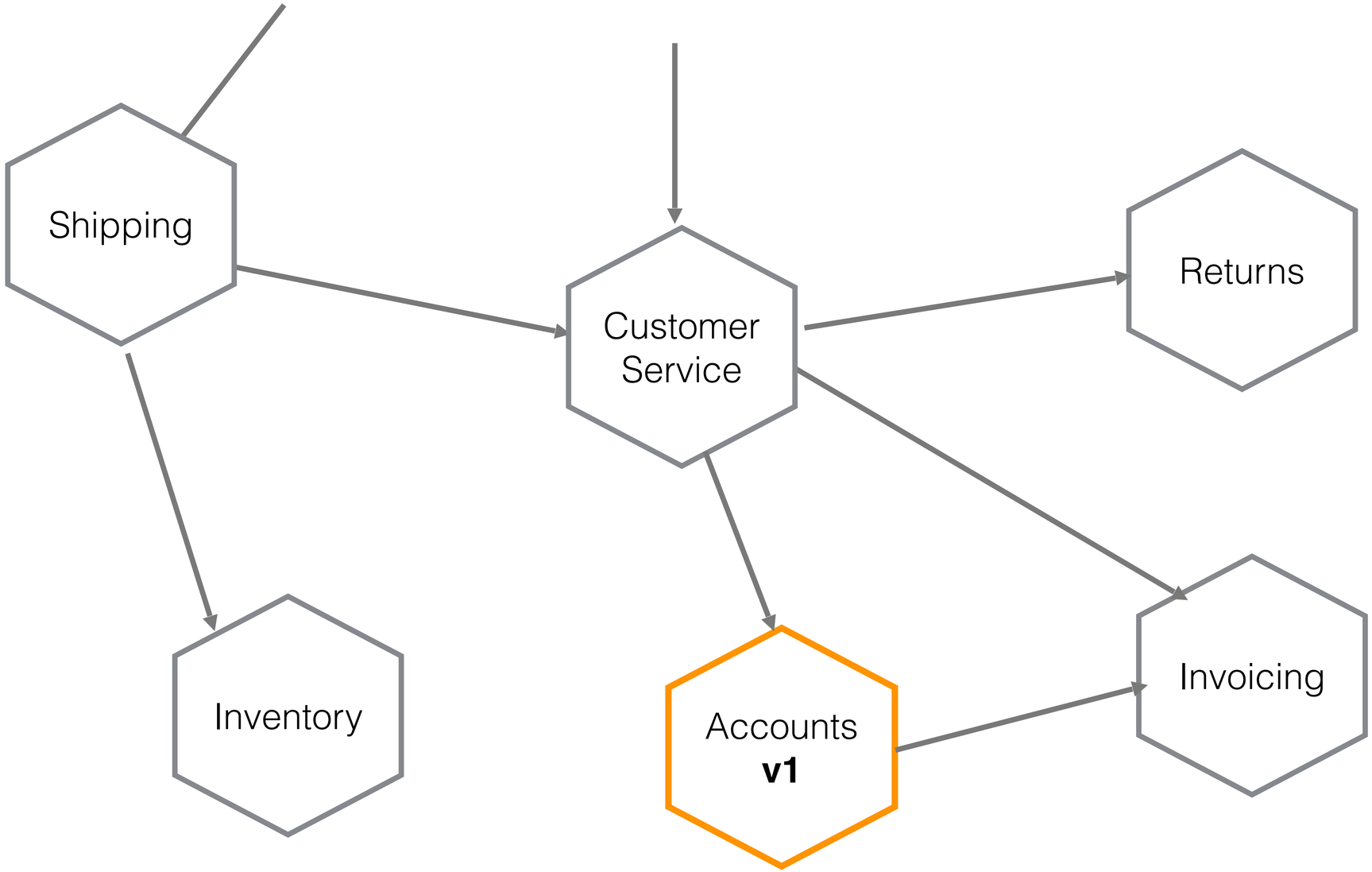
The book is now available, and you can order the dead-tree and Kindle versions over at Amazon. You can also read the book online on O'Reilly's online learning platform.

Read on O'Reilly Learning Online | Order at Amazon.com

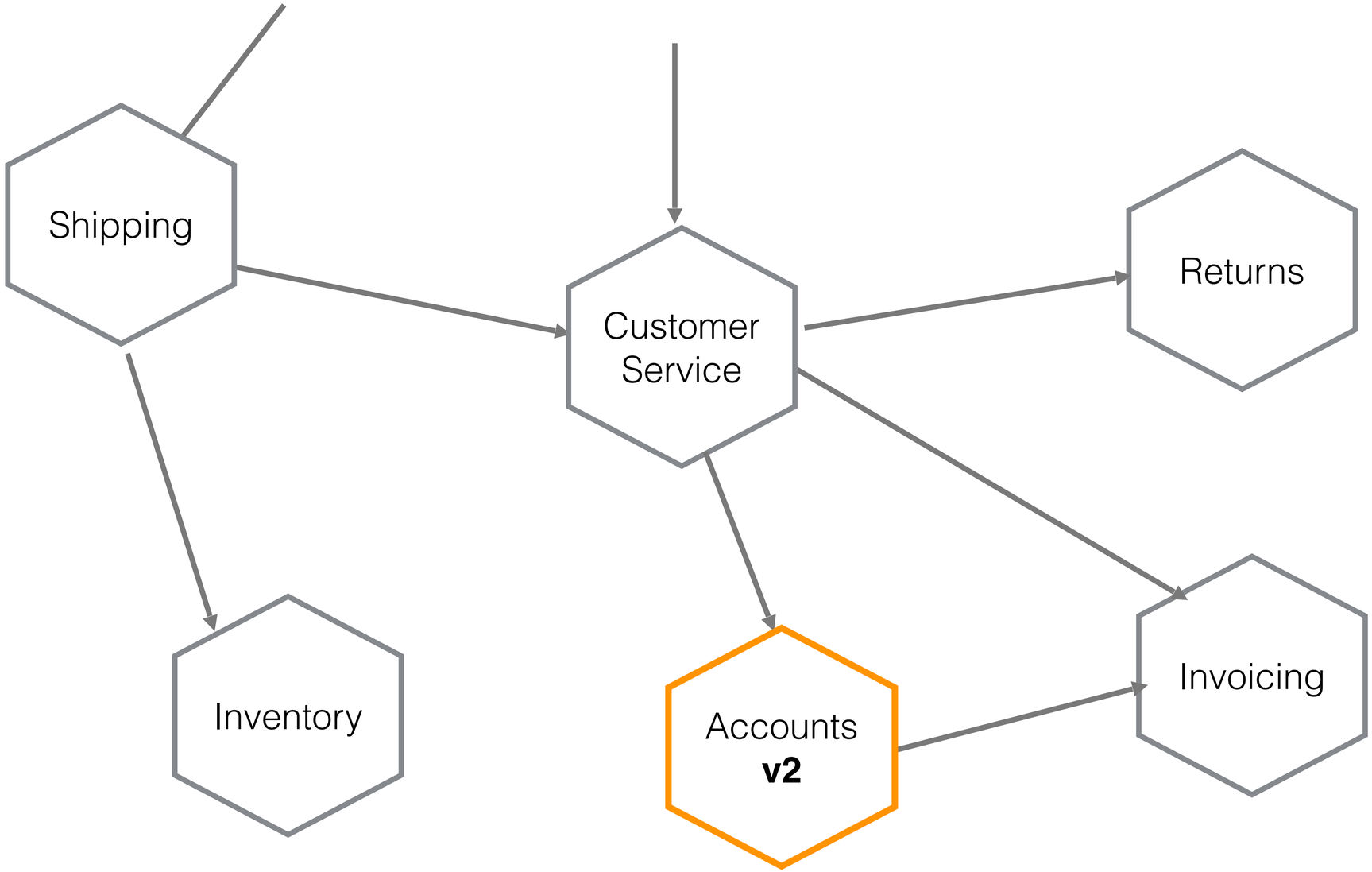
Order at Amazon.co.uk

https://samnewman.io/books/monolith-to-microservices/

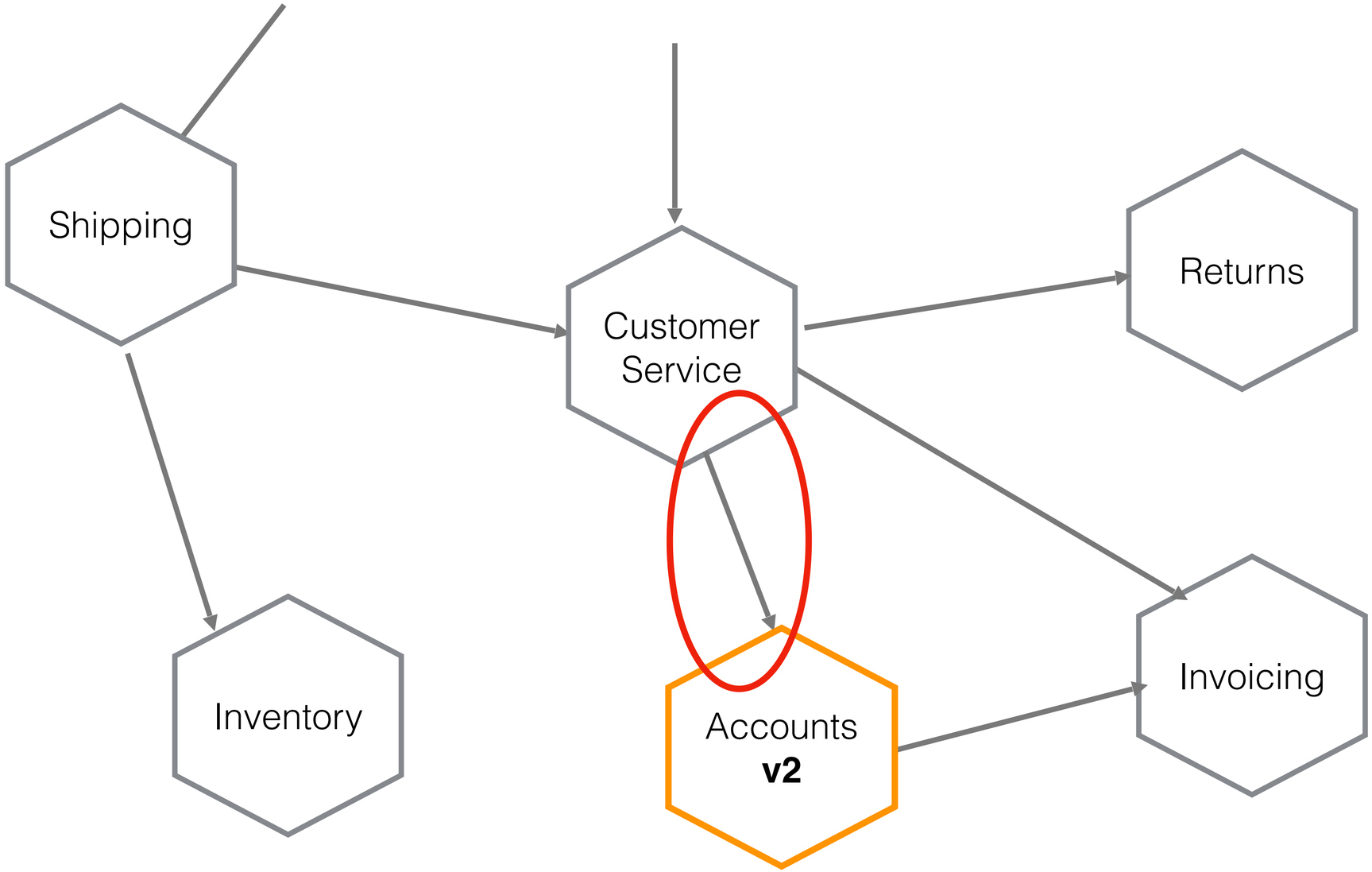
INDEPENDENT DEPLOYABILITY



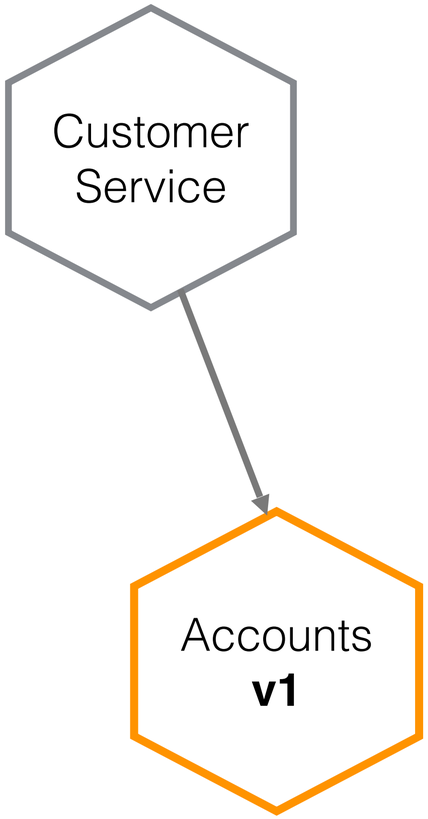
INDEPENDENT DEPLOYABILITY



INDEPENDENT DEPLOYABILITY

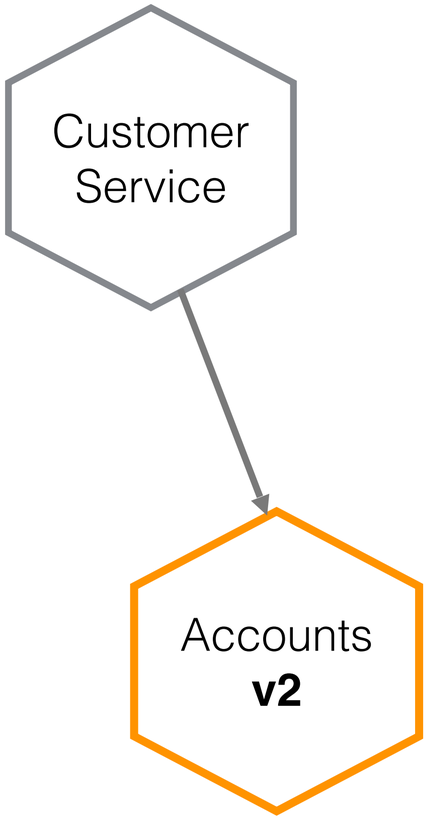


Independent deployability requires interface stability



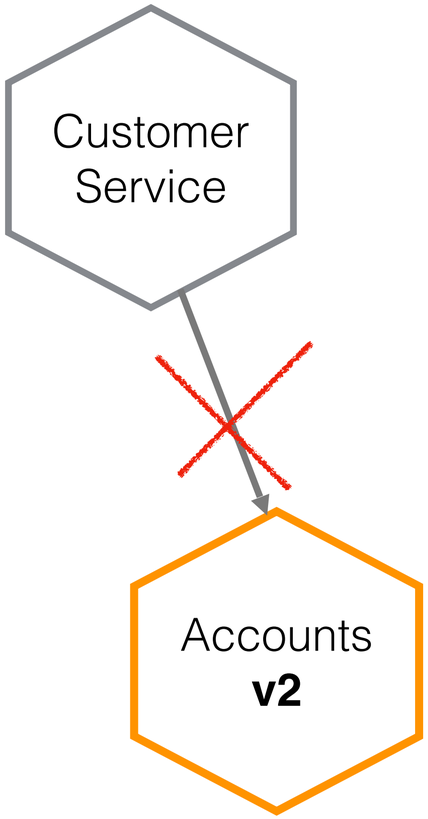
Maintaining backwards compatibility is key

