» Skan.ai - chief Architect

» Ai.robotics - chief Architect

» Genpact - solution Architect

» Welldoc - chief Architect

» Microsoft

» Mercedes

» Siemens

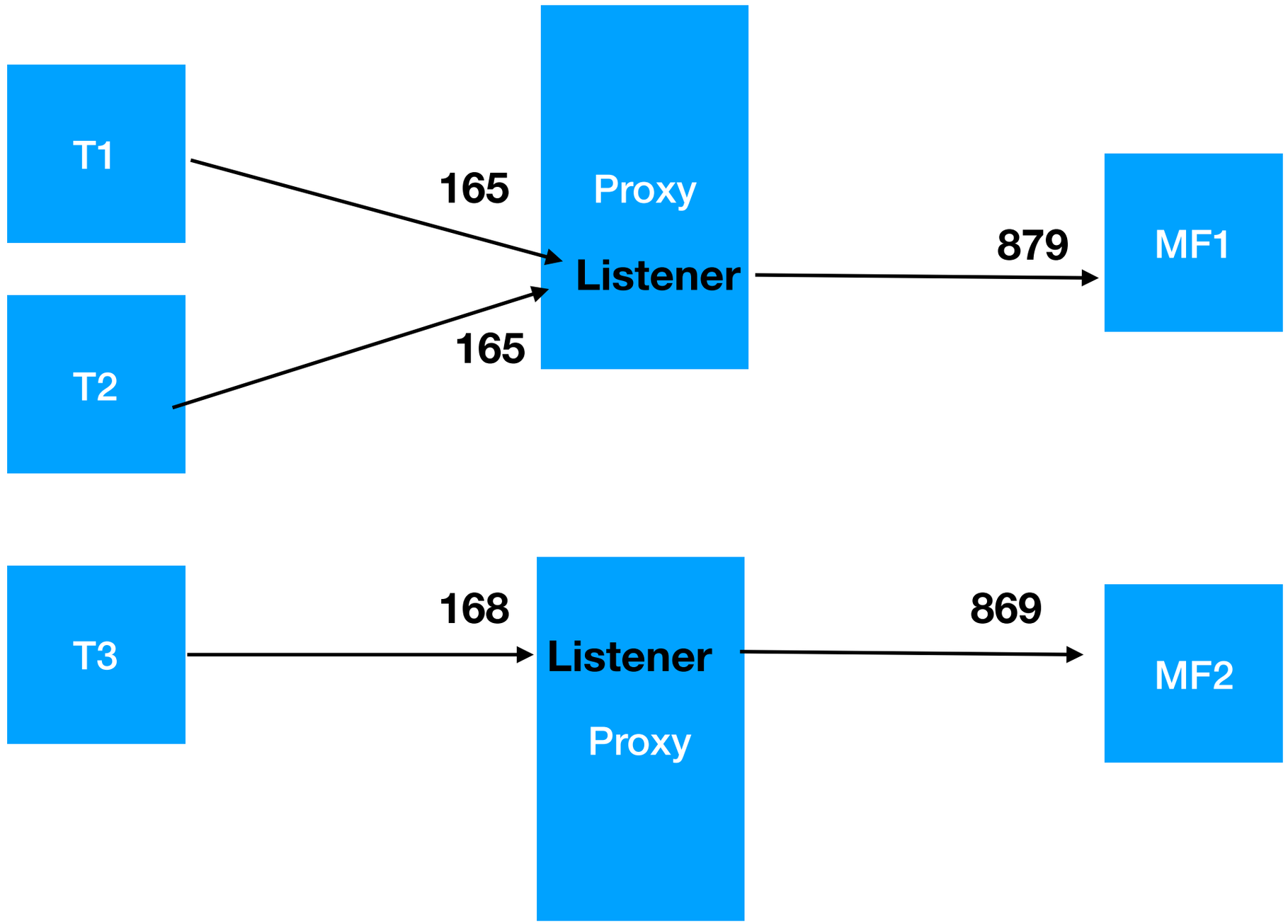
» Honeywell

Mubarak

- **Cloud Computing**
- **Micro services**
- **Data Science**
- **ML**
- **DevSecOps**