Independent deployability requires interface stability



Maintaining backwards compatibility is key

Independent deployability requires interface stability

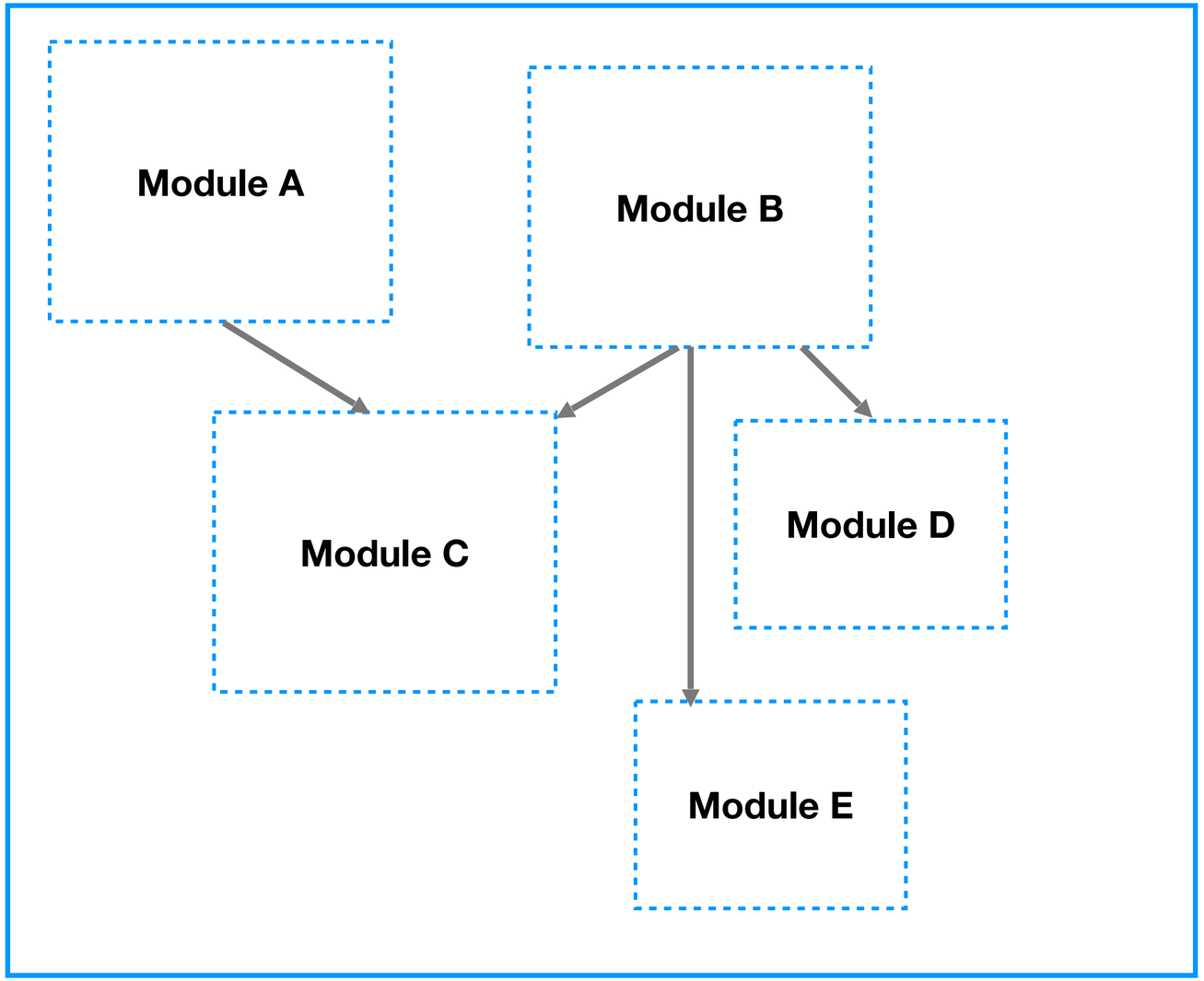


Maintaining backwards compatibility is key

https://www.flickr.com/photos/photographingtravis/16939917735/

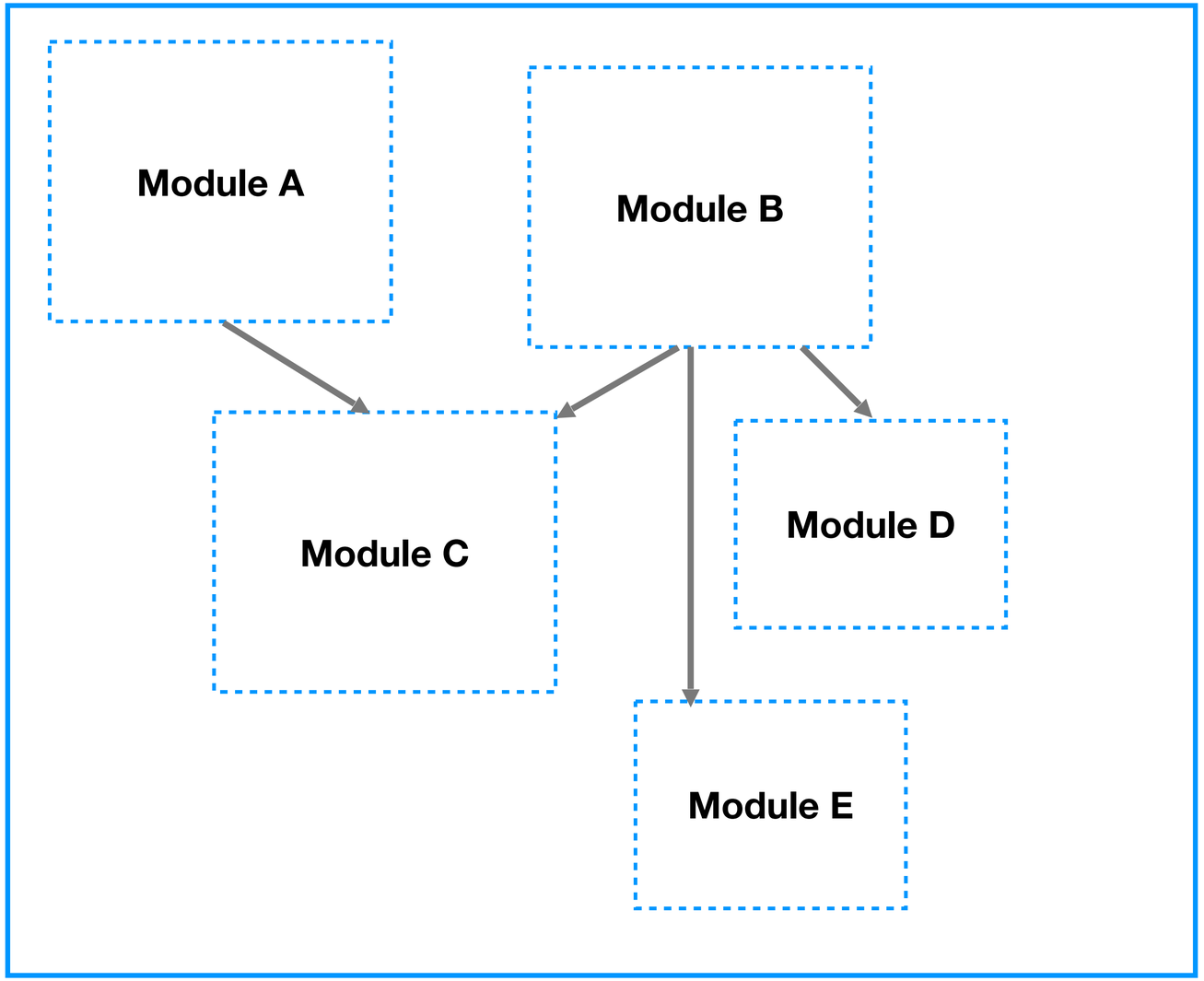
MODULES

MODULES



MODULES

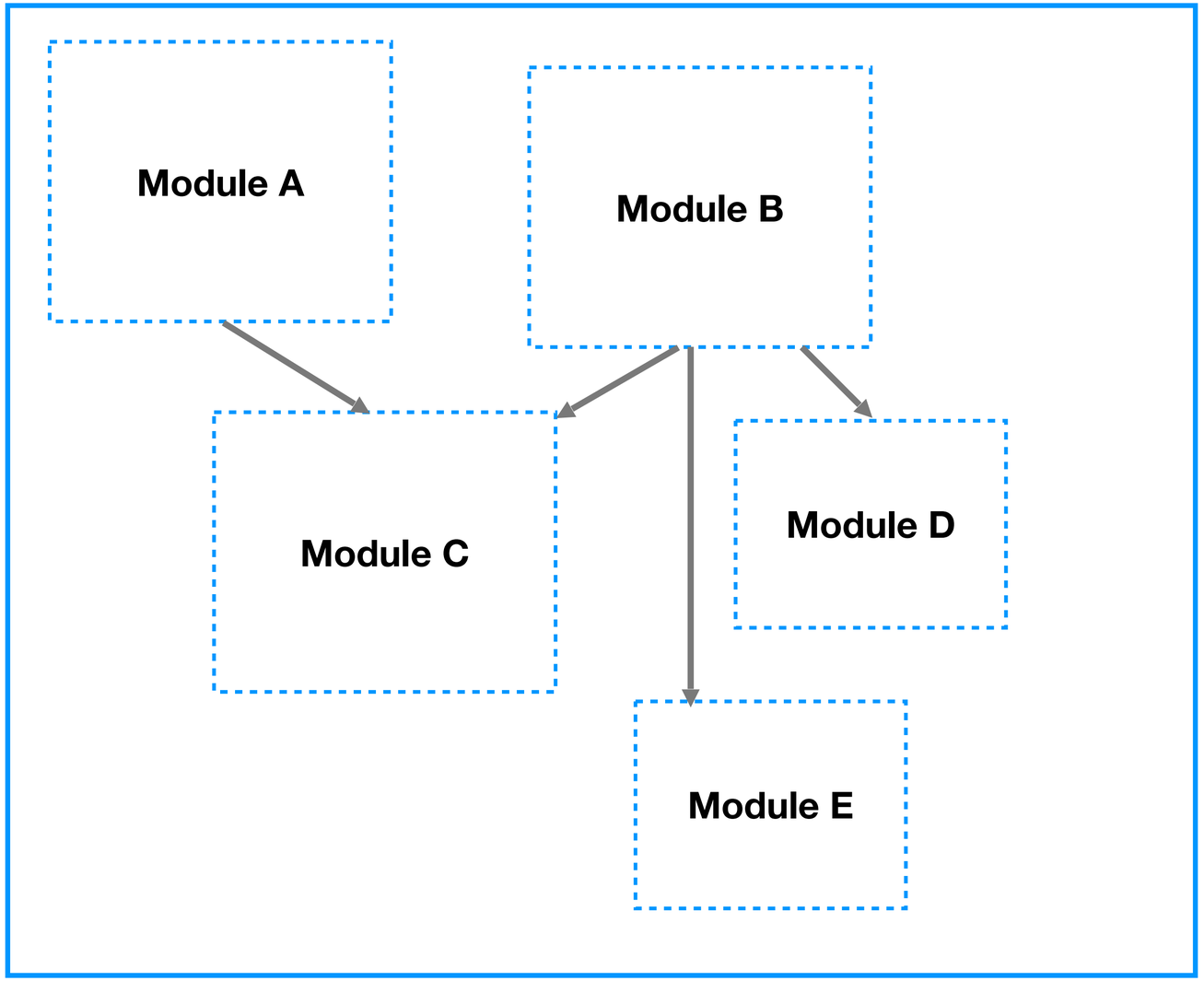
Allow for independent working



MODULES

Allow for independent working

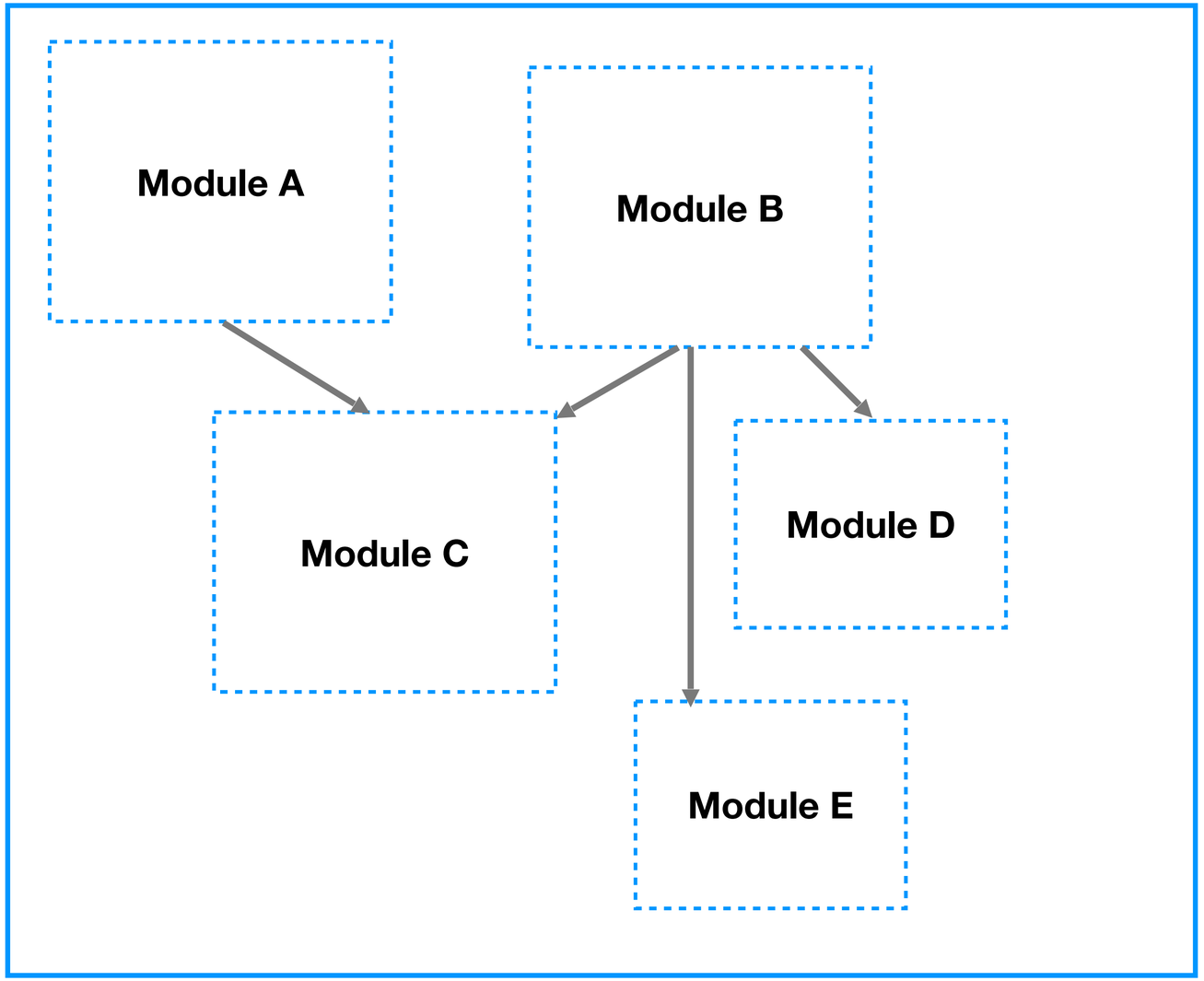
And reuse



MODULES

Allow for independent working

And reuse

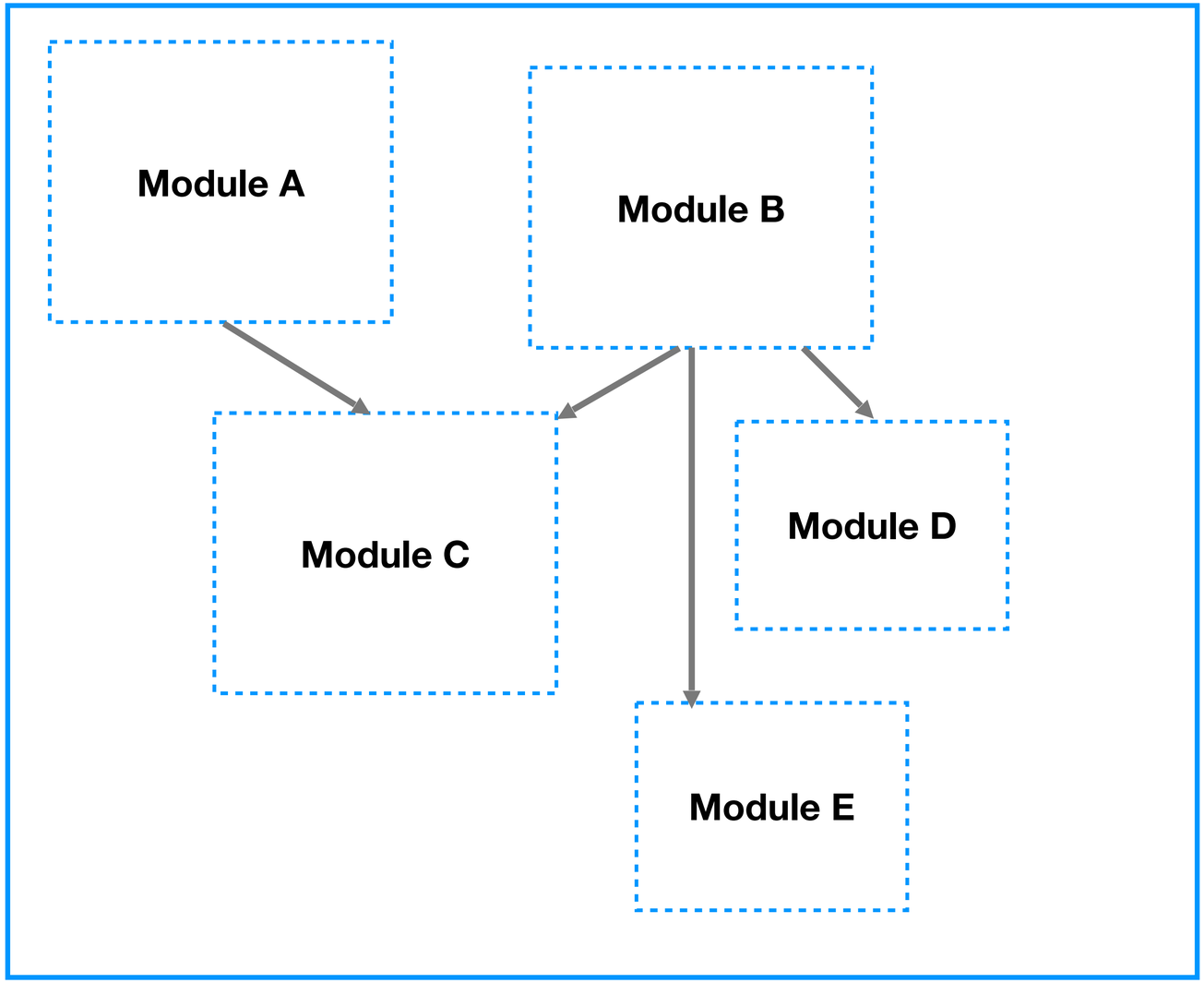


Namespaces, packages

MODULES

Allow for independent working

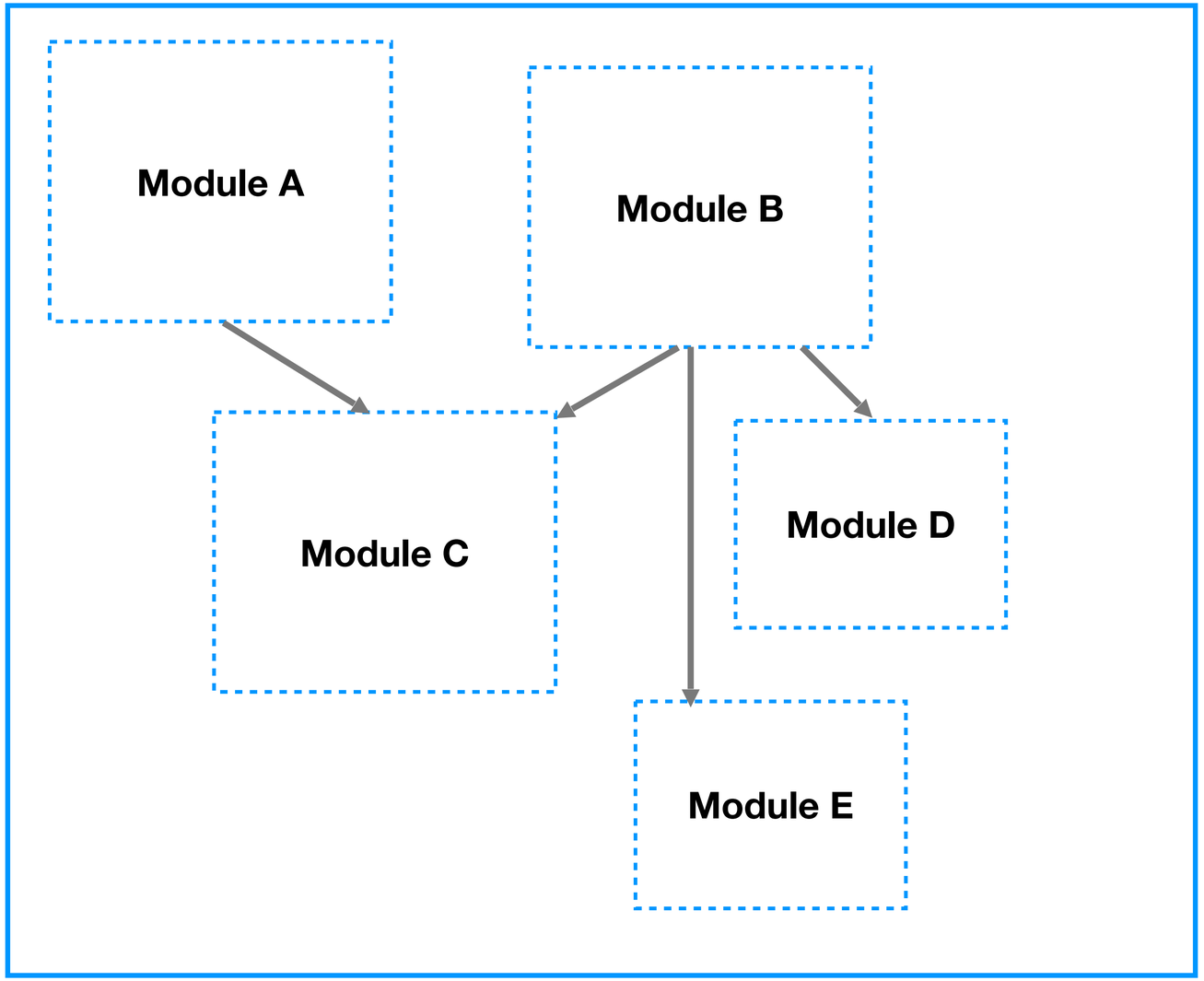
And reuse



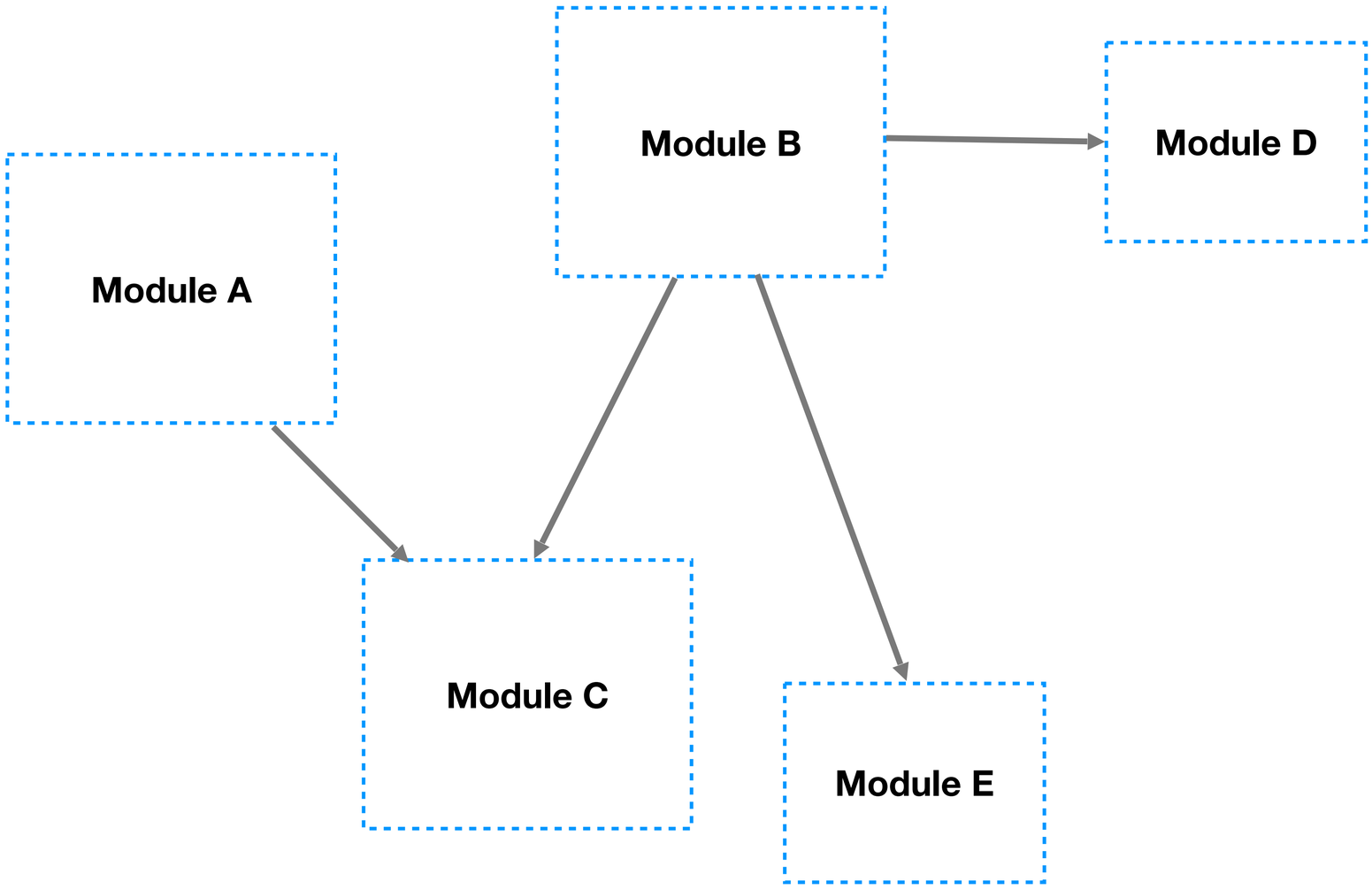
Namespaces, packages

NPMs, Gems, JARs etc

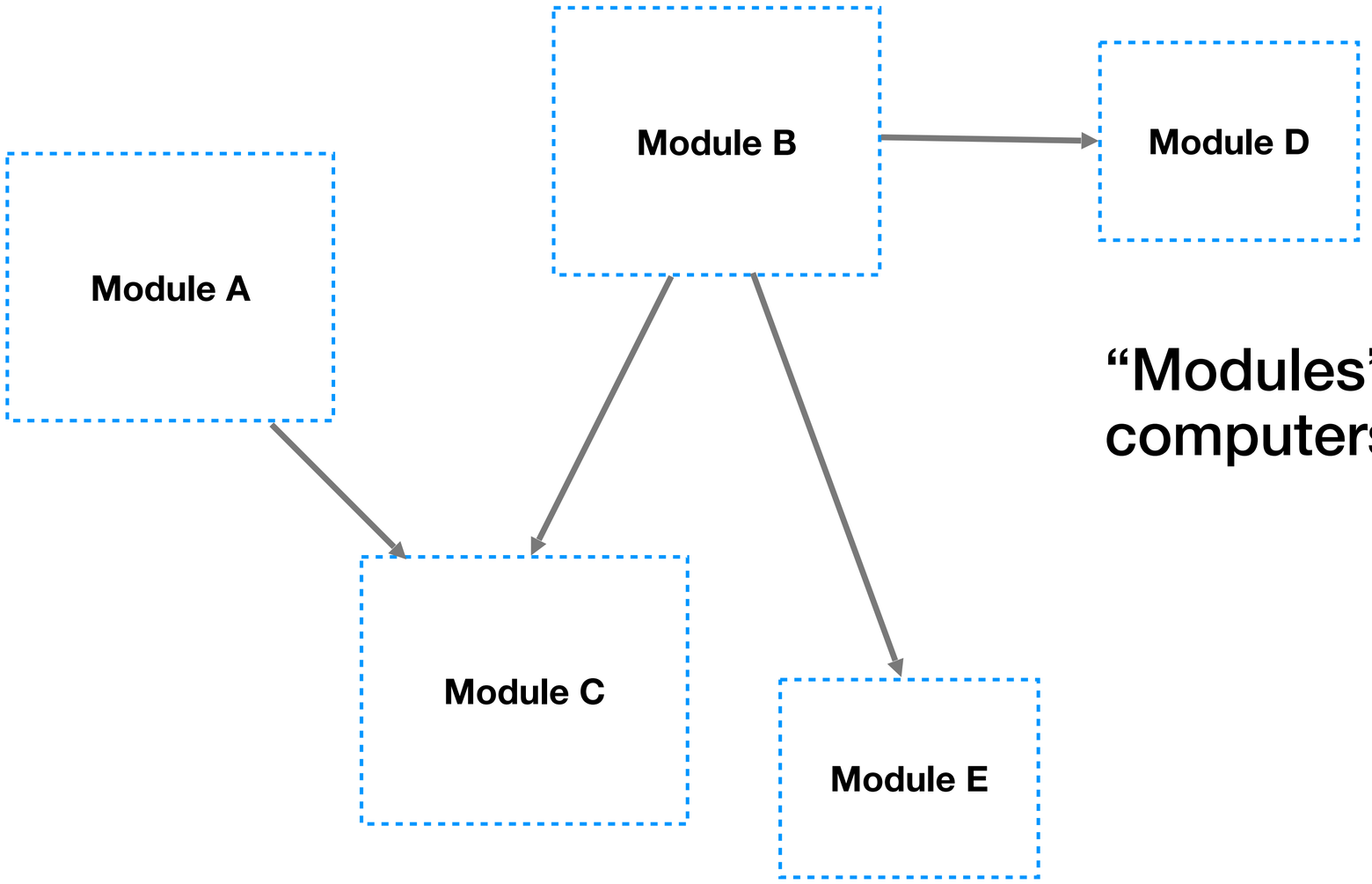
https://www.flickr.com/photos/ibm_media/33838065805/


Module A
Module B
Module C
Module D
Module E

AND MICROSERVICES?

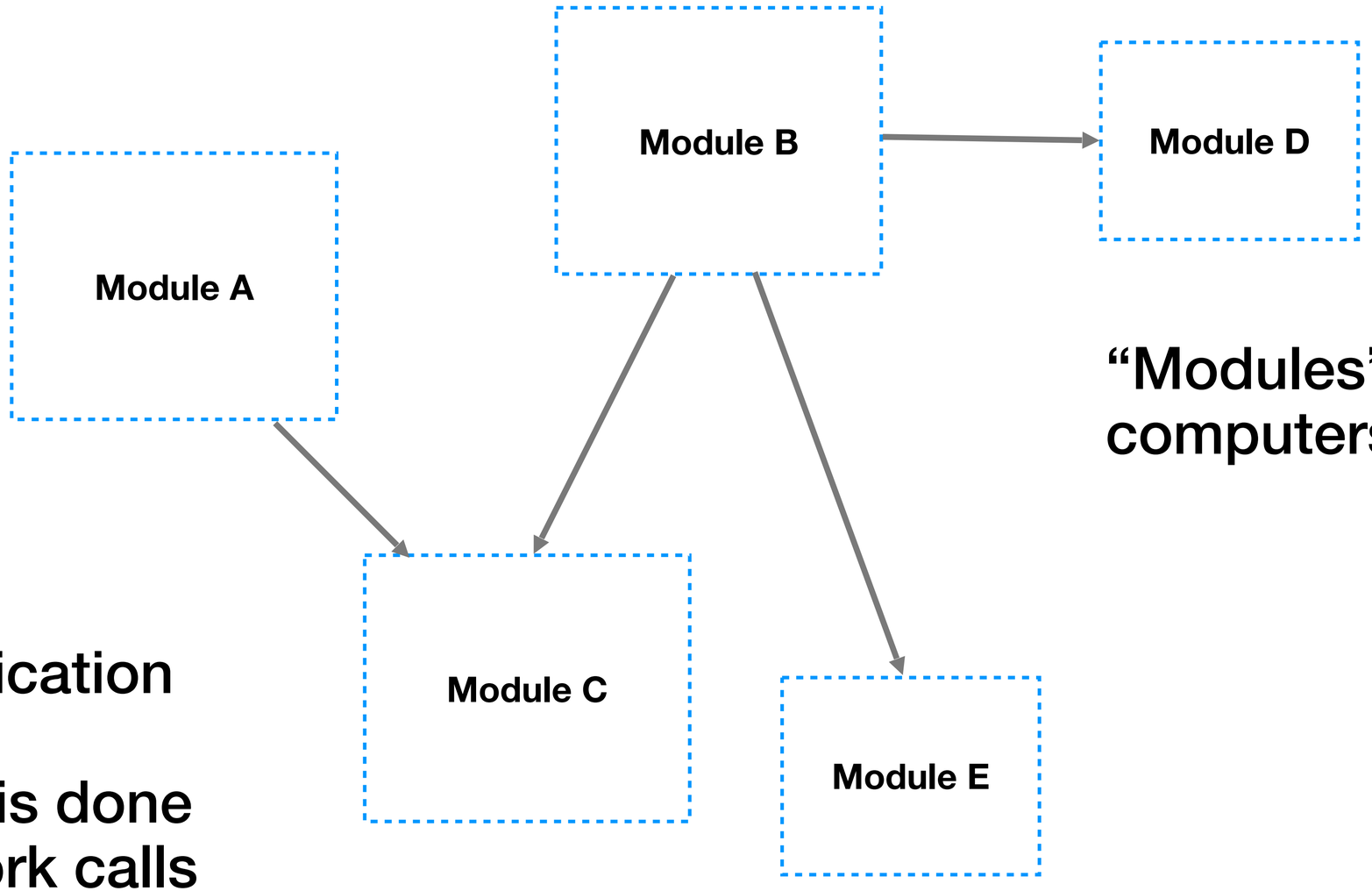


AND MICROSERVICES?



“Modules” run on different computers

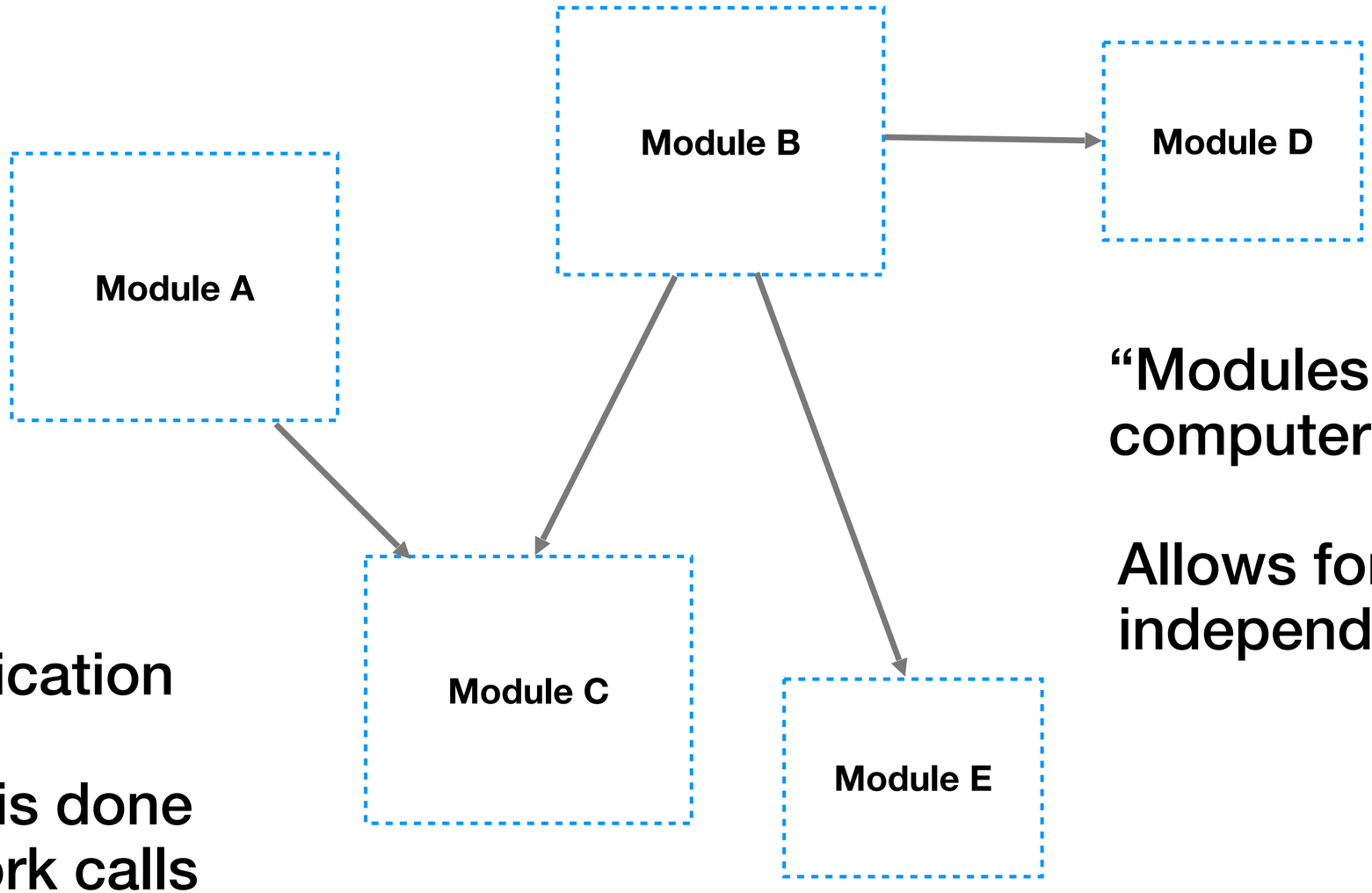
AND MICROSERVICES?



“Modules” run on different computers

Communication between modules is done via network calls

AND MICROSERVICES?



"Modules" run on different computers

Allows for easier independent deployability

Communication between modules is done via network calls

Microservices, when done right, are a form of modular architecture

INFORMATION HIDING

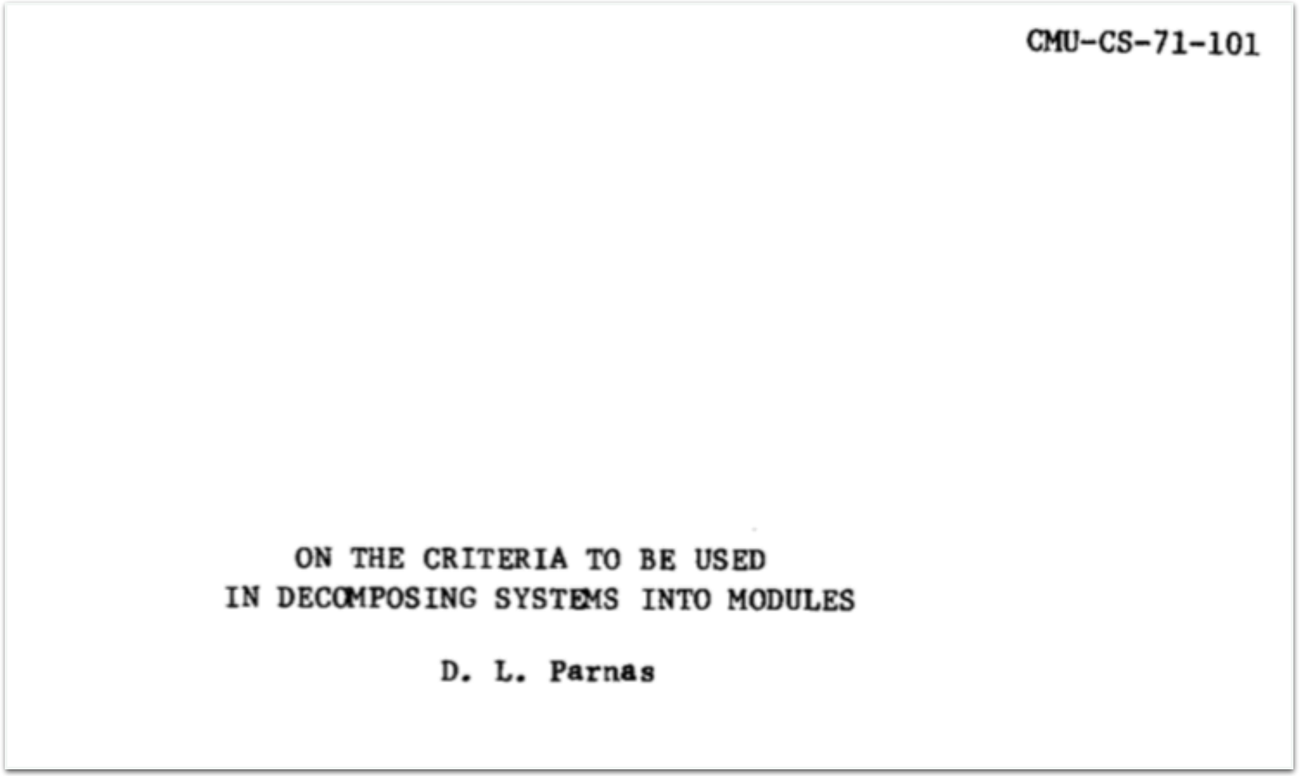
CMU-CS-71-101

ON THE CRITERIA TO BE USED
IN DECOMPOSING SYSTEMS INTO MODULES

D. L. Parnas

http://repository.cmu.edu/cgi/viewcontent.cgi?article=2979&context=compsci

INFORMATION HIDING



CMU-CS-71-101

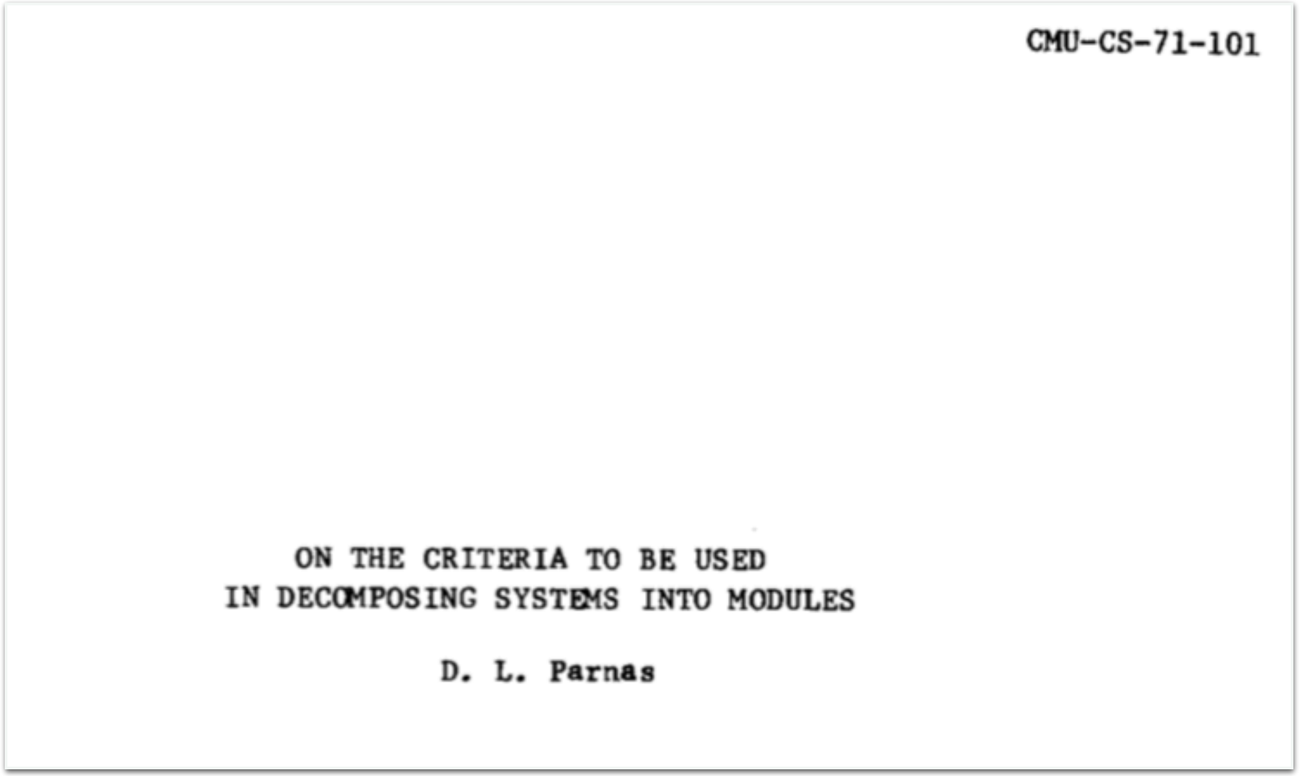
ON THE CRITERIA TO BE USED
IN DECOMPOSING SYSTEMS INTO MODULES

D. L. Parnas

Published in 1971

http://repository.cmu.edu/cgi/viewcontent.cgi?article=2979&context=compsci

INFORMATION HIDING


CMU-CS-71-101

ON THE CRITERIA TO BE USED
IN DECOMPOSING SYSTEMS INTO MODULES

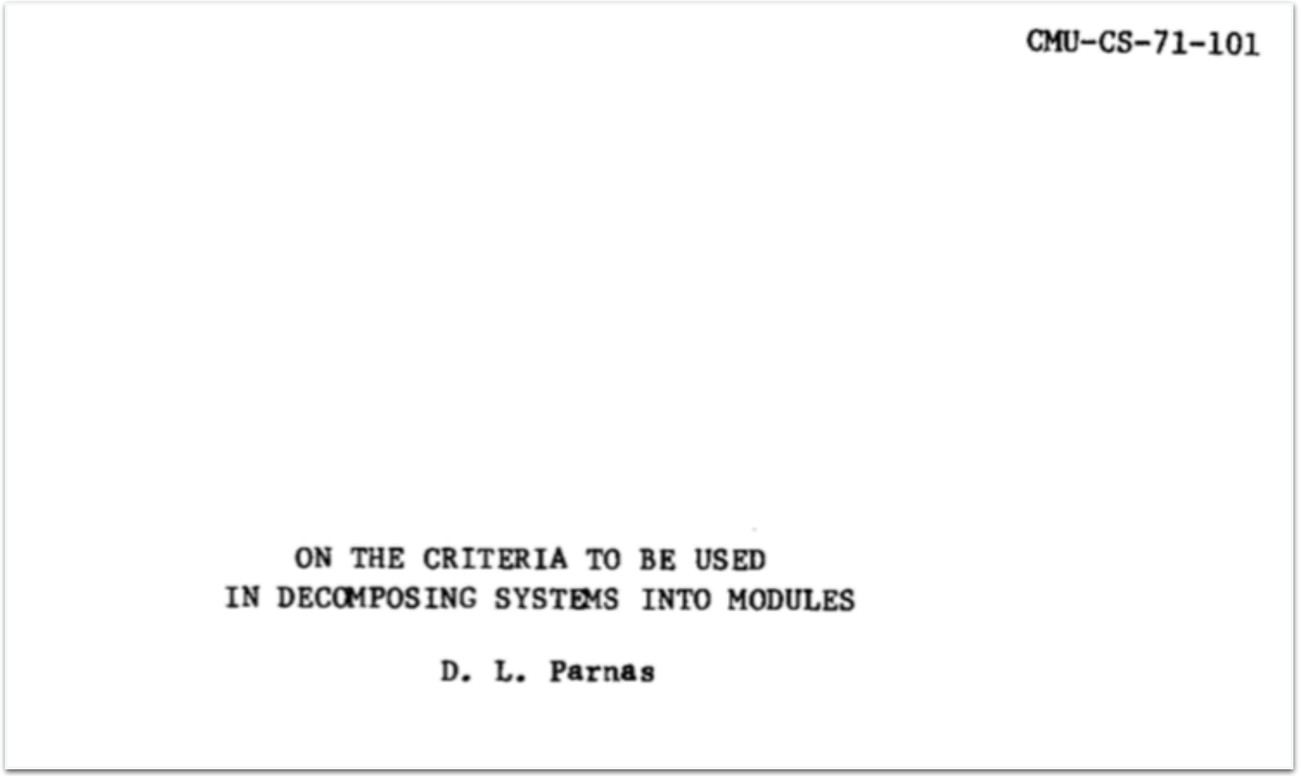
D. L. Parnas

Published in 1971

Looked at how best to define module boundaries

http://repository.cmu.edu/cgi/viewcontent.cgi?article=2979&context=compsci

INFORMATION HIDING


CMU-CS-71-101

ON THE CRITERIA TO BE USED
IN DECOMPOSING SYSTEMS INTO MODULES

D. L. Parnas

Published in 1971

Looked at how best to define module boundaries

Found that “information hiding” worked best

http://repository.cmu.edu/cgi/viewcontent.cgi?article=2979&context=compsci

INFORMATION HIDING AND MICROSERVICES

the morning paper

a random walk through Computer Science research, by Adrian Colyer

ABOUT ARCHIVES INFOQ QR EDITIONS SEARCH SUBSCRIBE TAGS PRIVACY

On the criteria to be used in decomposing systems into modules

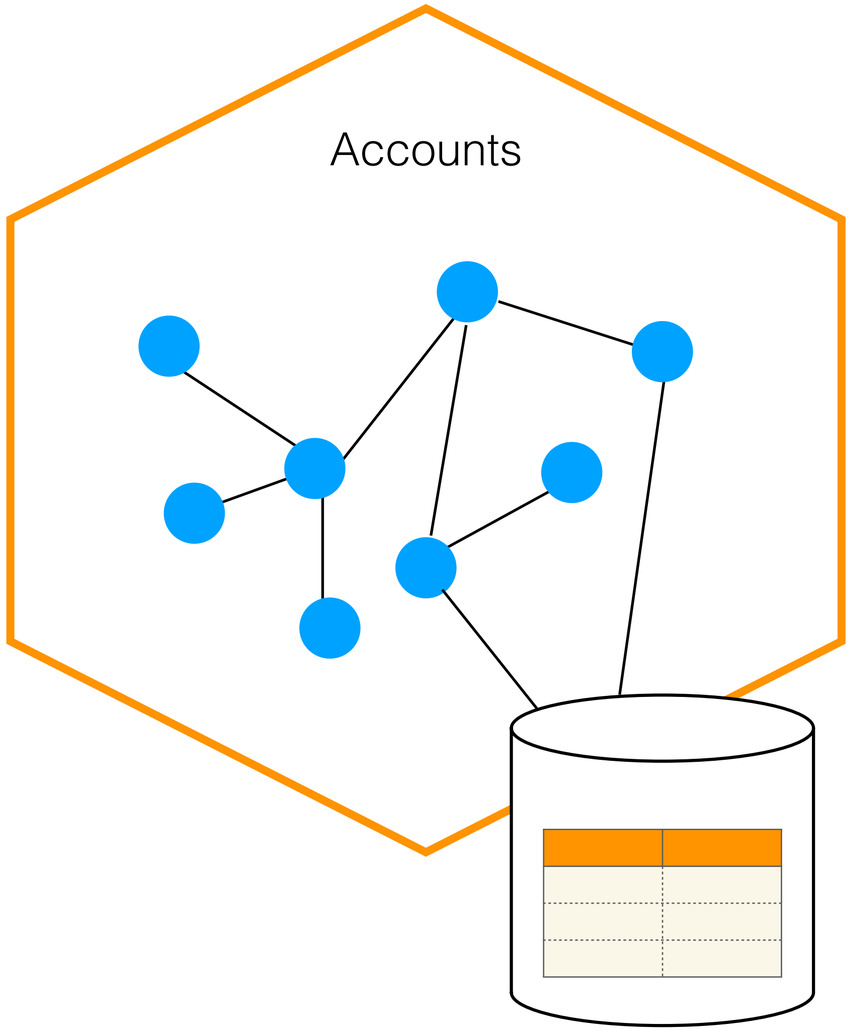
SEPTEMBER 5, 2016

On the criteria to be used in decomposing systems into modules *David L Parnas, 1971*

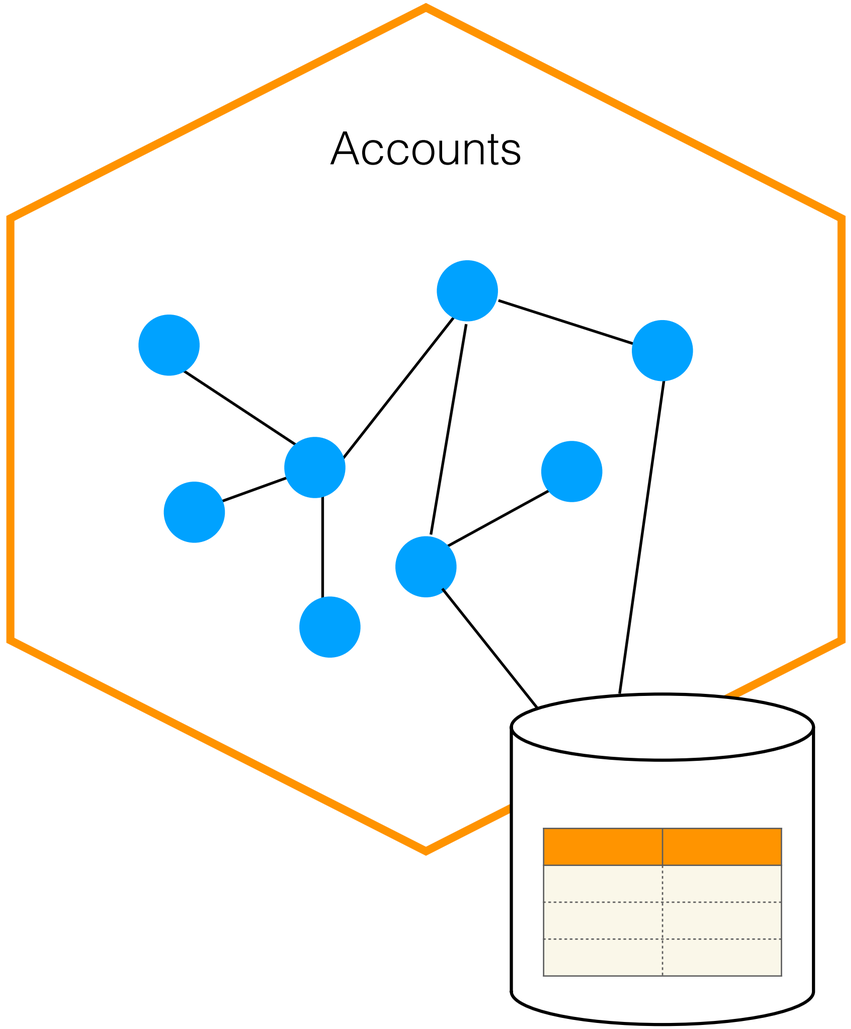
Welcome back to a new term of The Morning Paper! I thought I'd kick things off by revisiting a few of my favourite papers from when I very first started this exercise just over two years ago. At that time I wasn't posting blog summaries of the papers, so it's nice to go back and fill in that gap (blog posts started in October of 2014). Plus, revisiting some of the classics once every couple of years seems like a good idea – changing external circumstances can make them feel fresh again every time you read them.

https://blog.acolyer.org/2016/09/05/on-the-criteria-to-be-used-in-decomposing-systems-into-modules/

NOT HIDING?