- Years of experience
- Role
- Expectations

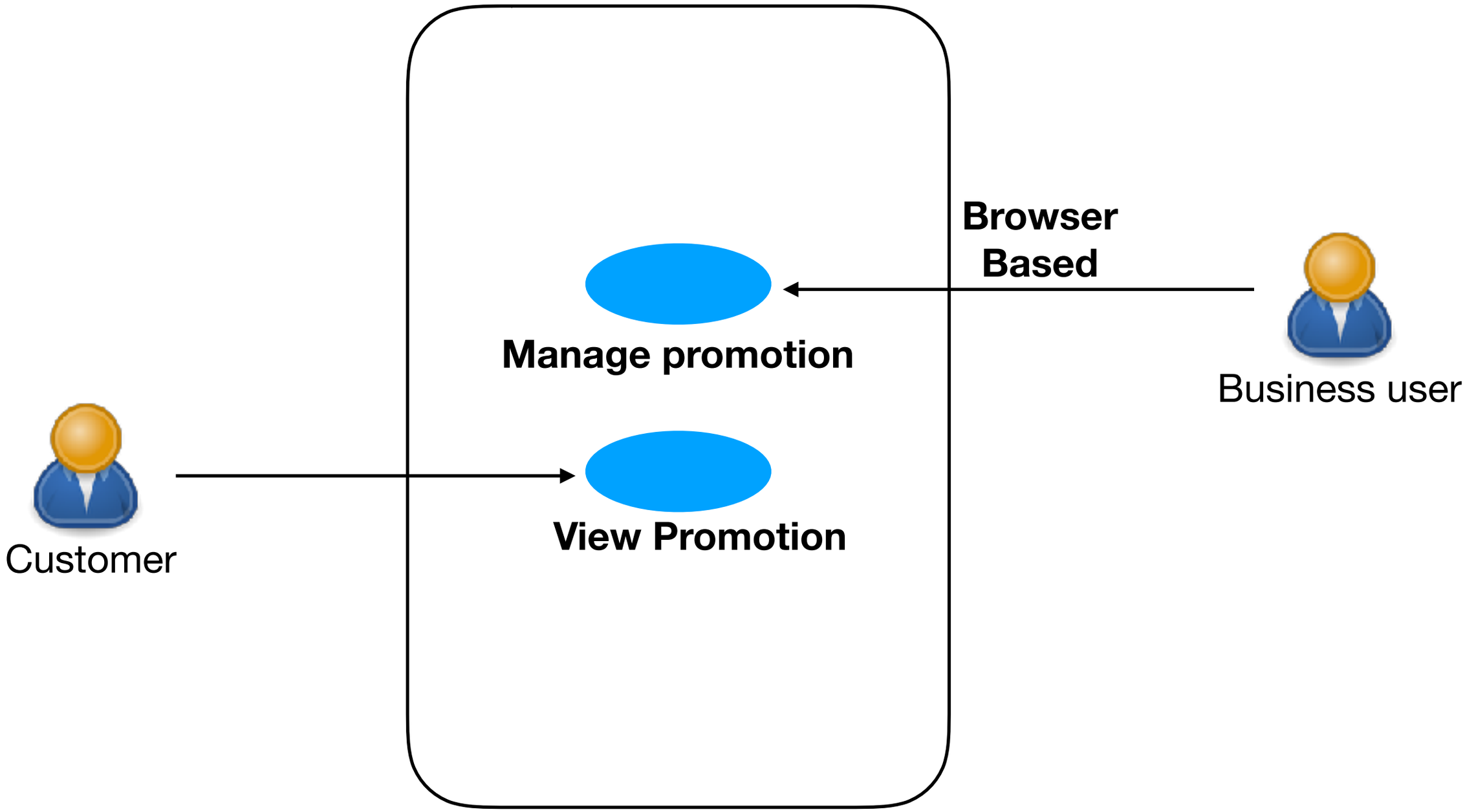
- Sub domain (bounded context)
- Transaction scope
- Aggregate (group of classes very closely related)
- Feature
-