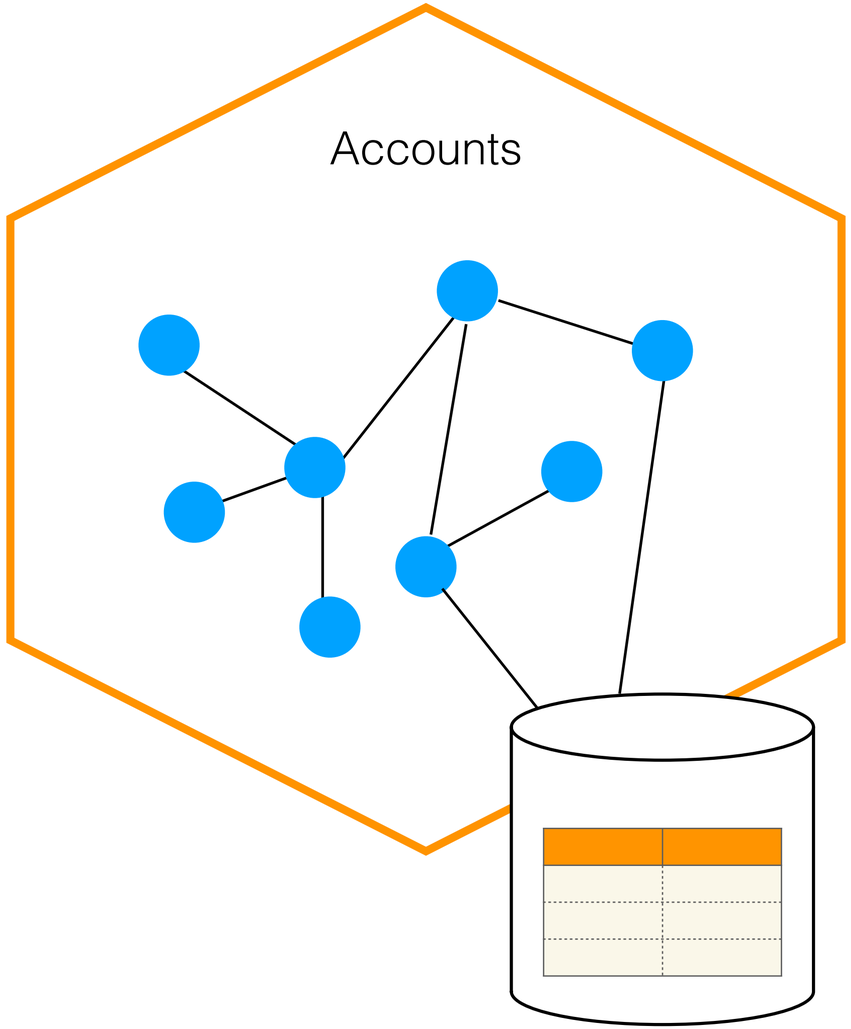


NOT HIDING?



If an upstream consumer can reach into your internal implementation..

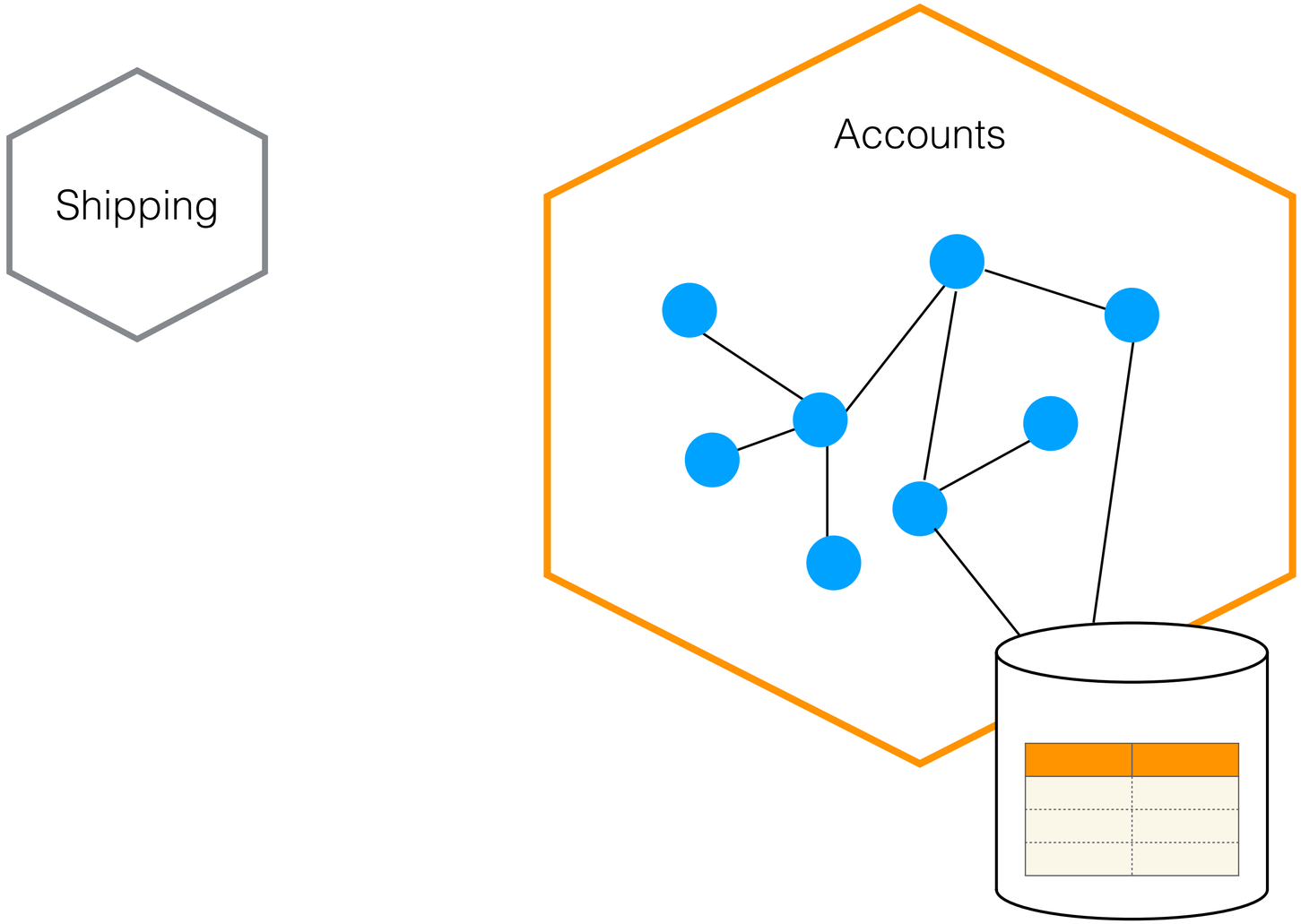
NOT HIDING?



If an upstream consumer can reach into your internal implementation..

…then you can’t change this implementation without breaking the consumer

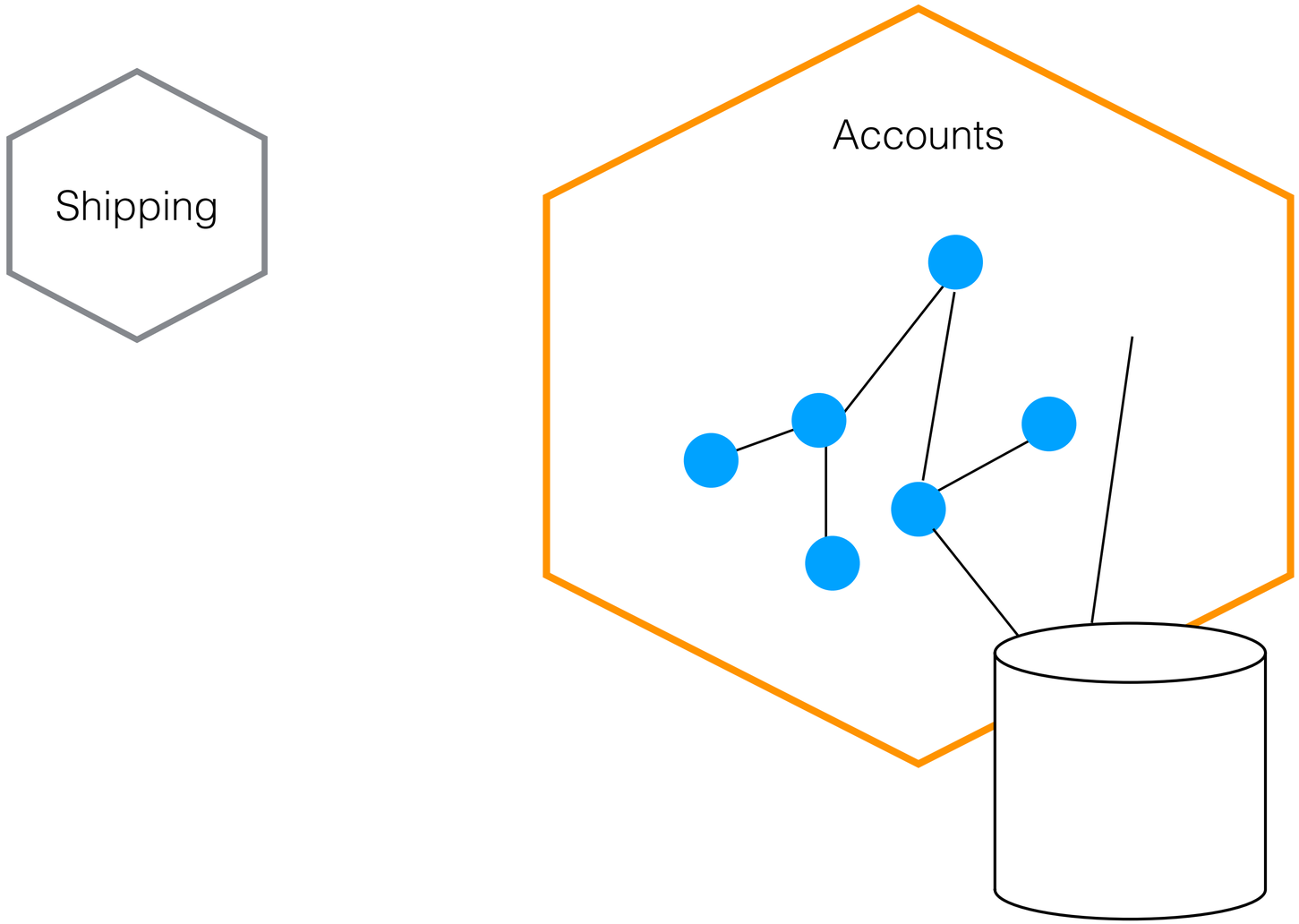
NOT HIDING?



If an upstream consumer can reach into your internal implementation..

…then you can't change this implementation without breaking the consumer

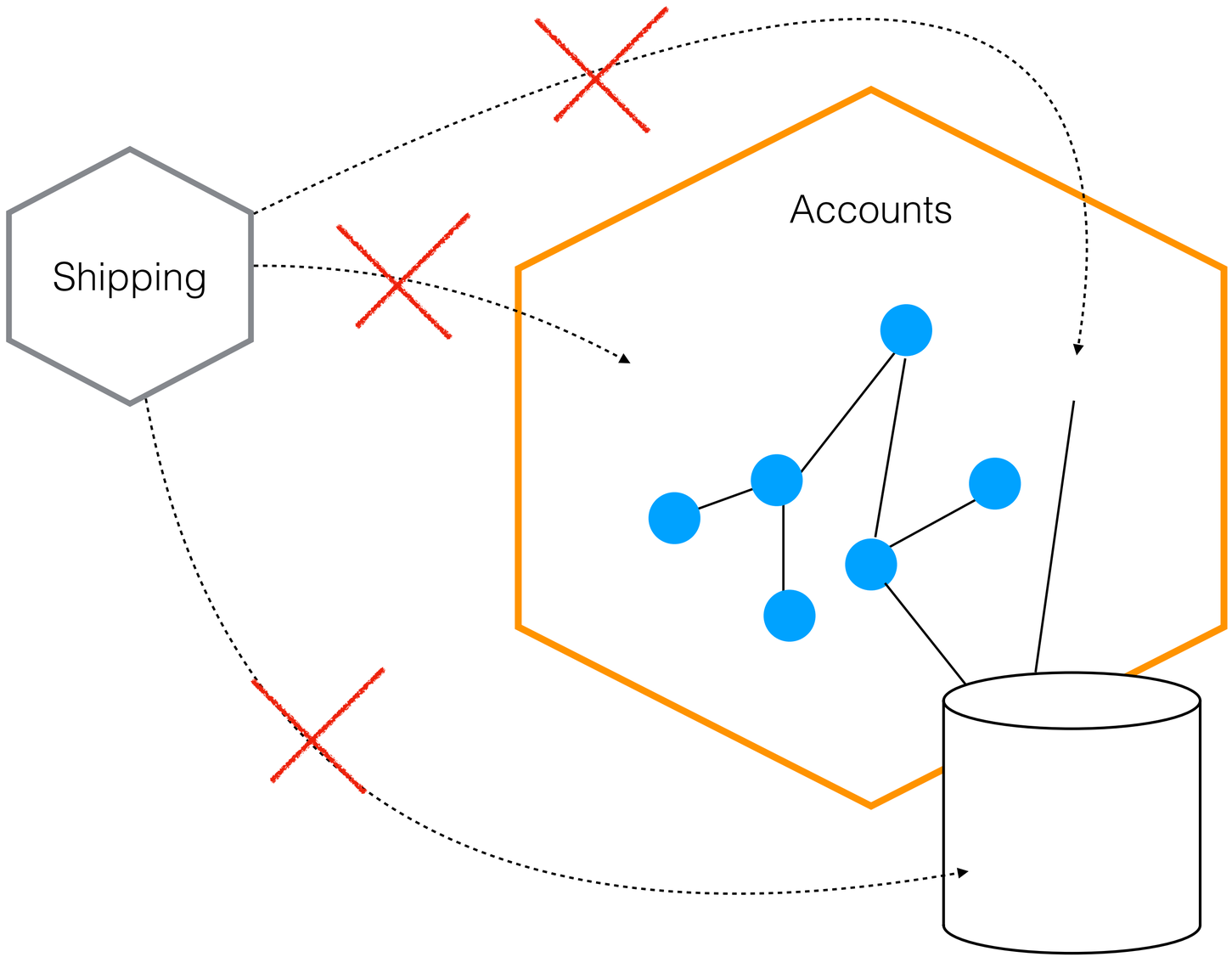
NOT HIDING?



If an upstream consumer can reach into your internal implementation..

…then you can’t change this implementation without breaking the consumer

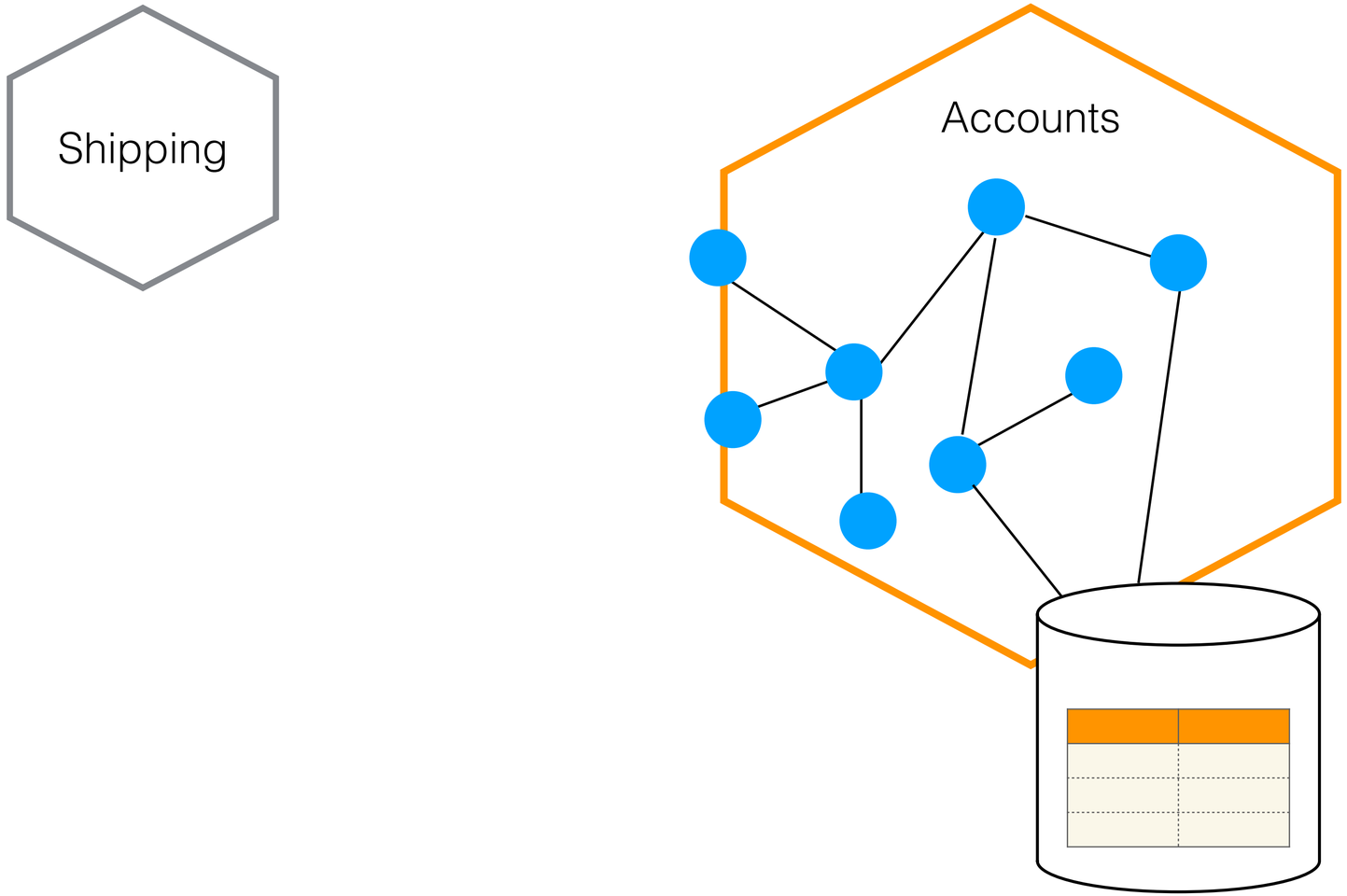
NOT HIDING?



If an upstream consumer can reach into your internal implementation..

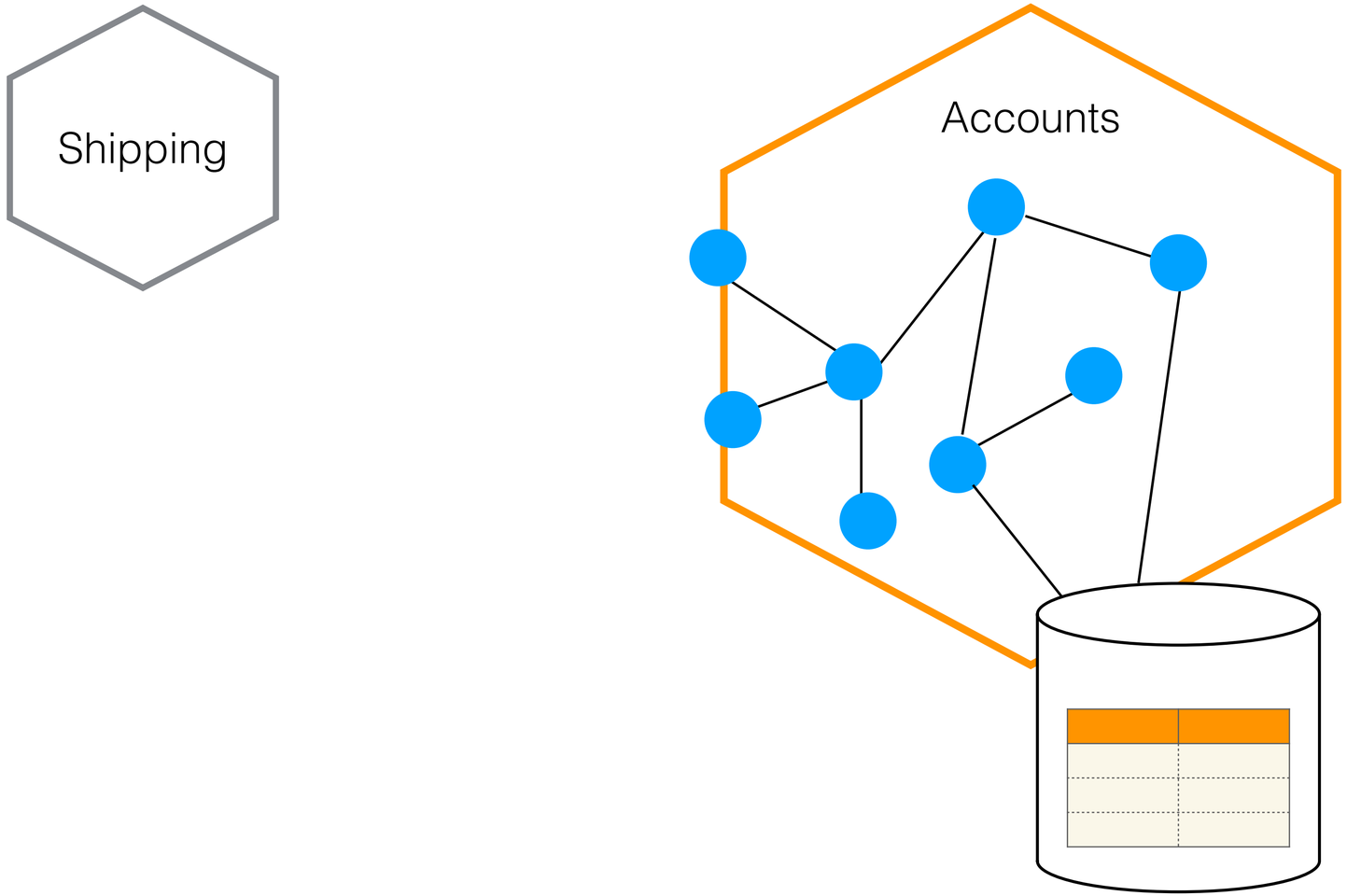
…then you can't change this implementation without breaking the consumer

HIDING!