T1
T2
165
165
Proxy
Listener
879
MF1
T3
168
Listener
Proxy
869
MF2

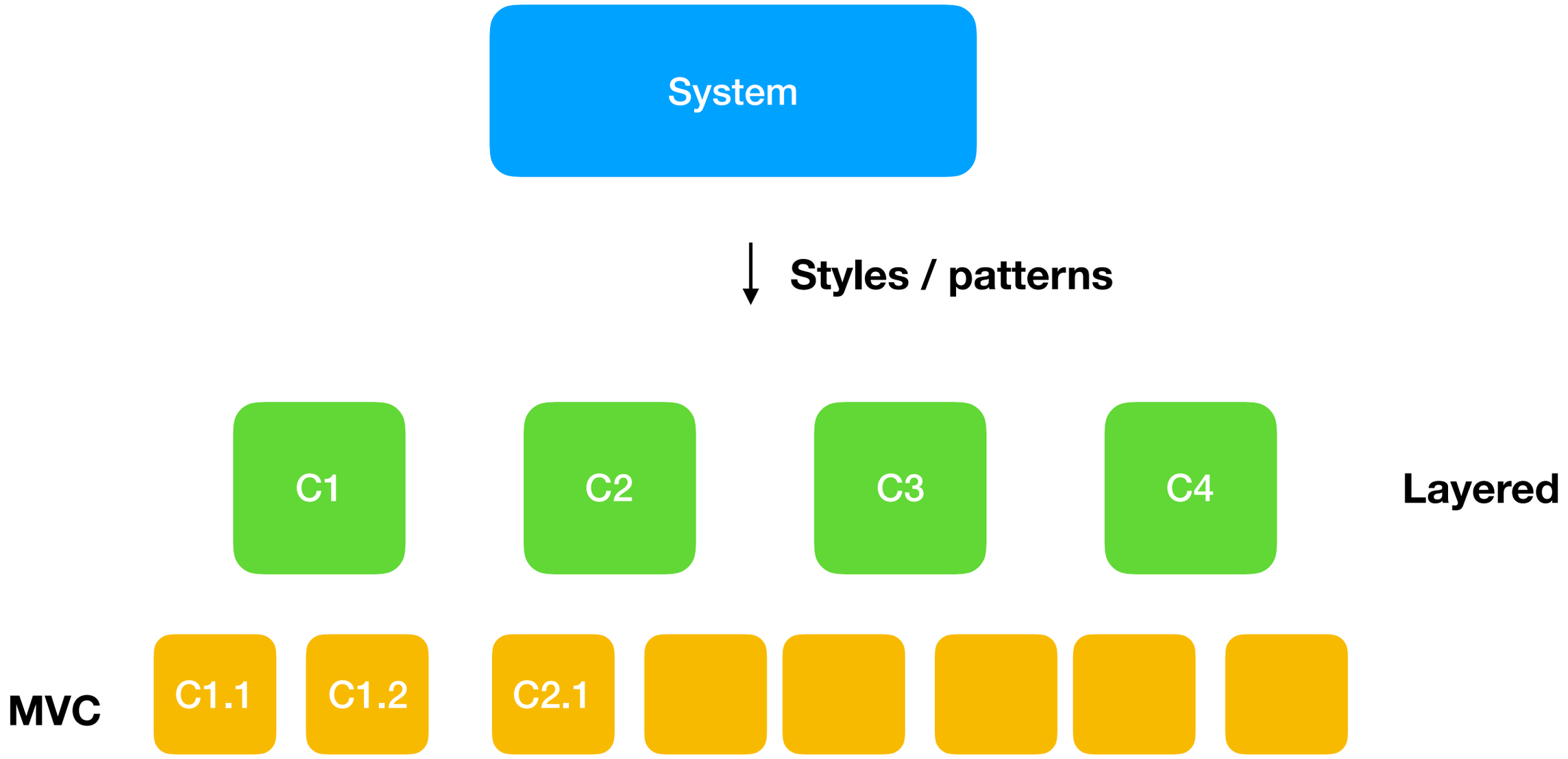
Case Study

Context view

Functional view



Logical view



System Decomposition

Client Application

UI Application
Embedded App (0)
* Ras pi

Native Application (1)
* WPF
* Swing
* Android
# Browser - Native Application (2)
* Flash
* Activex
* Applet
# Browser - Server Pages (3)
* JSP
* Php
# Browser - SPA(4)
* Angular
* React