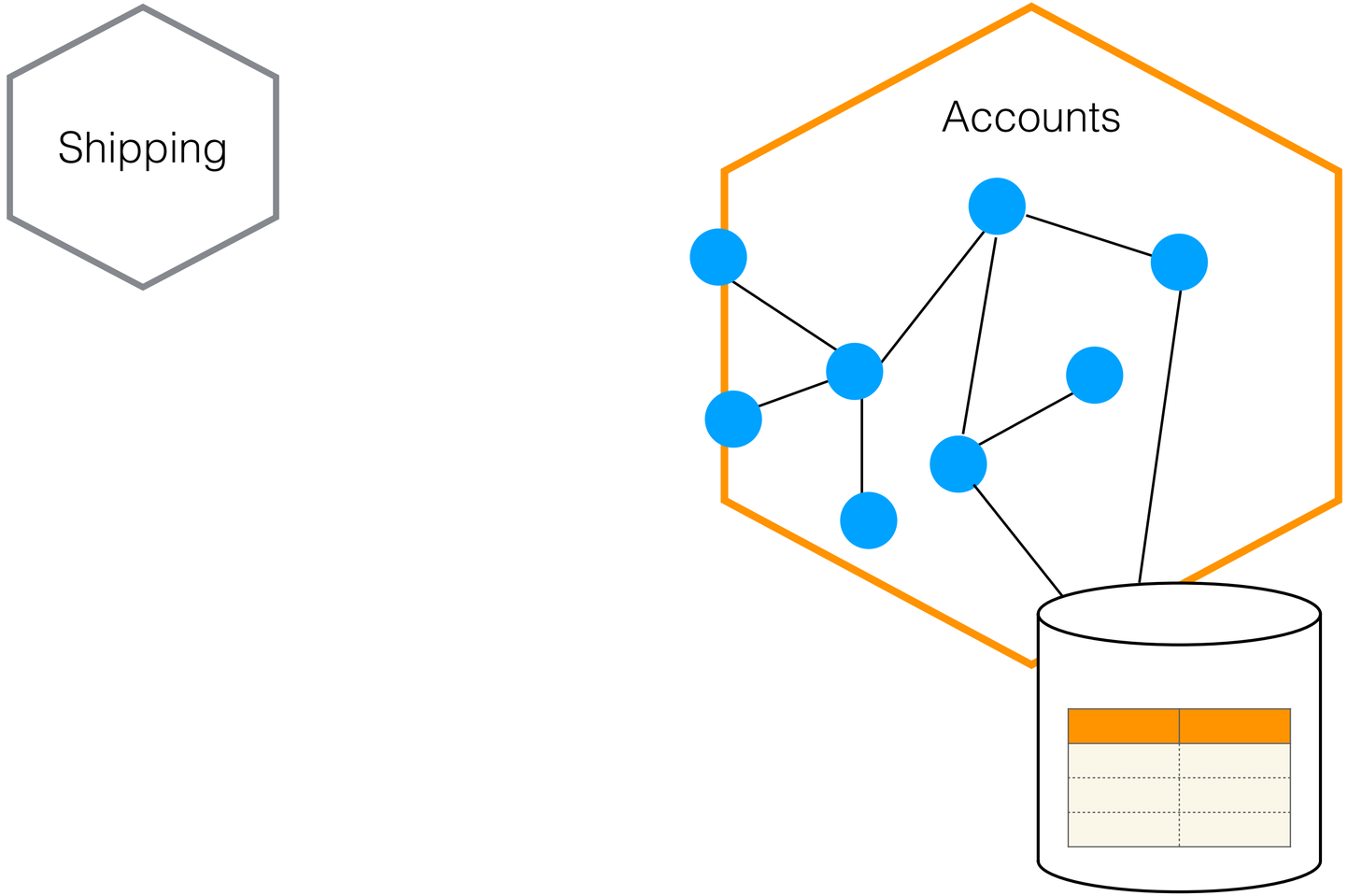


HIDING!

Hide your secrets!



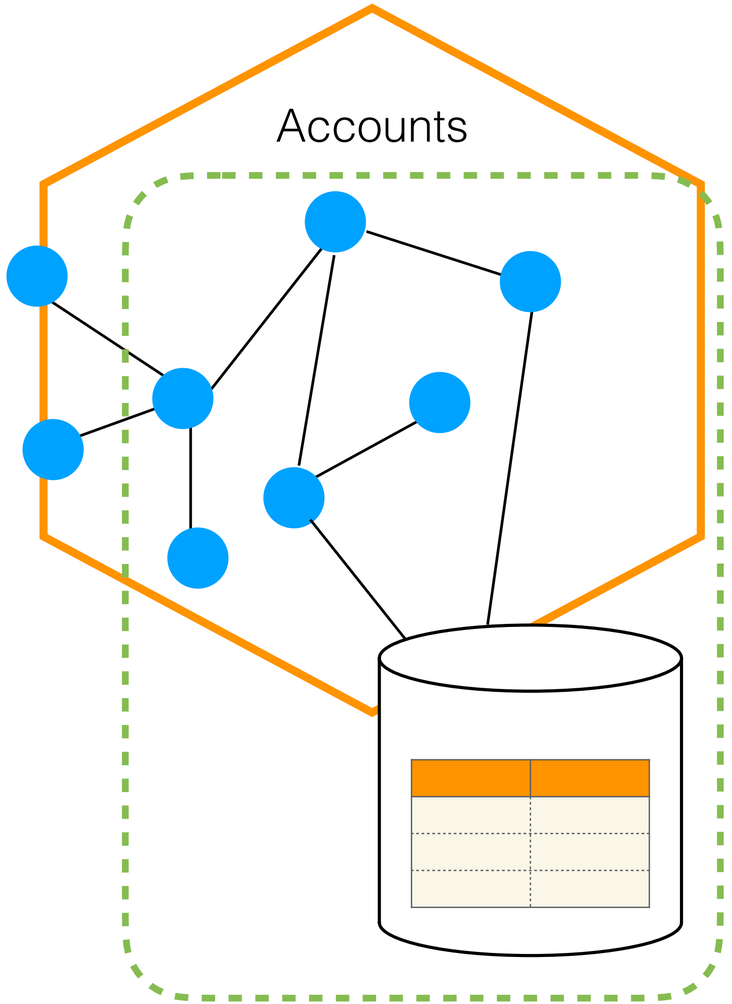
HIDING!



Hide your secrets!

Be explicit about what is shared, and what is hidden

HIDING!



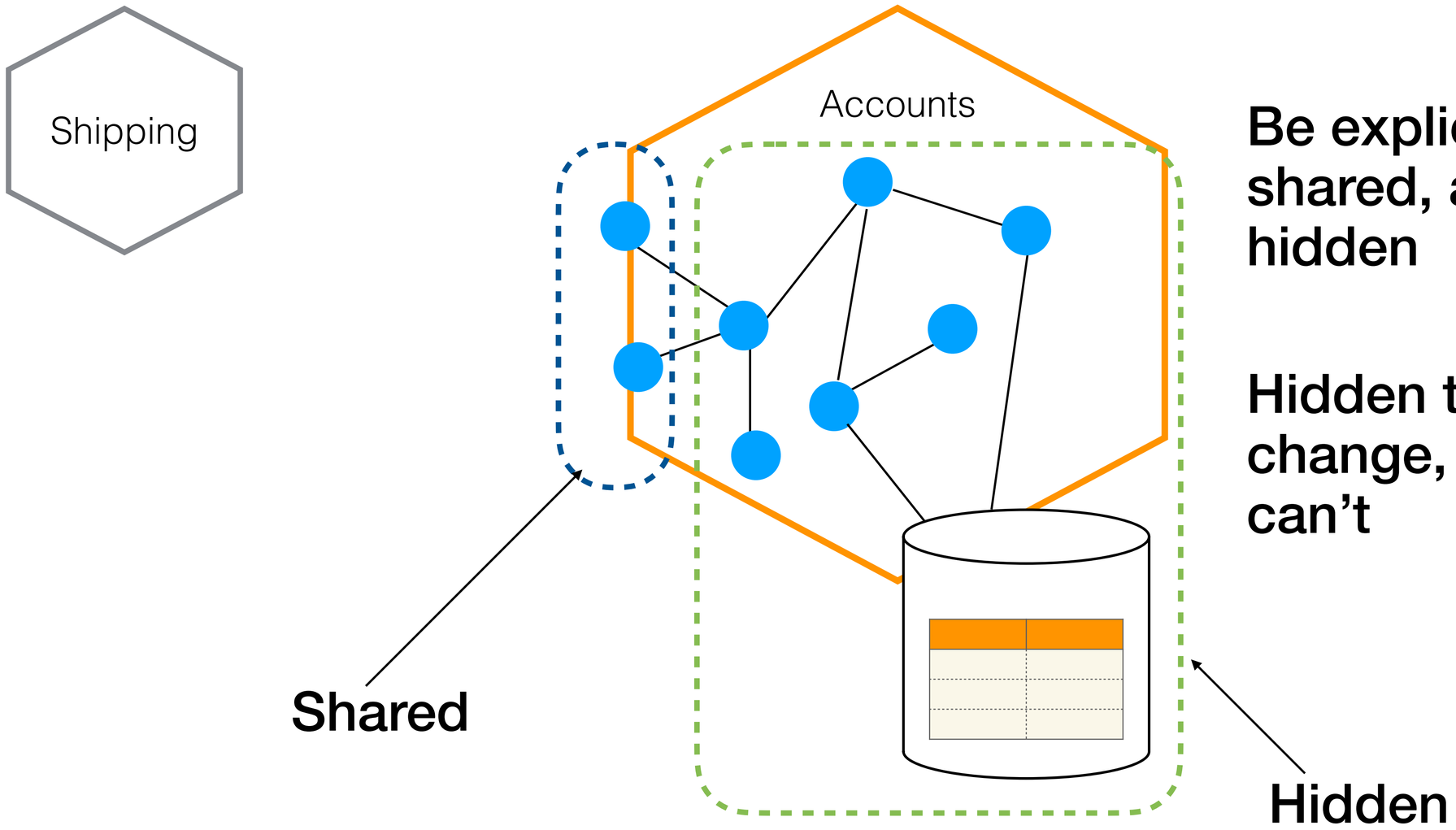
Hide your secrets!

Be explicit about what is shared, and what is hidden

Hidden things can change, shared things can't

Hidden

HIDING!

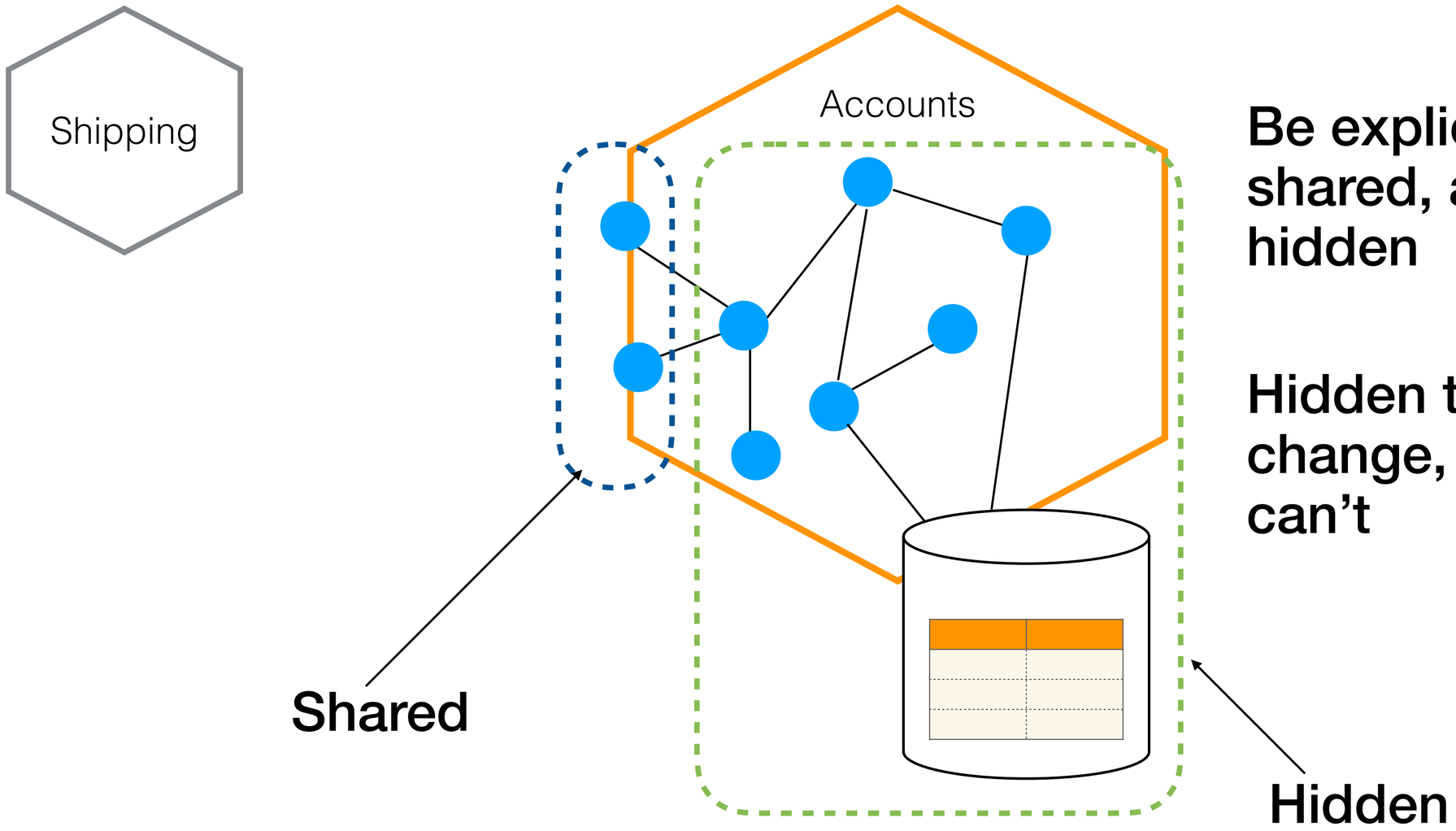


Hide your secrets!

Be explicit about what is shared, and what is hidden

Hidden things can change, shared things can't

HIDING!

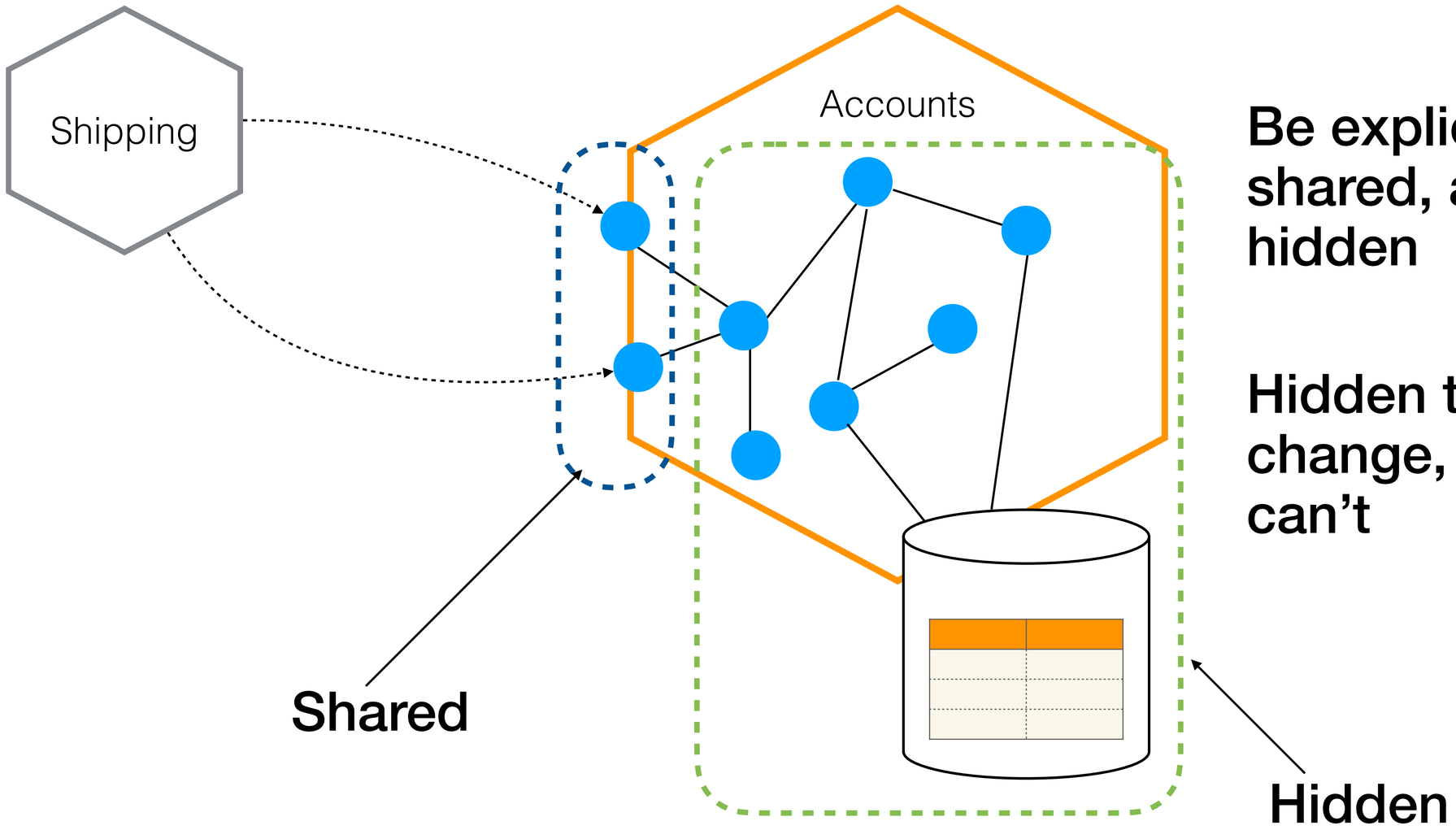


Hide your secrets!

Be explicit about what is shared, and what is hidden

Hidden things can change, shared things can't

HIDING!



Hide your secrets!

Be explicit about what is shared, and what is hidden

Hidden things can change, shared things can't

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

JSON

```
{
  “id” : 123,
  “name” : “sam”,
  “age” : 15
}
```

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

JSON

```
{
  “id” : 123,
  “name” : “sam”,
  “age” : 15
}
```

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

JSON

```
{
  “id” : 123,
  “name” : “sam”,
  “age” : 15
}
```

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

JSON

```
{
  “id” : 123,
  “name” : “sam”,
  “age” : 15
}
```

Code

```
public class Customer {
  private int id;
  private String name;
  private LocalDate dob;
  …
}
```

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

JSON

```
{
  “id” : 123,
  “name” : “sam”,
  “age” : 15
}
```

Code

```
public class Customer {
  private int id;
  private String name;
  private LocalDate dob;
  …
}
```

ACCIDENTAL BREAKAGE

Code

```
public class Customer {
  private int id;
  private String name;
  private int age;
  …
}
```

JSON

```
{
  “id” : 123,
  “name” : “sam”,
  “age” : 15
}
```

Code

```
public class Customer {
  private int id;
  private String name;
  private LocalDate dob;
  …
}
```

JSON

```
{
  “id” : 123,
  “name” : “sam”,
  “dob” : 15/02/1974
}
```

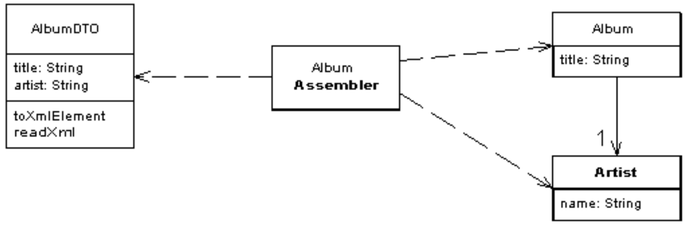
DATA TRANSFER OBJECT

| P of EAA Catalog |

Data Transfer Object

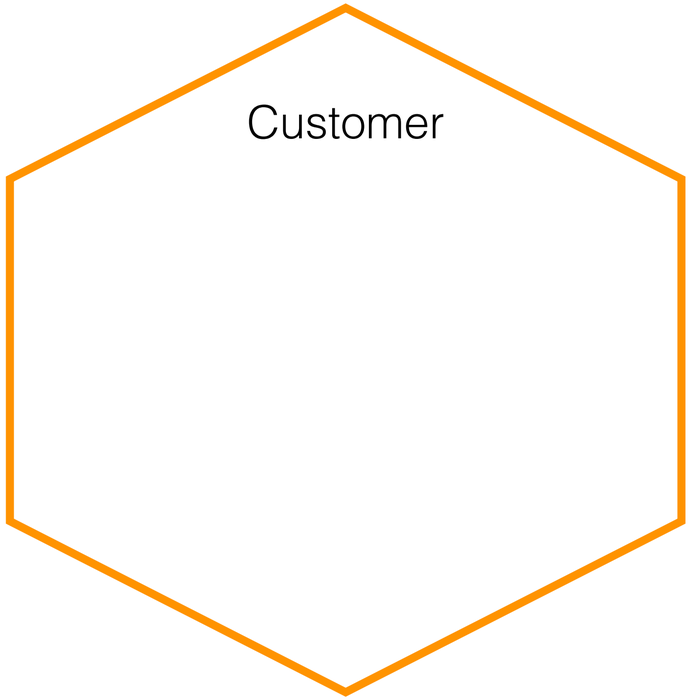
An object that carries data between processes in order to reduce the number of method calls.

For a full description see P of EAA page **401**



When you're working with a remote interface, such as Remote Facade (388), each call to it is expensive. As a result you need to reduce the number of calls, and that means that you need to transfer more data with each call. One way to do this is to use lots of parameters. However, this is often awkward to program - indeed, it's often impossible with languages such as Java that return only a single value.

The solution is to create a Data Transfer Object that can hold all the data for the call. It needs to be serializable to go across the connection. Usually an assembler is used on the server side to transfer data between the DTO and any domain objects.



https://martinfowler.com/eaaCatalog/dataTransferObject.html

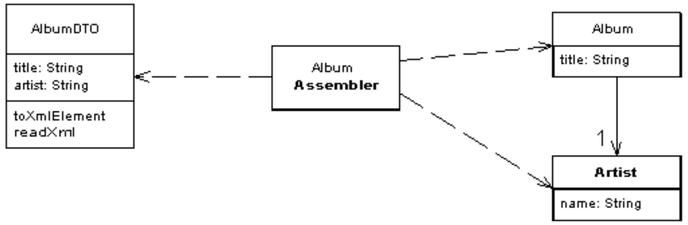
DATA TRANSFER OBJECT

| P of EAA Catalog |

Data Transfer Object

An object that carries data between processes in order to reduce the number of method calls.

For a full description see P of EAA page **401**



When you're working with a remote interface, such as Remote Facade (388), each call to it is expensive. As a result you need to reduce the number of calls, and that means that you need to transfer more data with each call. One way to do this is to use lots of parameters. However, this is often awkward to program - indeed, it's often impossible with languages such as Java that return only a single value.

The solution is to create a Data Transfer Object that can hold all the data for the call. It needs to be serializable to go across the connection. Usually an assembler is used on the server side to transfer data between the DTO and any domain objects.