Background Application
* Windows Service
* Daemon

Low Level Application
* Device Driver

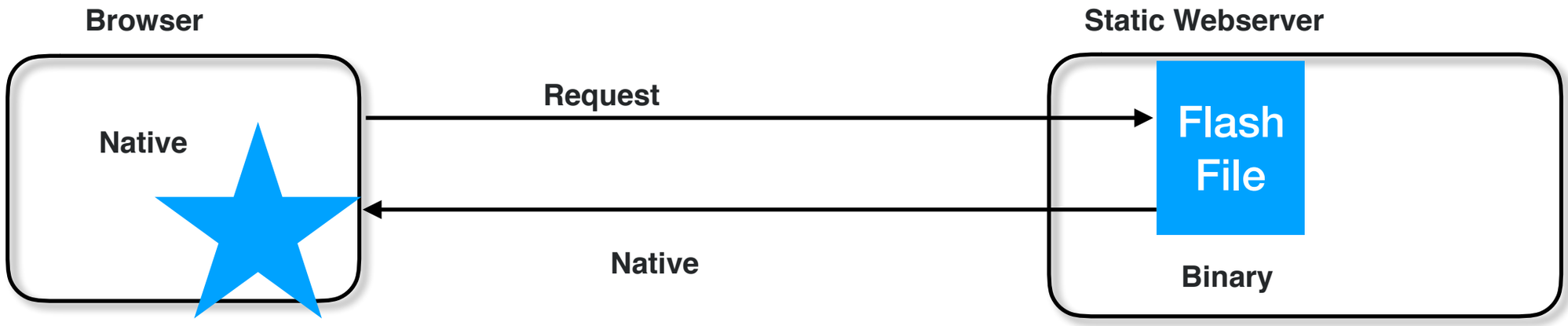
Server Application

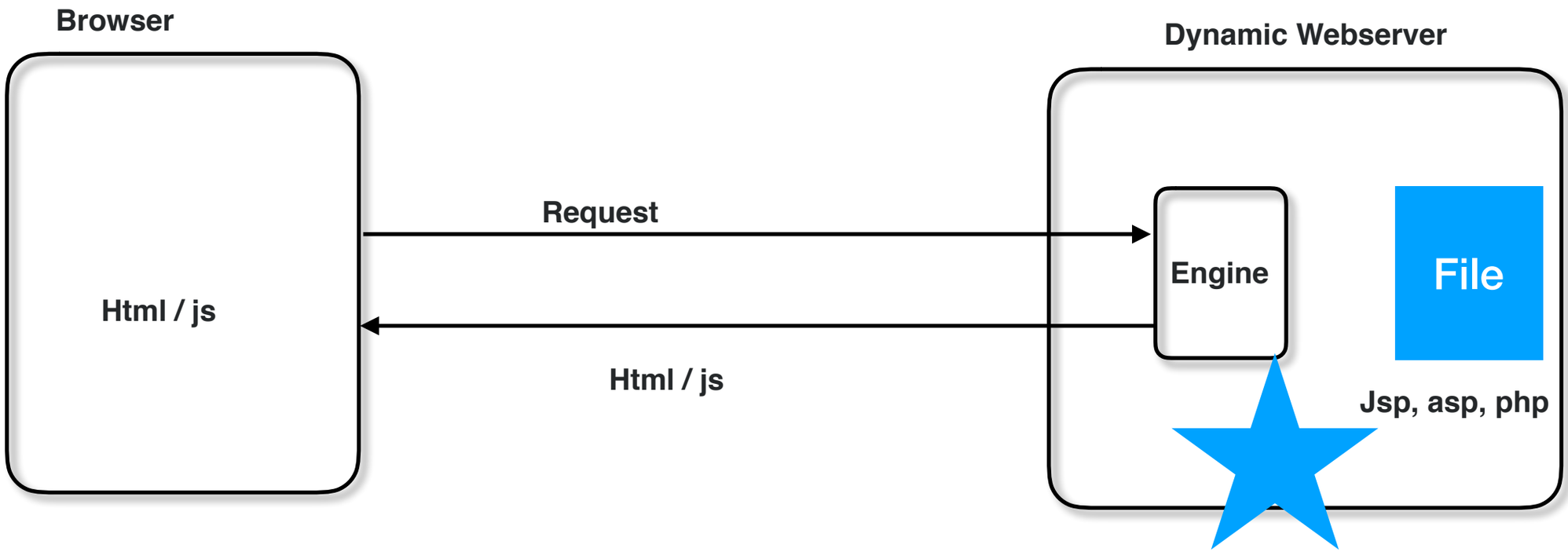
Foreground Application
(Request/response)
* UI App
* API Application

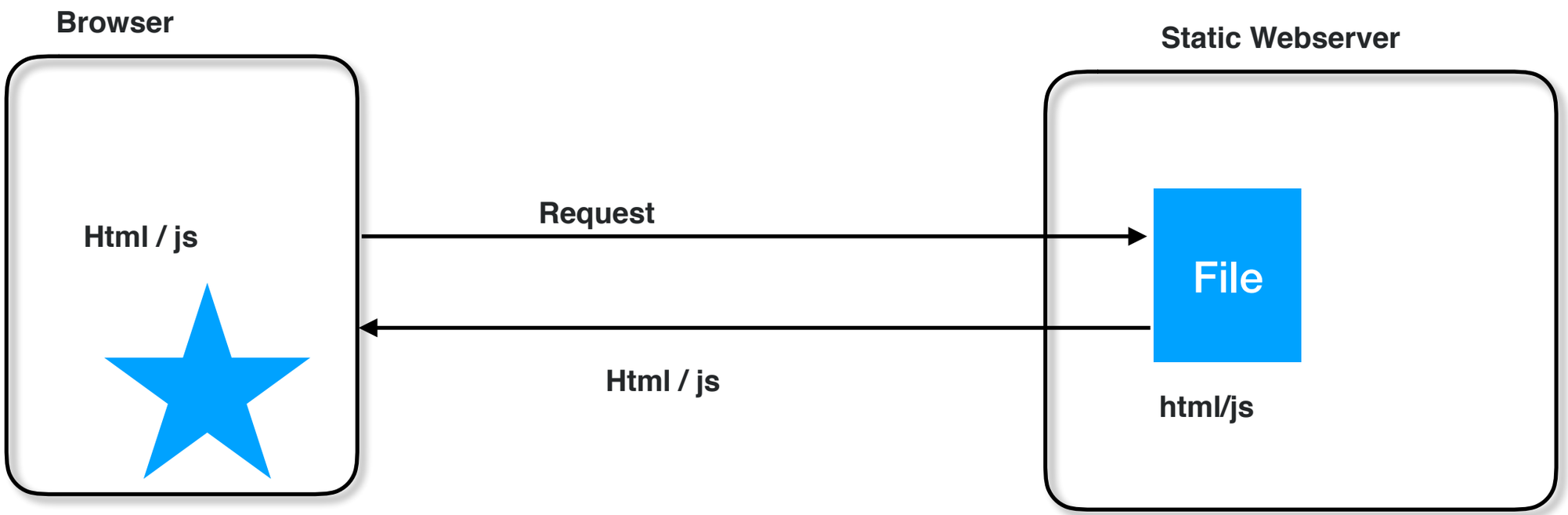
Background Application
Event Based Application
Scheduled Application

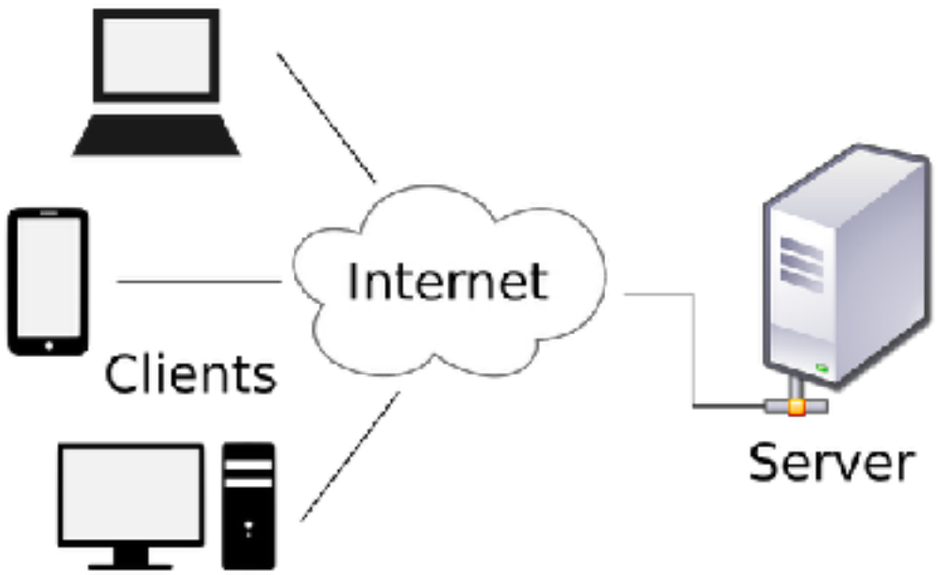
Client-Server
Applications

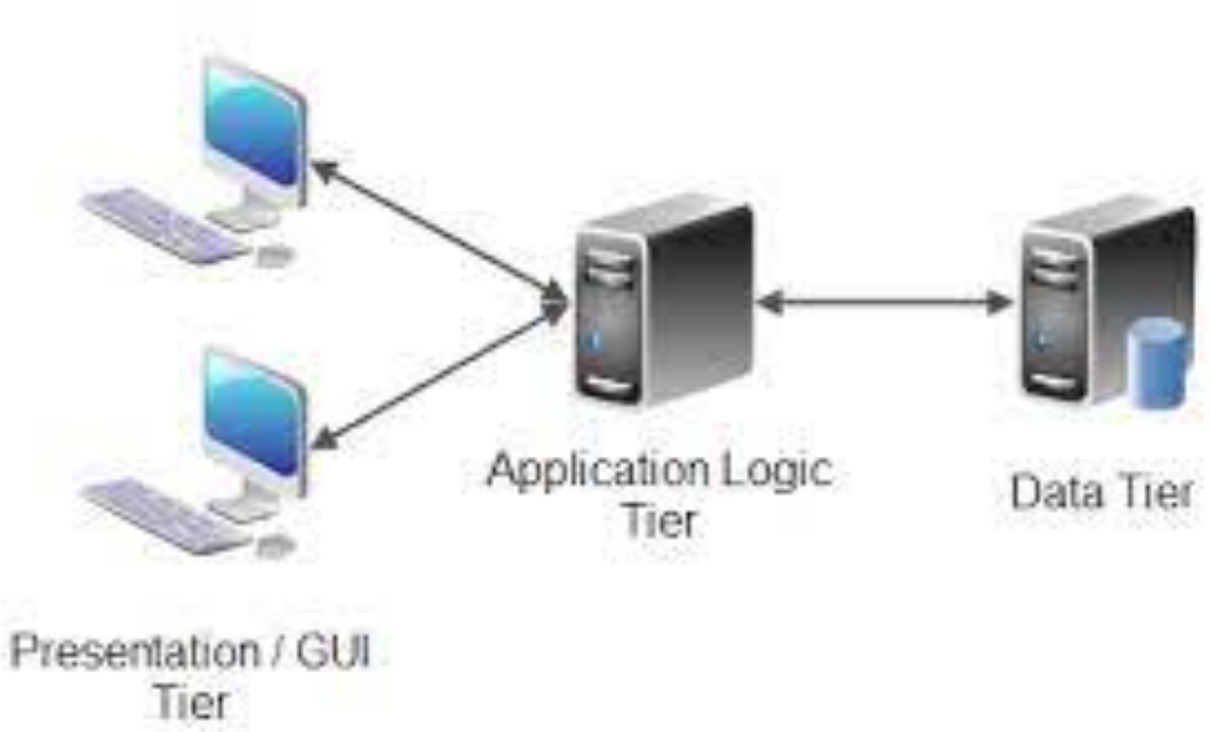
Two-tier Client Server
Multi(N)-tier client-server