Customer

```
public class Customer {
  private int id;
  private String name;
  private LocalDate dob;
  …
}
```

https://martinfowler.com/eaaCatalog/dataTransferObject.html

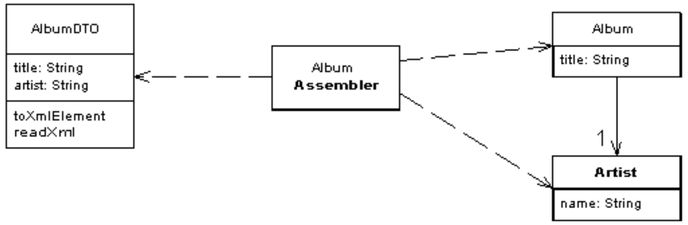
DATA TRANSFER OBJECT

| P of EAA Catalog |

Data Transfer Object

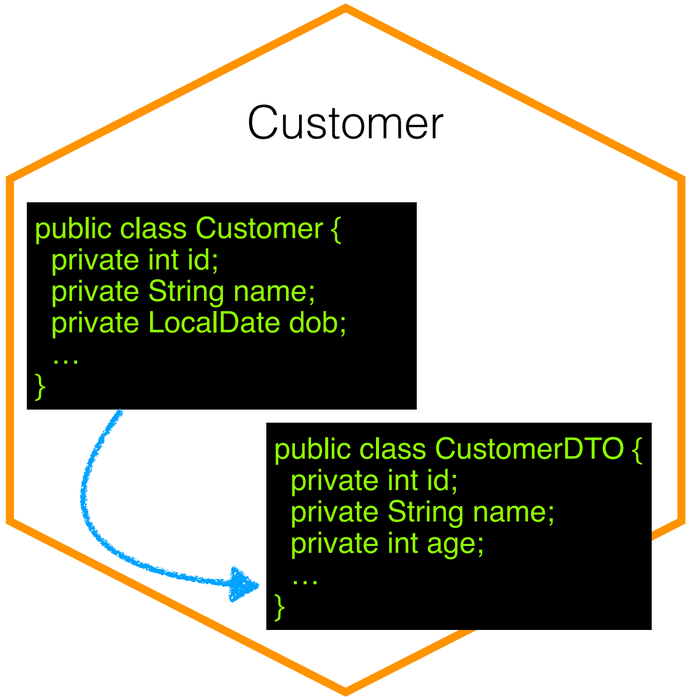
An object that carries data between processes in order to reduce the number of method calls.

For a full description see P of EAA page **401**



When you're working with a remote interface, such as Remote Facade (388), each call to it is expensive. As a result you need to reduce the number of calls, and that means that you need to transfer more data with each call. One way to do this is to use lots of parameters. However, this is often awkward to program - indeed, it's often impossible with languages such as Java that return only a single value.

The solution is to create a Data Transfer Object that can hold all the data for the call. It needs to be serializable to go across the connection. Usually an assembler is used on the server side to transfer data between the DTO and any domain objects.



https://martinfowler.com/eaaCatalog/dataTransferObject.html

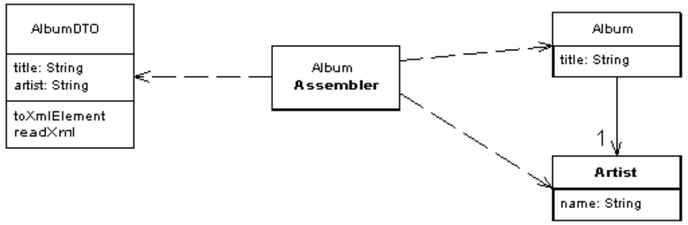
DATA TRANSFER OBJECT

| P of EAA Catalog |

Data Transfer Object

An object that carries data between processes in order to reduce the number of method calls.

For a full description see P of EAA page **401**



When you're working with a remote interface, such as Remote Facade (388), each call to it is expensive. As a result you need to reduce the number of calls, and that means that you need to transfer more data with each call. One way to do this is to use lots of parameters. However, this is often awkward to program - indeed, it's often impossible with languages such as Java that return only a single value.

The solution is to create a Data Transfer Object that can hold all the data for the call. It needs to be serializable to go across the connection. Usually an assembler is used on the server side to transfer data between the DTO and any domain objects.

Customer

```
public class Customer {
  private int id;
  private String name;
  private LocalDate dob;
  …
}
```

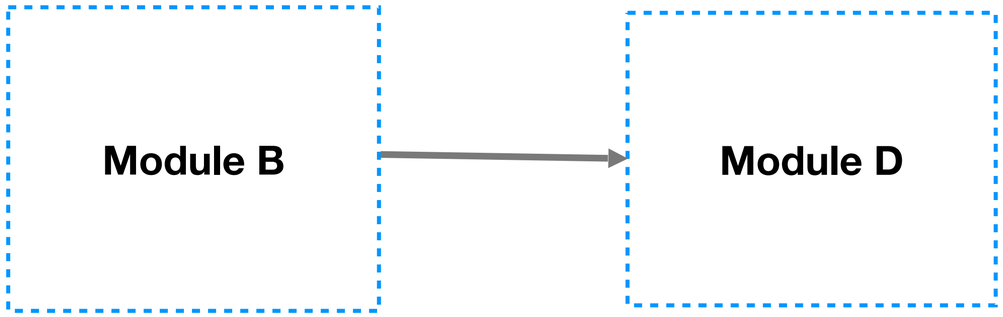
```
public class CustomerDTO {
  private int id;
  private String name;
  private int age;
  …
}
```

JSON

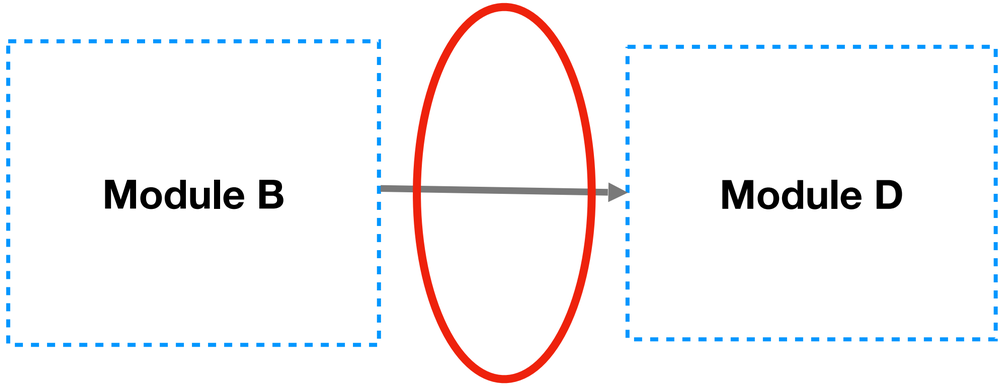
```
{
  “id” : 123,
  “name” : “sam”,
  “age” : 15
}
```

https://martinfowler.com/eaaCatalog/dataTransferObject.html

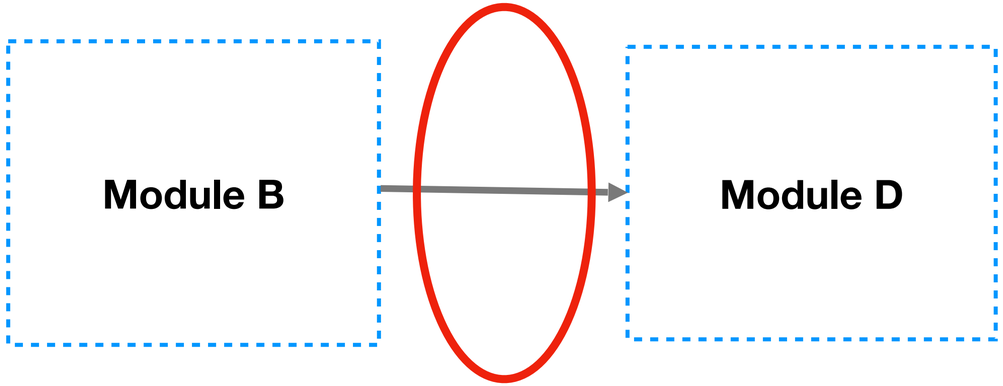
EXPLICIT CONTRACTS?



EXPLICIT CONTRACTS?

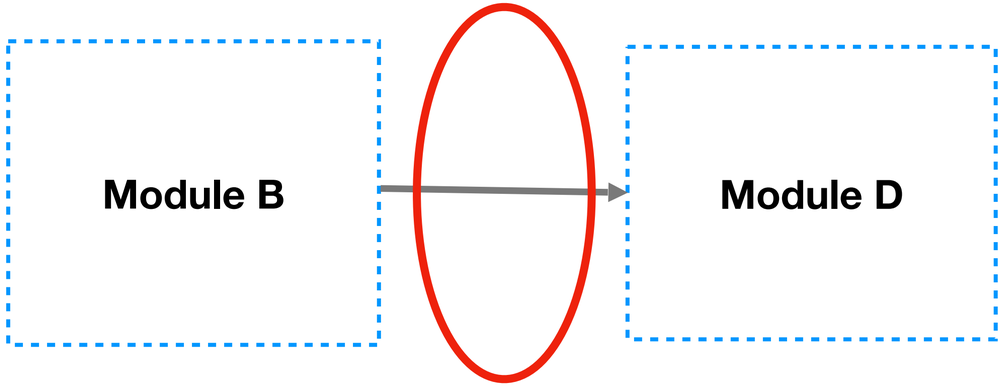


EXPLICIT CONTRACTS?



Need an explicit understanding of the assumptions of consumers

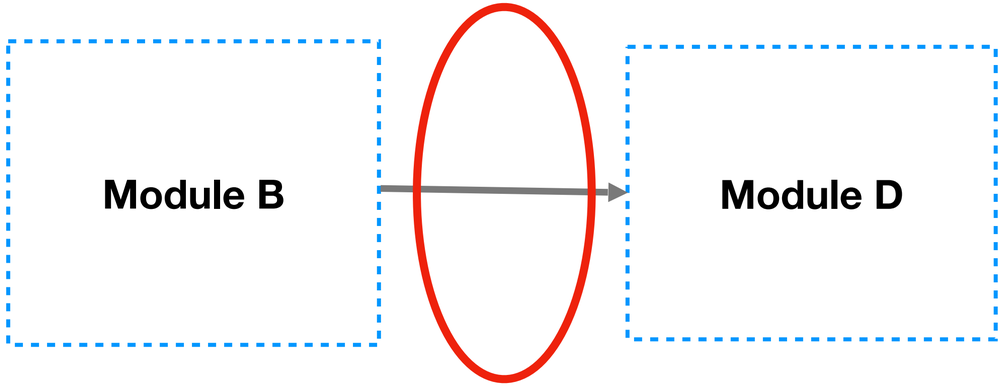
EXPLICIT CONTRACTS?



Need an explicit understanding of the assumptions of consumers

To help us understand what can change easily, and what cannot

EXPLICIT CONTRACTS?



Need an explicit understanding of the assumptions of consumers

To help us understand what can change easily, and what cannot

Schemas can help!

Explicit Schemas FTW!

SCHEMAS

JSON Schema

NEW DRAFT PUBLISHED!

The current version is 2019-09!

JSON Schema is a vocabulary that allows you to **annotate** and **validate** JSON documents.

Advantages

JSON Schema

- Describes your existing data format(s).
- Provides clear human- and machine- readable documentation.
- Validates data which is useful for:
  - Automated testing.
  - Ensuring quality of client submitted data.

JSON Hyper-Schema

- Make any JSON format a hypermedia format with no constraints on document structure
- Allows use of URI Templates with instance data
- Describe client data for use with links using JSON Schema.
- Recognizes collections and collection items.

Project Status

16 September 2019: Draft 2019-09 (formerly known as draft-08) has been published!

The IETF document IDs are of the form `draft-handrews-*-02`. We are now using dates for meta-schemas, which are what implementations should use to determine behavior, so we will usually refer to `2019-09` (without the word "draft") on this web site.

https://json-schema.org/

```
syntax = "proto2";

package tutorial;

option java_package = "com.example.tutorial";
option java_outer_classname = "AddressBookProtos";

message Person {
  required string name = 1;
  required int32 id = 2;
  optional string email = 3;

  enum PhoneType {
    MOBILE = 0;
    HOME = 1;
    WORK = 2;
  }

  message PhoneNumber {
    required string number = 1;
    optional PhoneType type = 2 [default = HOME];
  }

  repeated PhoneNumber phones = 4;
}

message AddressBook {
  repeated Person people = 1;
}
```

https://developers.google.com/protocol-buffers/

CHECKING SCHEMA COMPATIBILITY

Protolock

PASSED | godoc reference | docker build passing

Protocol Buffer companion tool. Track your .proto files and prevent changes to messages and services which impact API compatibility.

Source: **github.com/nilslice/protolock**

DEMO

Step 1: Define a .proto

Either use the demo `demo.proto` or write your own here:

demo.proto

```
    UNKNOWN = 0;
    SERVING = 1;
    NOT_SERVING = 2;
    SERVICE_UNKNOWN = 3;
  }
  ServingStatus status = 1;
}

service Health {
  rpc Check(HealthCheckRequest) returns (HealthCheckResponse);
  rpc Watch(HealthCheckRequest) returns (stream HealthCheckResponse);
}
```

Step 2: Initialize a `proto.lock` file:

`$ protolock init` (click to lock your .proto definition)

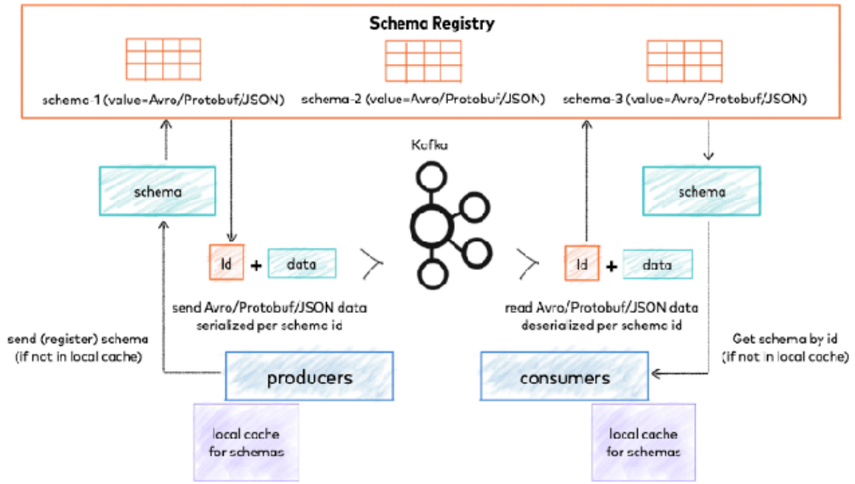
https://protolock.dev/

SCHEMA CHECKING

Schema Management

Confluent Schema Registry provides a serving layer for your metadata. It provides a RESTful interface for storing and retrieving your Avro®, JSON Schema, and Protobuf schemas. It stores a versioned history of all schemas based on a specified subject name strategy, provides multiple compatibility settings and allows evolution of schemas according to the configured compatibility settings and expanded support for these schema types. It provides serializers that plug into Apache Kafka® clients that handle schema storage and retrieval for Kafka messages that are sent in any of the supported formats.

Schema Registry lives outside of and separately from your Kafka brokers. Your producers and consumers still talk to Kafka to publish and read data (messages) to topics. Concurrently, they can also talk to Schema Registry to send and retrieve schemas that describe the data models for the messages.



Confluent Schema Registry for storing and retrieving schemas

https://docs.confluent.io/current/schema-registry/index.html

README.md

OpenAPI-diff

Compare two OpenAPI specifications (3.x) and render the difference to HTML plaintext, or Markdown files.

Main Build passing | quality gate passed | maven-central v2.0.0-beta.6 | Slack Join the chat room

docker build automated | docker build passing | version v2.0.0-beta.6

Requirements

- Java 8

Feature

- Supports OpenAPI spec v3.0.
- Depth comparison of parameters, responses, endpoint, http method (GET,POST,PUT,DELETE...)
- Supports swagger api Authorization
- Render difference of property with Expression Language
- HTML & Markdown render

Maven

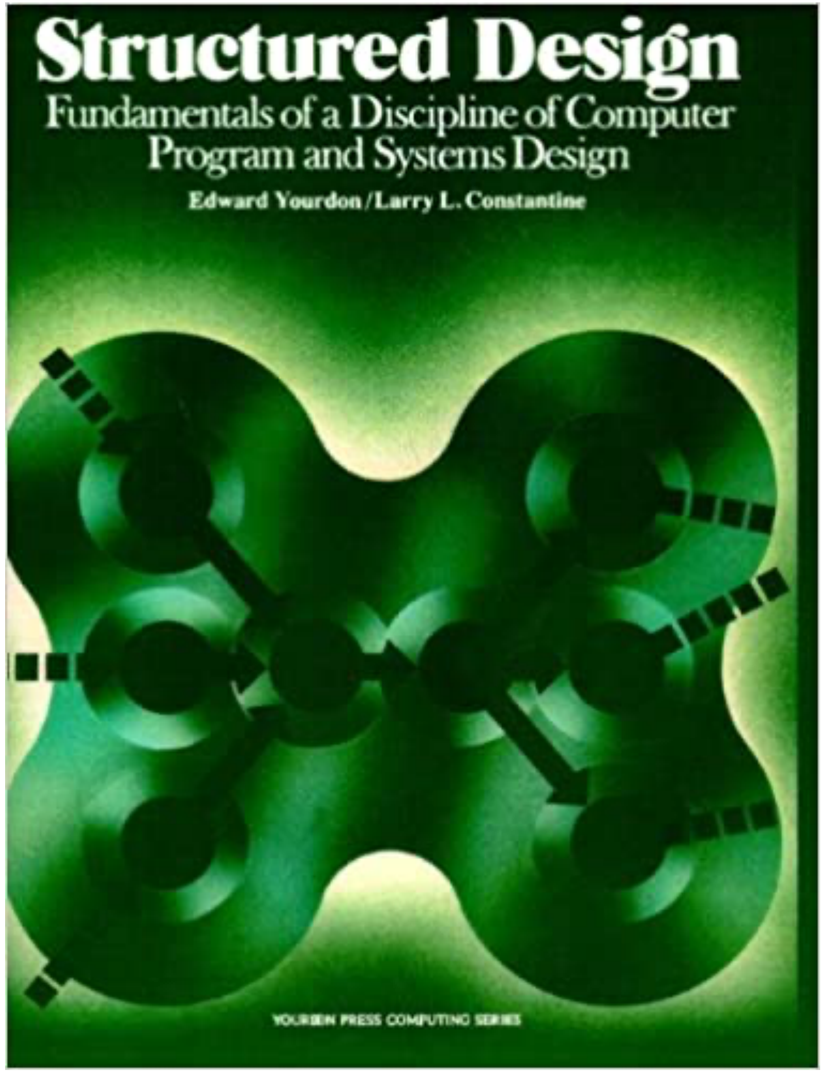
Available on Maven Central

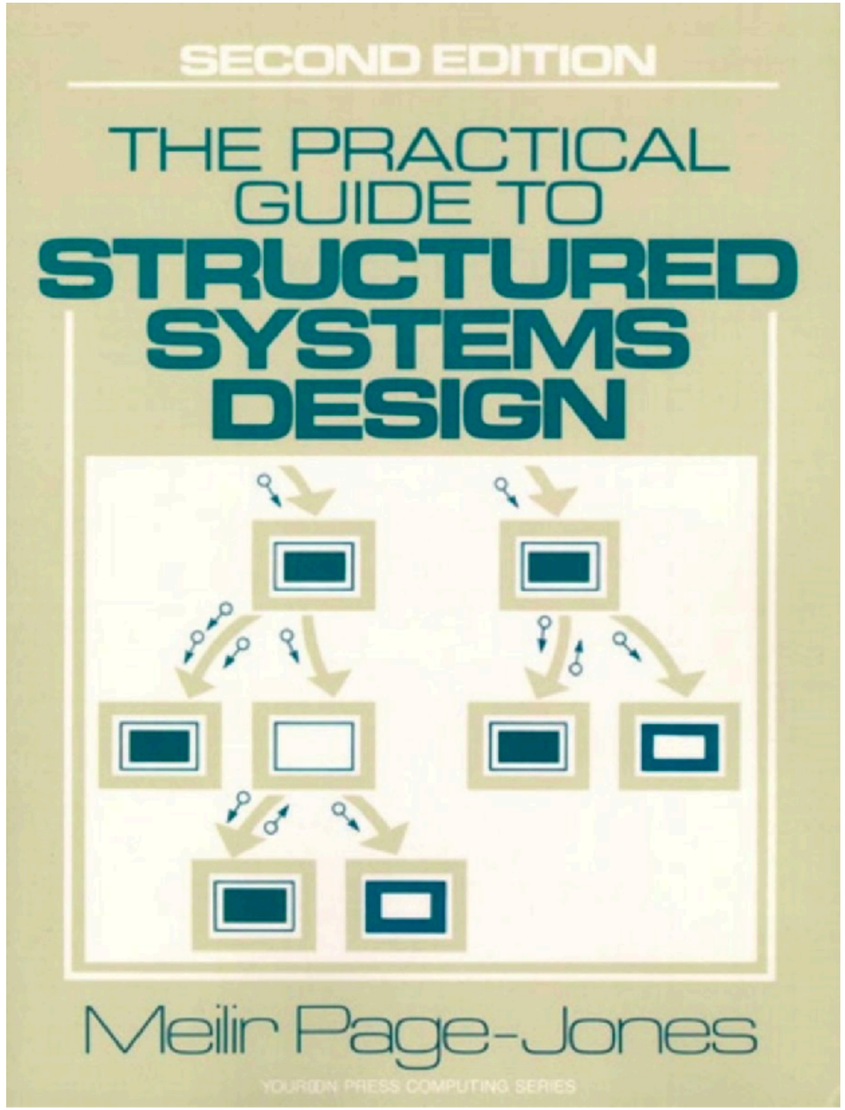
```
<dependency>
  <groupId>org.openapitools.openapidiff</groupId>
  <artifactId>openapi-diff-core</artifactId>
  <version>${openapi-diff-version}</version>
</dependency>
```

https://github.com/OpenAPITools/openapi-diff

And of course testing…

STRUCTURED PROGRAMMING





@samnewman

COUPLING AND COHESION

Coupling

Coupling

Cohesion

Low

Coupling

Cohesion

COUPLING AND COHESION

Low

Coupling

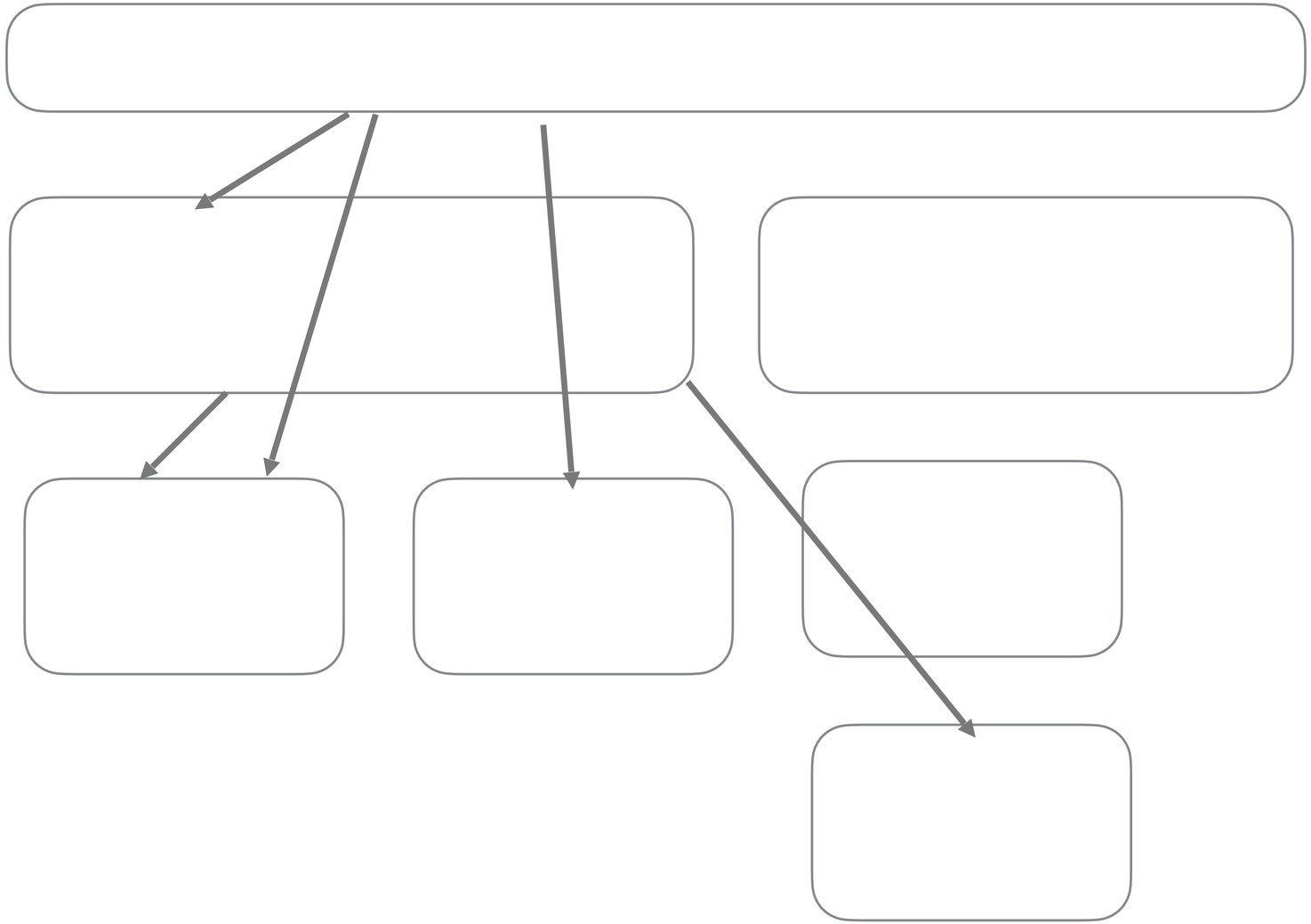
Strong

Cohesion

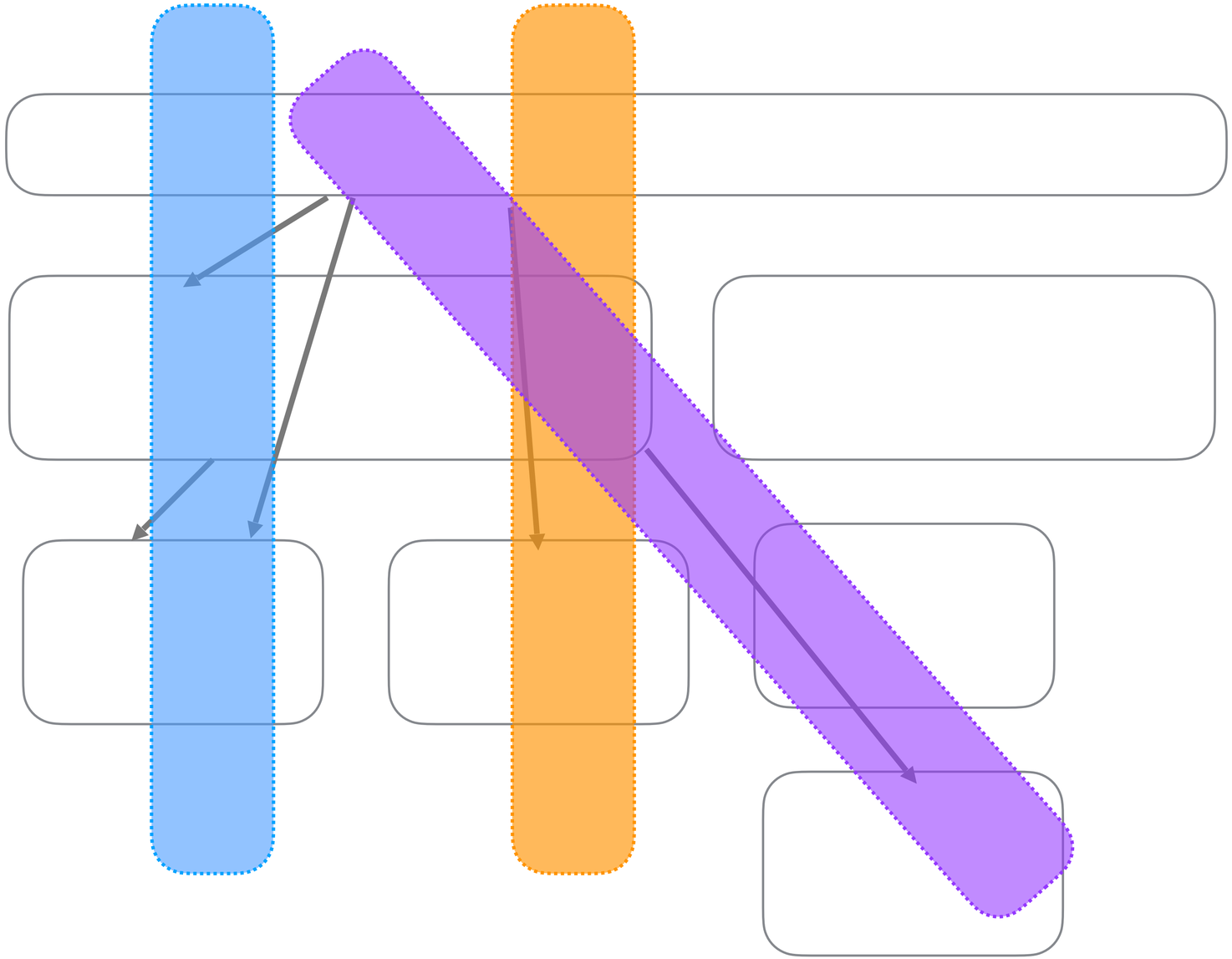
COHESION

The code that changes together, stays together

WEAK COHESION



WEAK COHESION



COUPLING

The degree to which changing one module requires a change in another

CONSTANTINE'S LAW

"A structure is stable if cohesion is strong and coupling is low."