Browser
Native
Request
Native
Static Webserver
Flash
File
Binary

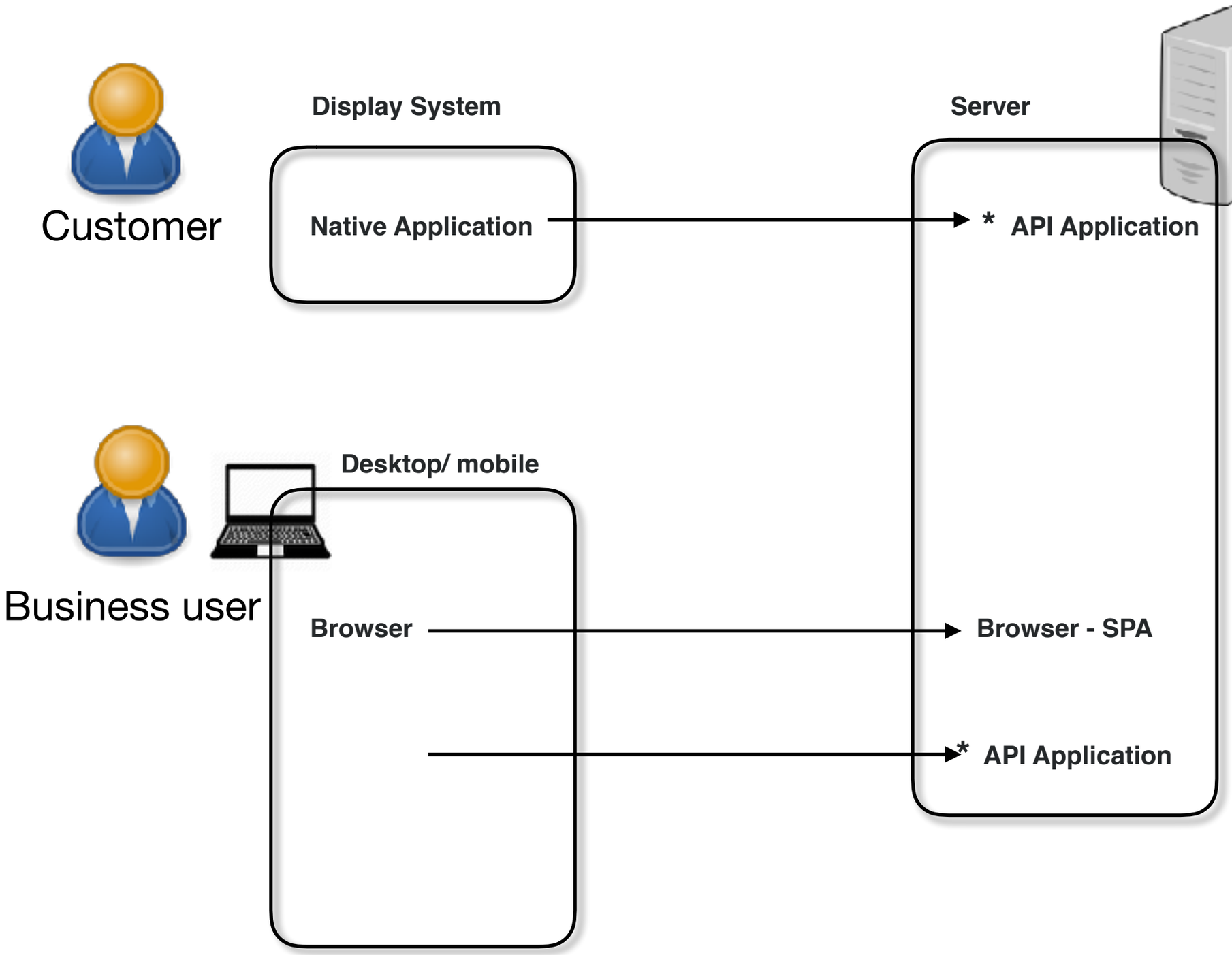

Browser
Html / js
Request
Html / js
Dynamic Webserver
Engine
File
Jsp, asp, php

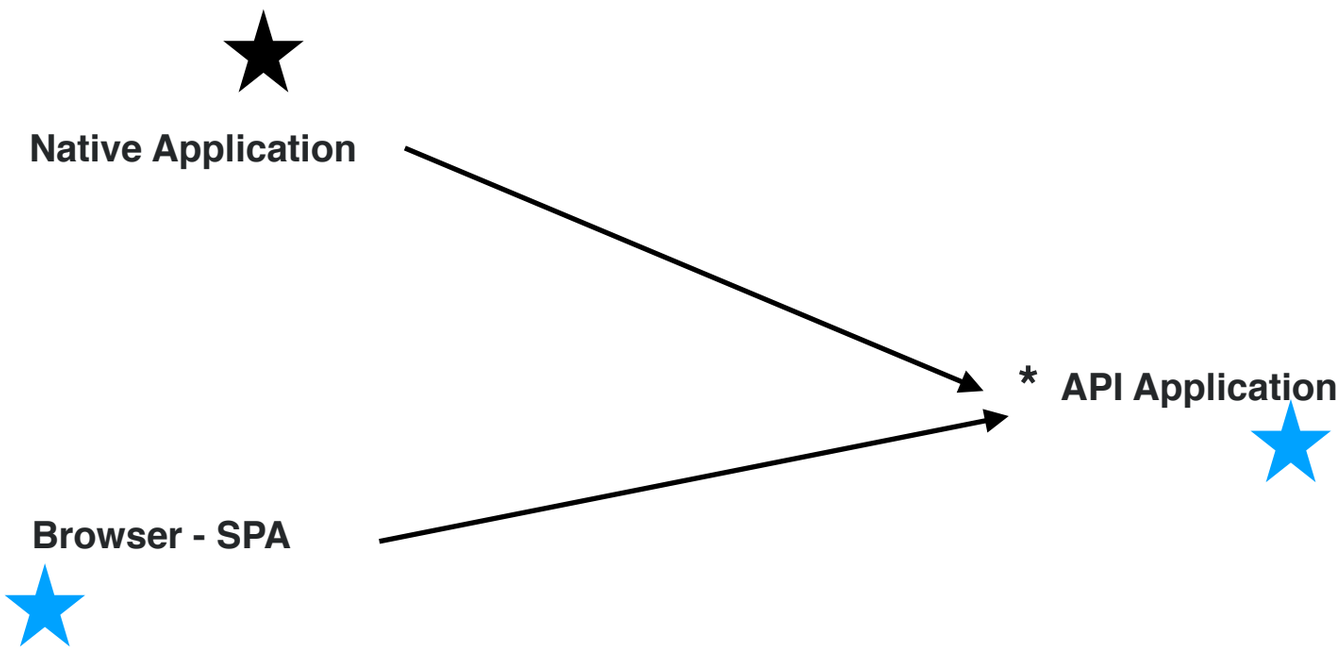

Browser
Html / js
Request
Html / js
Static Webserver
File
html/js


Internet
Clients
Server


Application Logic Tier
Data Tier
Presentation / GUI Tier

Logical view - todo




Native Application
* API Application
Browser - SPA

Application Decomposition

Transformations at Similar levels of abstraction

1

Pipes and filter

Transformations at Different levels of abstraction

2

Layered
Hexagonal

Decomposition based on core and Variance

3

Micro Kernel

Decompose based on business capabilities (features)

4

Component based
Distributed component
Microservice
SOA

ToDo App is simple and awesome app to organize your tasks with very easy to use interface. ToDo can help you to make list of your tasks and also you can set Reminder with specific tasks. It reminds you at you specified Time.

Order Processing. Receive an order, Decode an encrypted order, Authenticate the order, then Catch duplicated orders (for example if the order was sent twice) and send to order management system.

Application to upload and manage Photos.

The customer screen is responsible for accepting the request and displaying the customer information.