- Albert Endres and Dieter Rombach

TYPES OF COUPLING

TYPES OF COUPLING

Domain

TYPES OF COUPLING

Domain

Common

TYPES OF COUPLING

Domain

Common

Content

TYPES OF COUPLING

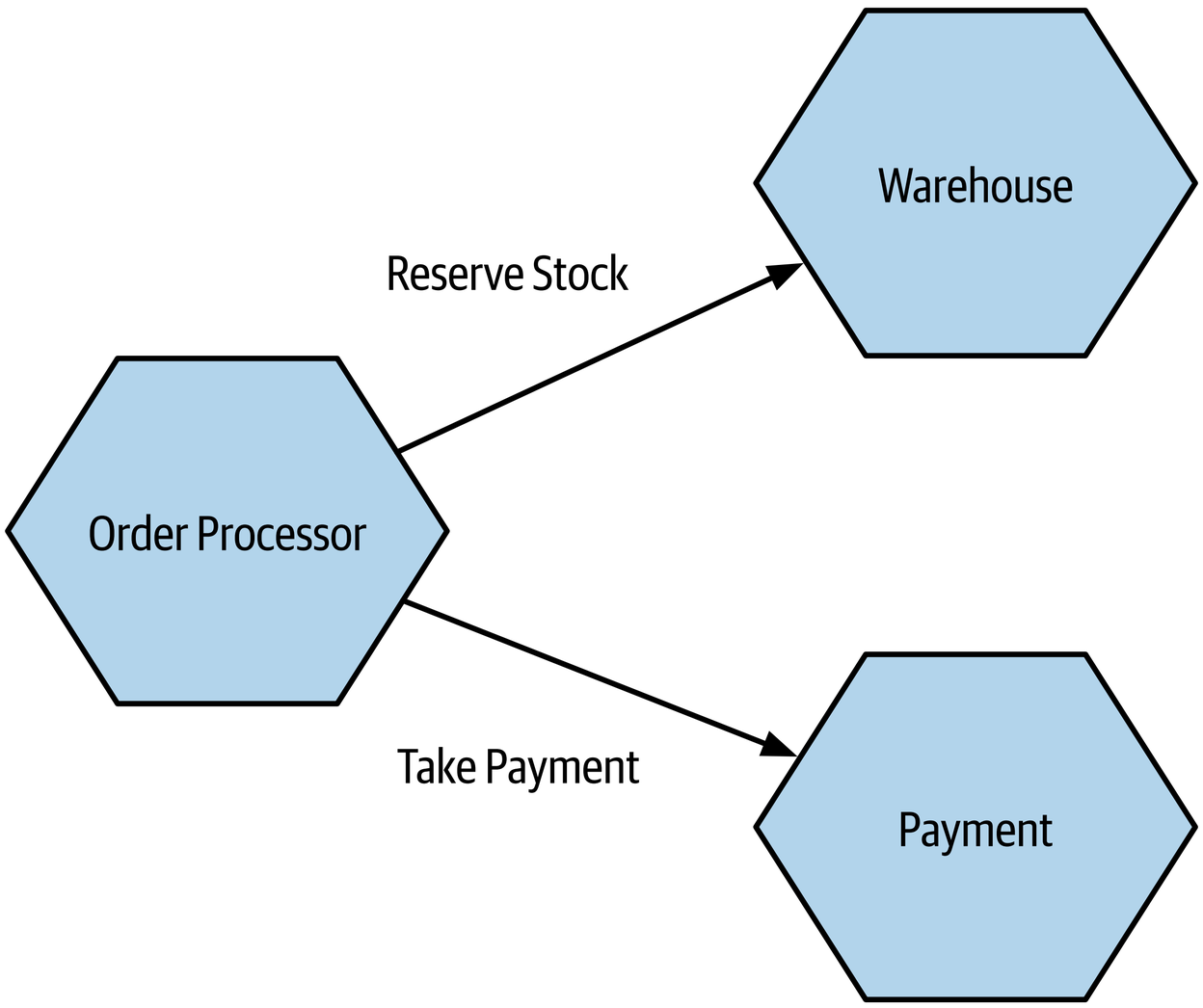
Domain Common Content

TYPES OF COUPLING

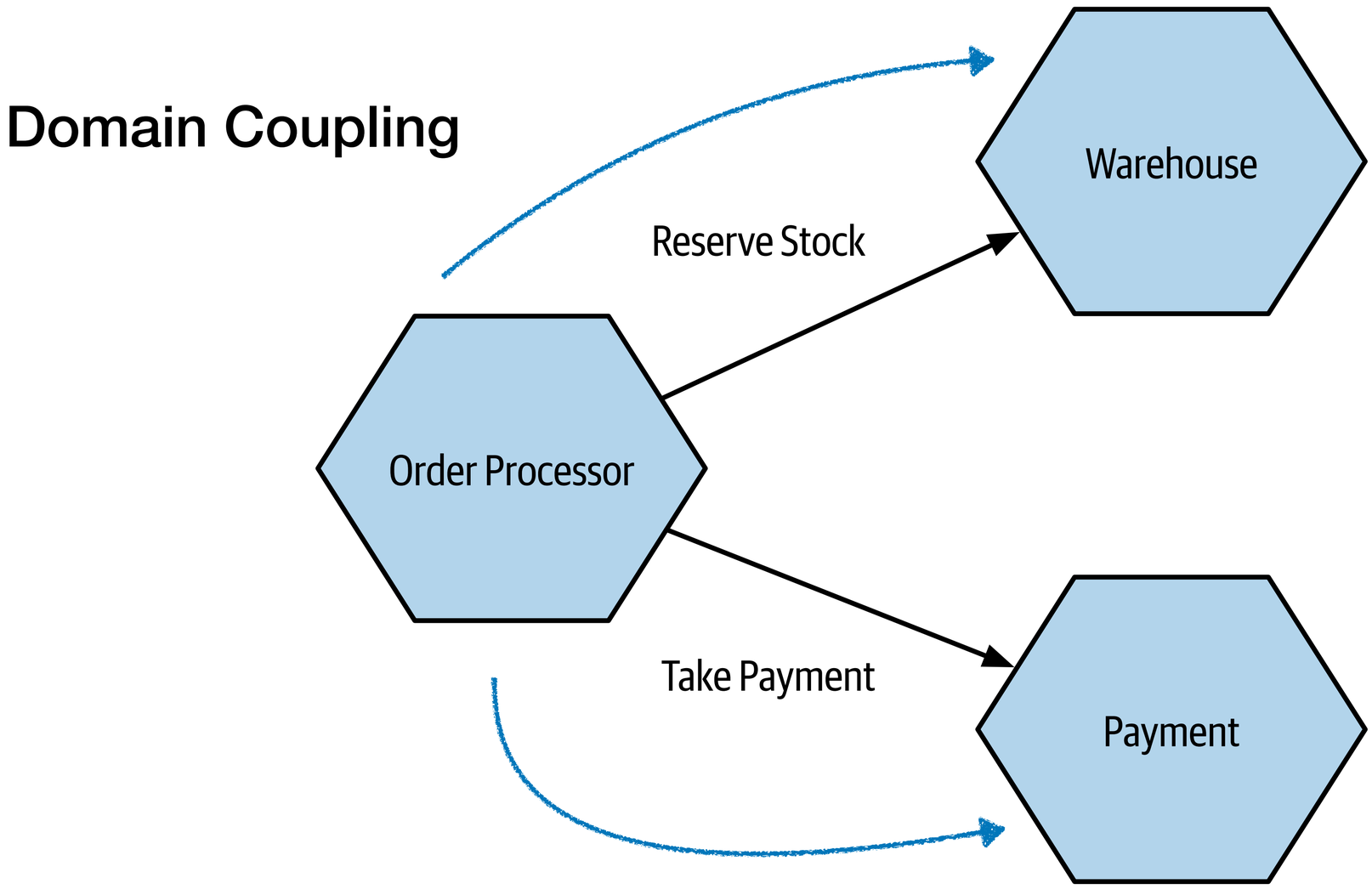
Domain Common Content

Increasing coupling

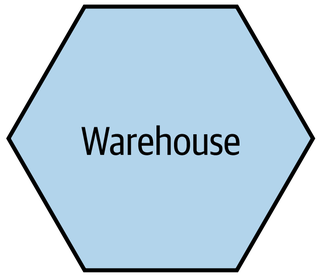
DOMAIN COUPLING



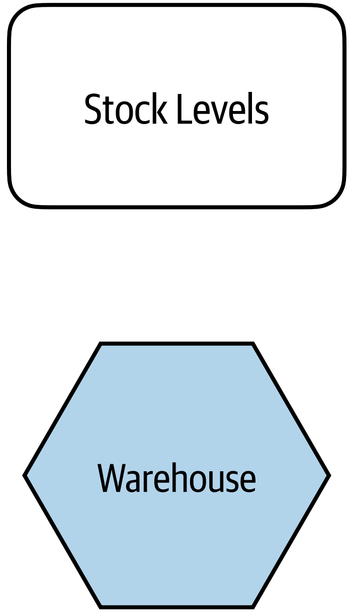
DOMAIN COUPLING



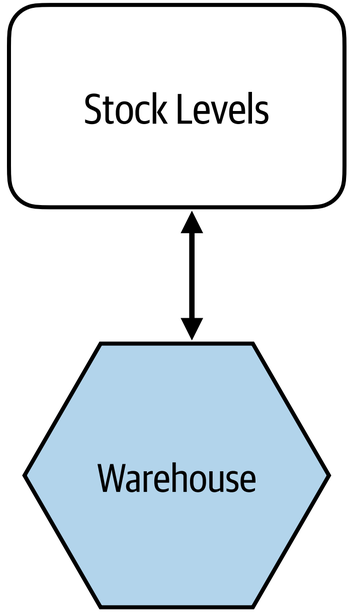
COMMON COUPLING



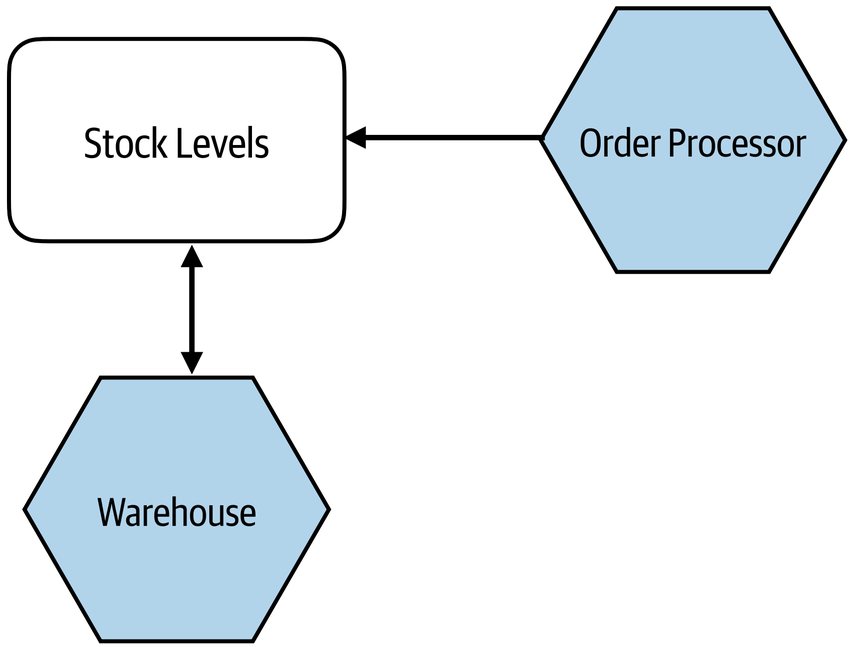
COMMON COUPLING



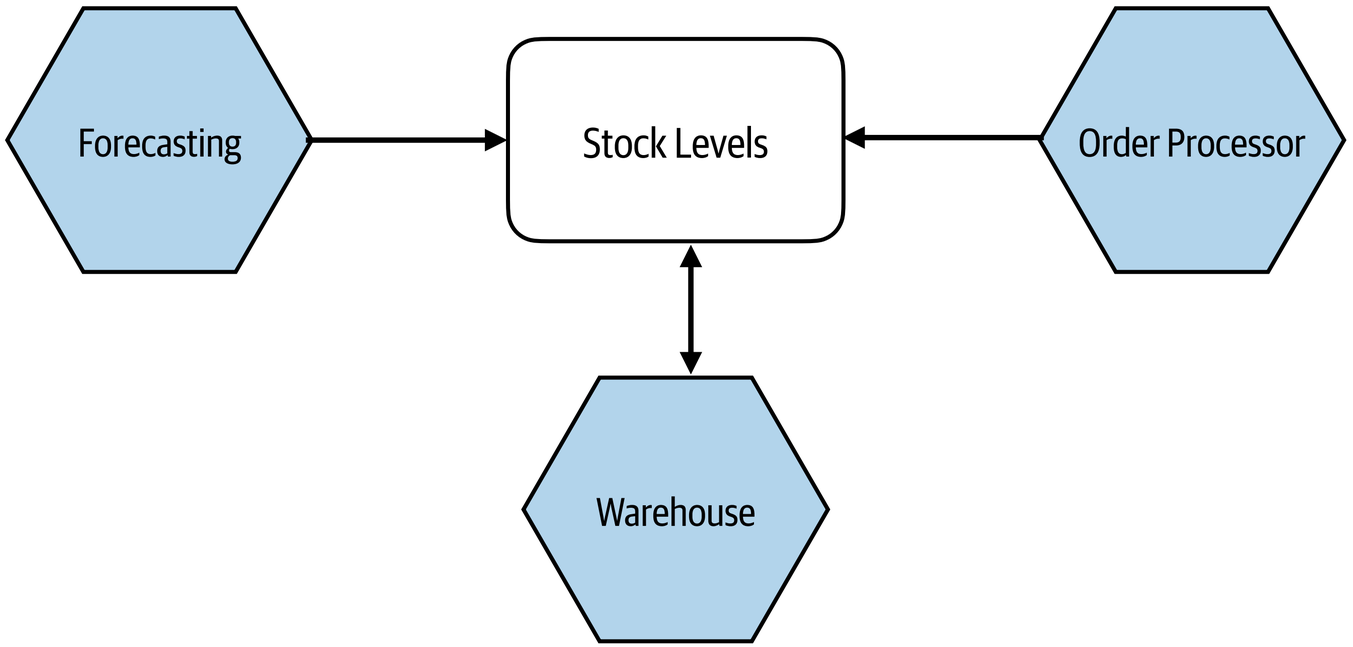
COMMON COUPLING



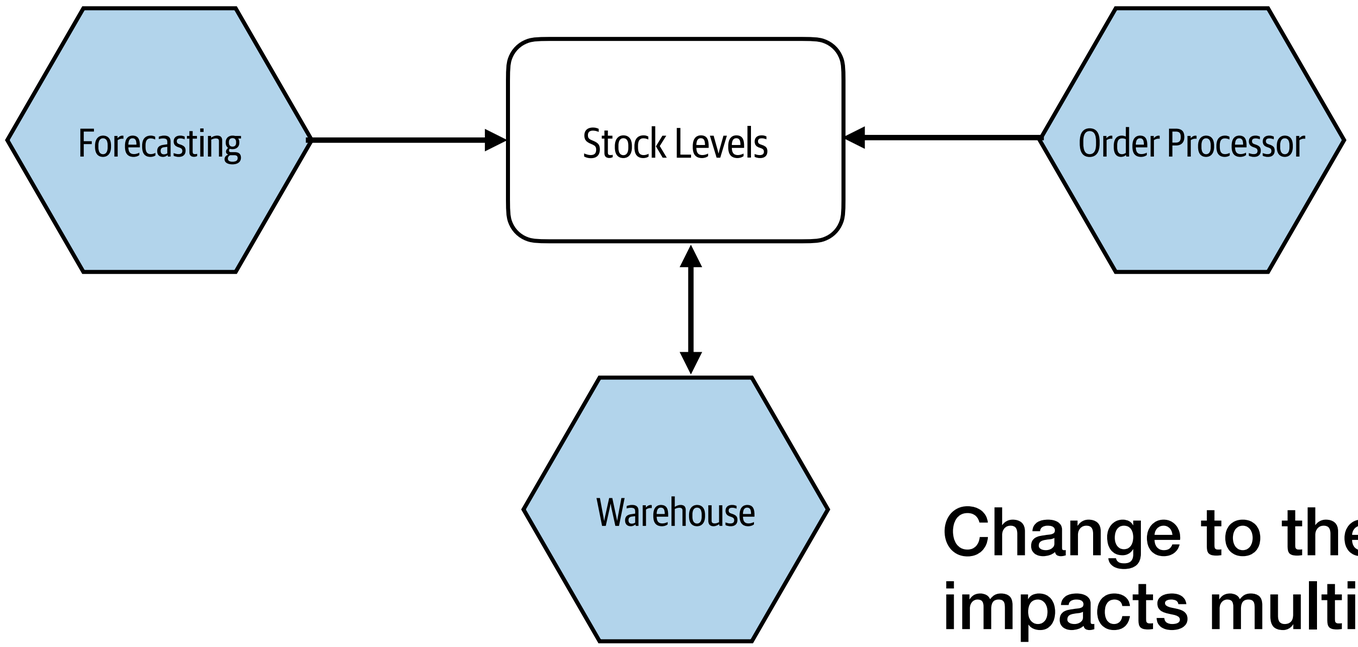
COMMON COUPLING



COMMON COUPLING

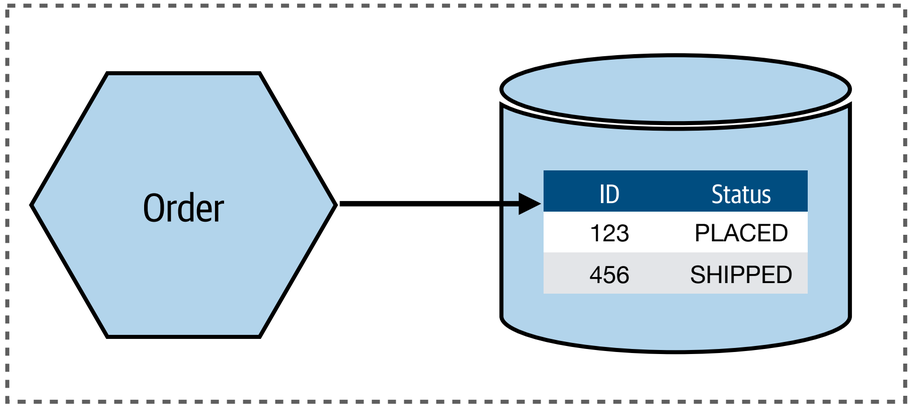


COMMON COUPLING

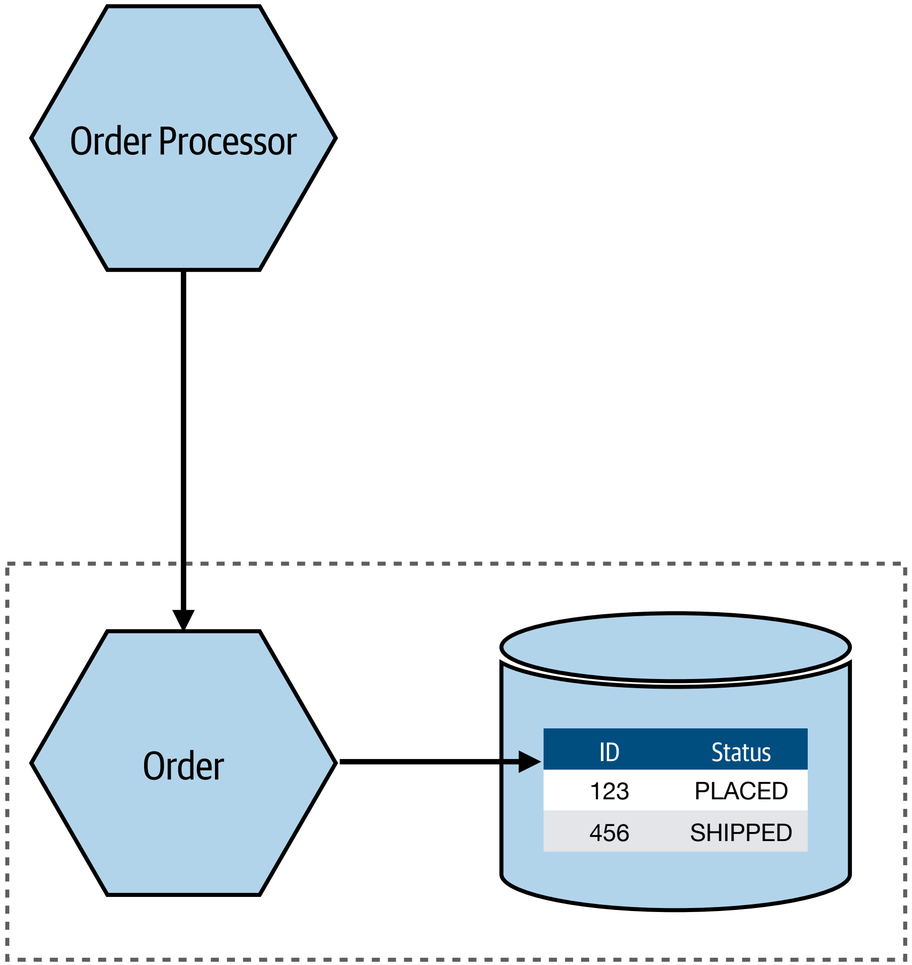


Change to the common data impacts multiple microservices

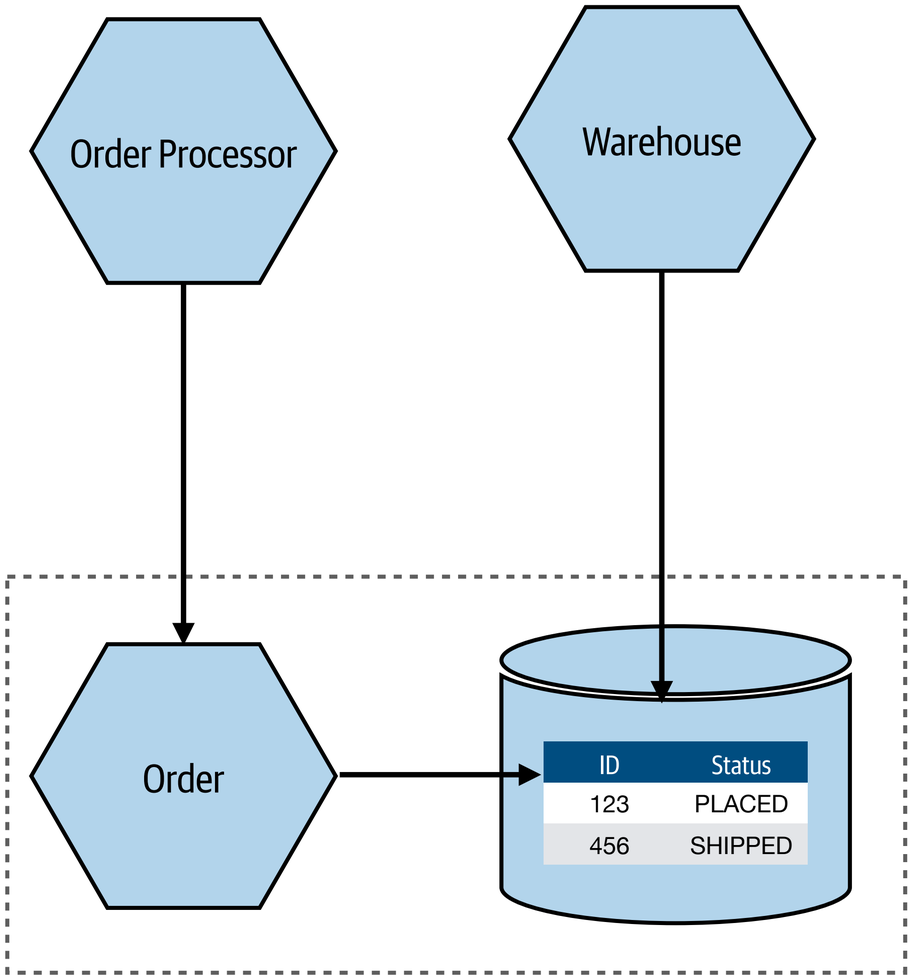
CONTENT COUPLING



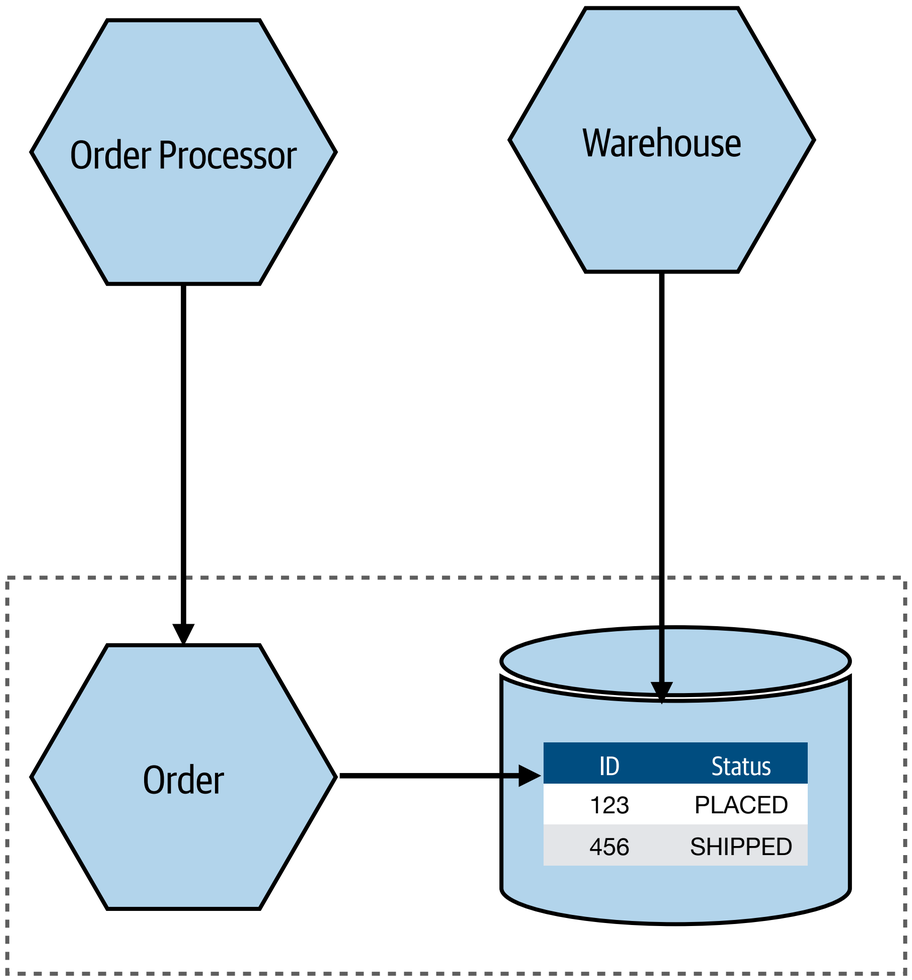
CONTENT COUPLING



CONTENT COUPLING

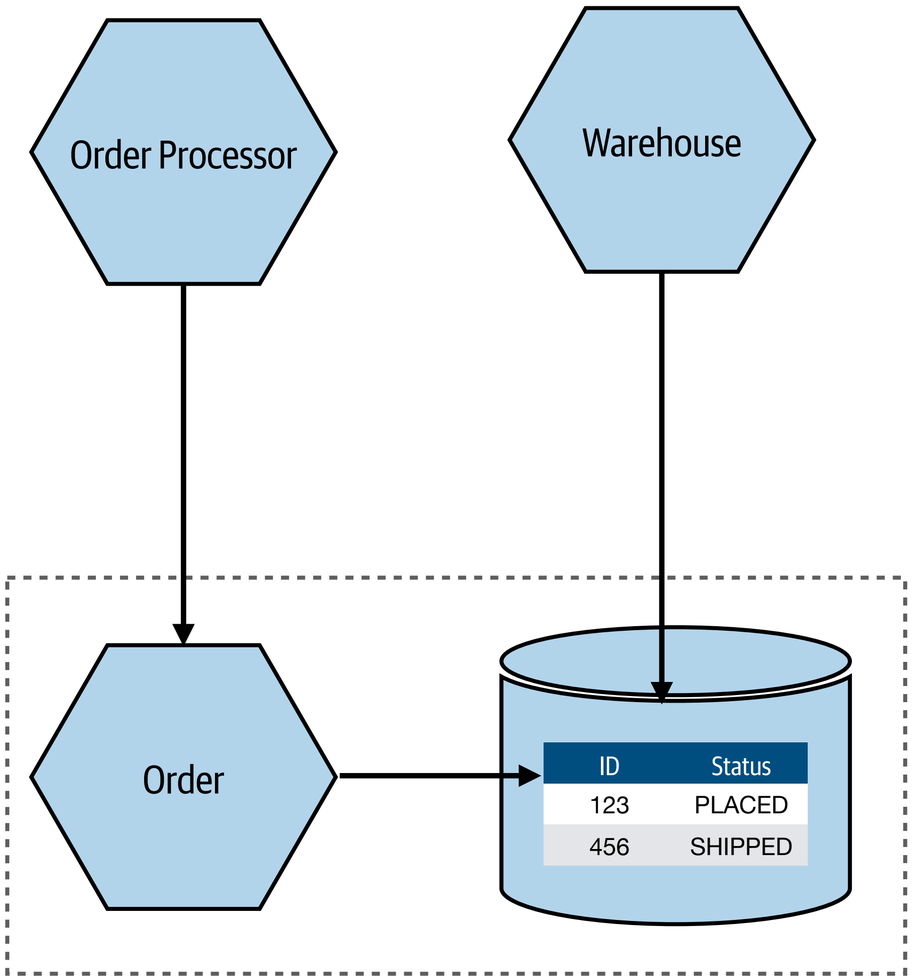


CONTENT COUPLING



Information hiding bypassed

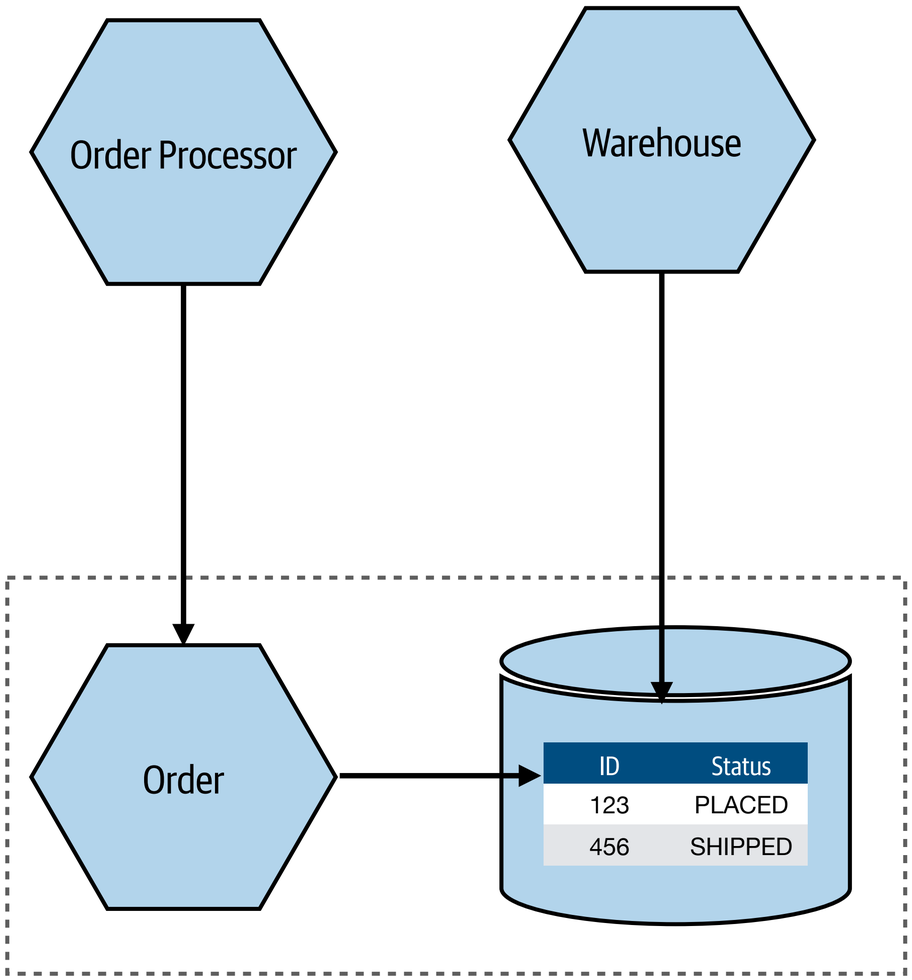
CONTENT COUPLING



Information hiding bypassed

Unclear what can safely be changed in the Order service

CONTENT COUPLING

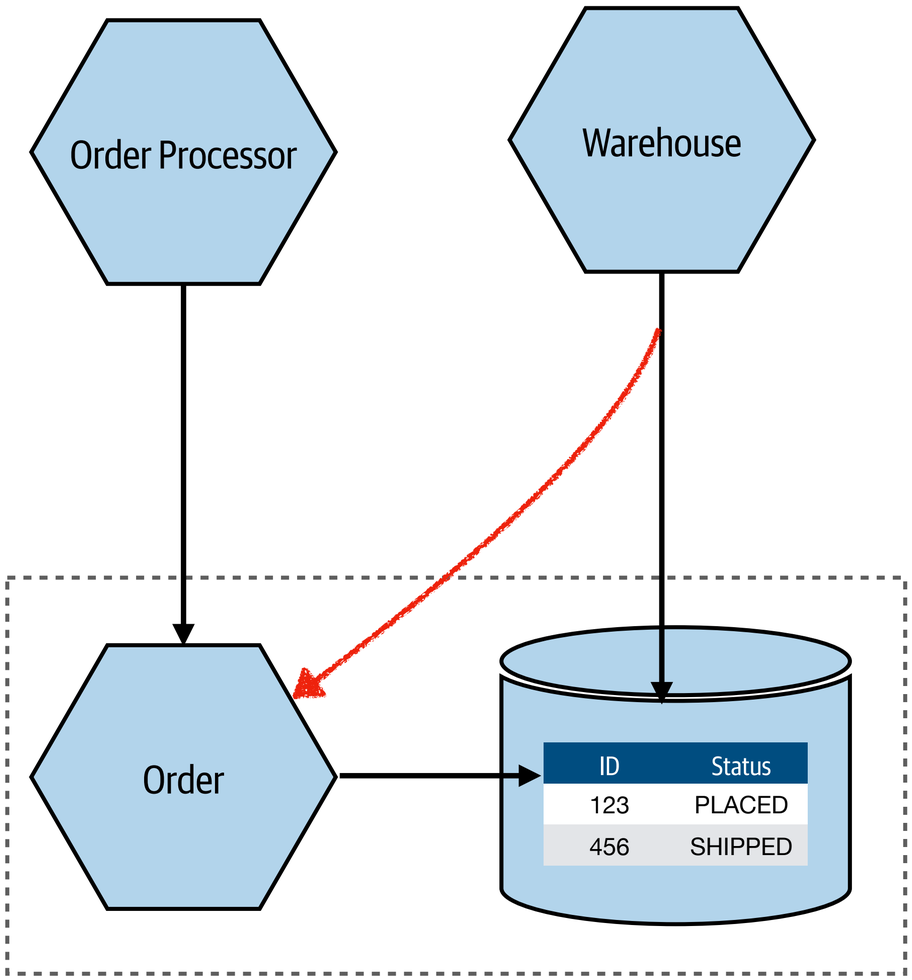


Information hiding bypassed

Unclear what can safely be changed in the Order service

Aka Pathological Coupling

CONTENT COUPLING



Information hiding bypassed

Unclear what can safely be changed in the Order service

Aka Pathological Coupling

TYPES OF COUPLING

TYPES OF COUPLING

TYPES OF COUPLING

Domain

Increasing coupling

TYPES OF COUPLING

Domain

Common

Increasing coupling

TYPES OF COUPLING

Domain Common Content

Increasing coupling

IN SUMMARY

IN SUMMARY

Information hiding is super important

IN SUMMARY

Information hiding is super important

Some coupling is worse than others

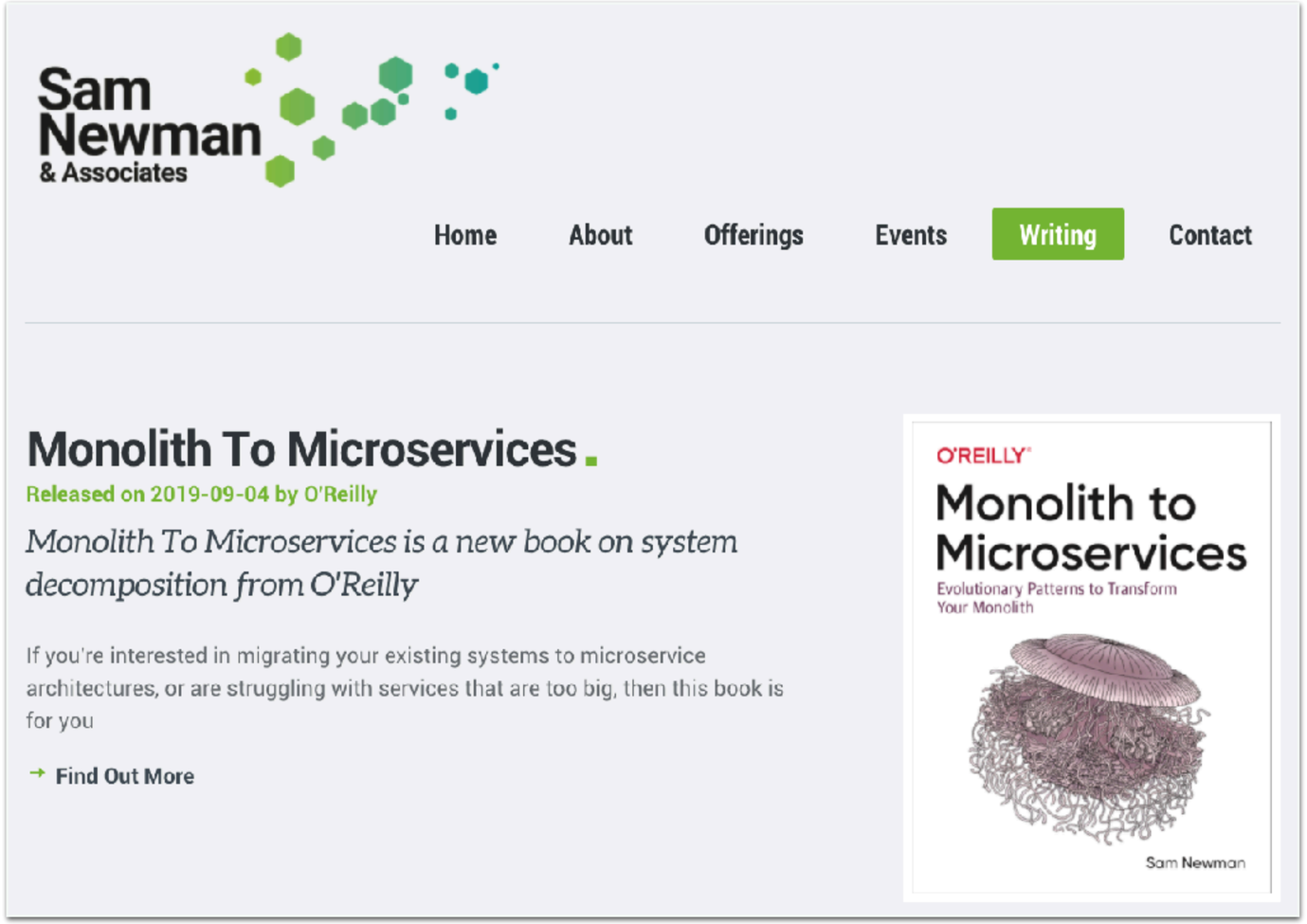
IN SUMMARY

Information hiding is super important

Some coupling is worse than others

Information hiding, low coupling, strong cohesion = Independent Deployability

THANKS!



Sam Newman & Associates

https://samnewman.io/