A task scheduler. A scheduler contains all the logic for scheduling and triggering tasks

A workflow implementation. The implementation of a workflow contains concepts like the order of the different steps, evaluating the results of steps, deciding what the next step is, etc.

Payment Service receives a data file which consists of,
- Payment information related to the invoices already delivered
- Notes regarding already delivered invoices
- Invoice information,
- And Credit Notes (Invoice Cancellations)

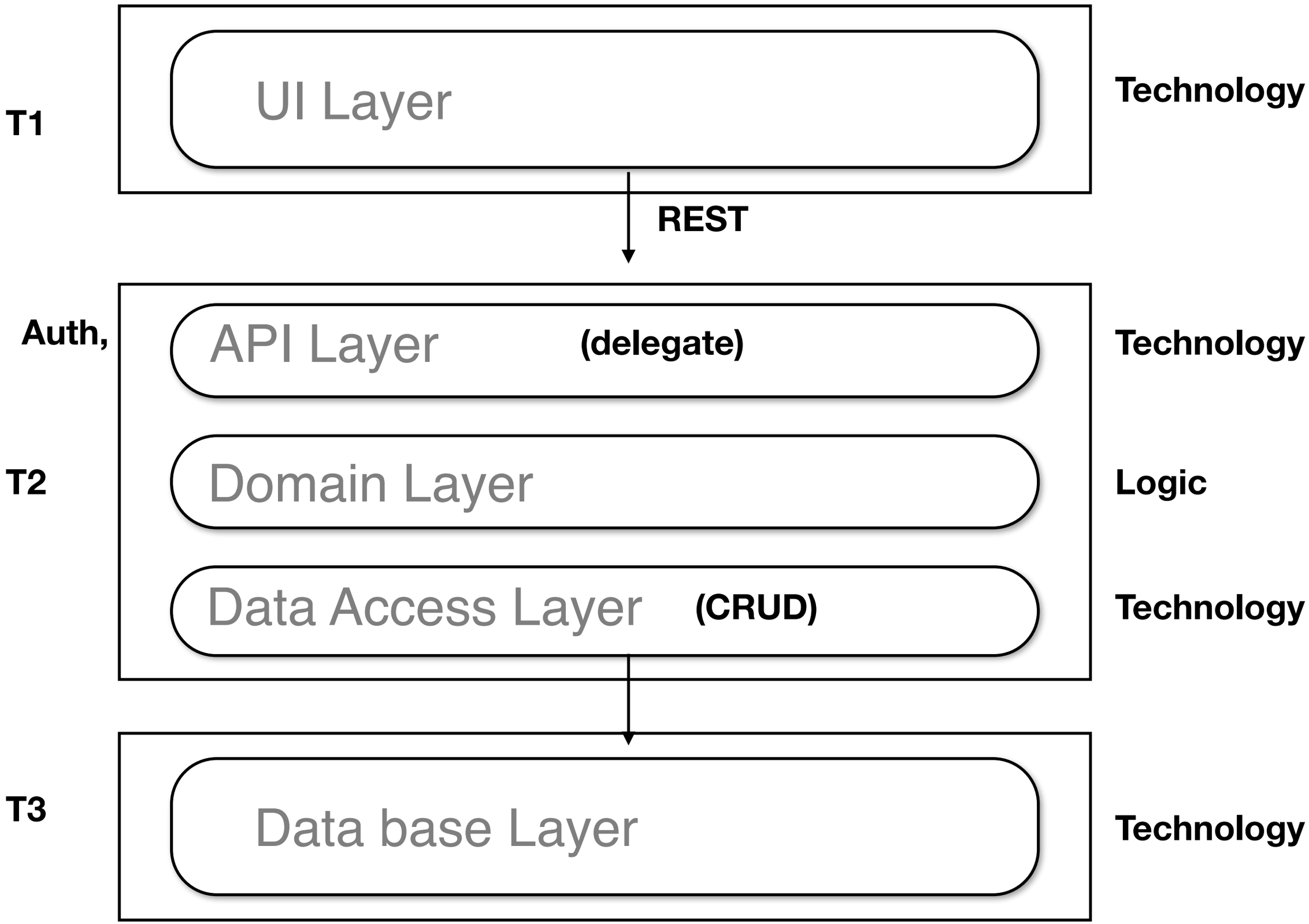
we need to import into the system.
- Invoices
- Payments
- Notes
- Credit Notes

Build two health care applications
1. Diabetes Management
2. Asthma Management.

Diabetes and asthma share a lot in common

Design #1

Logical View



Design #1

Logical View

T1

UI Layer

Technology

(delegate) **REST**

API Layer

Domain Layer

DAL **(CRUD)**

API Layer

Domain Layer

DAL **(CRUD)**

API Layer

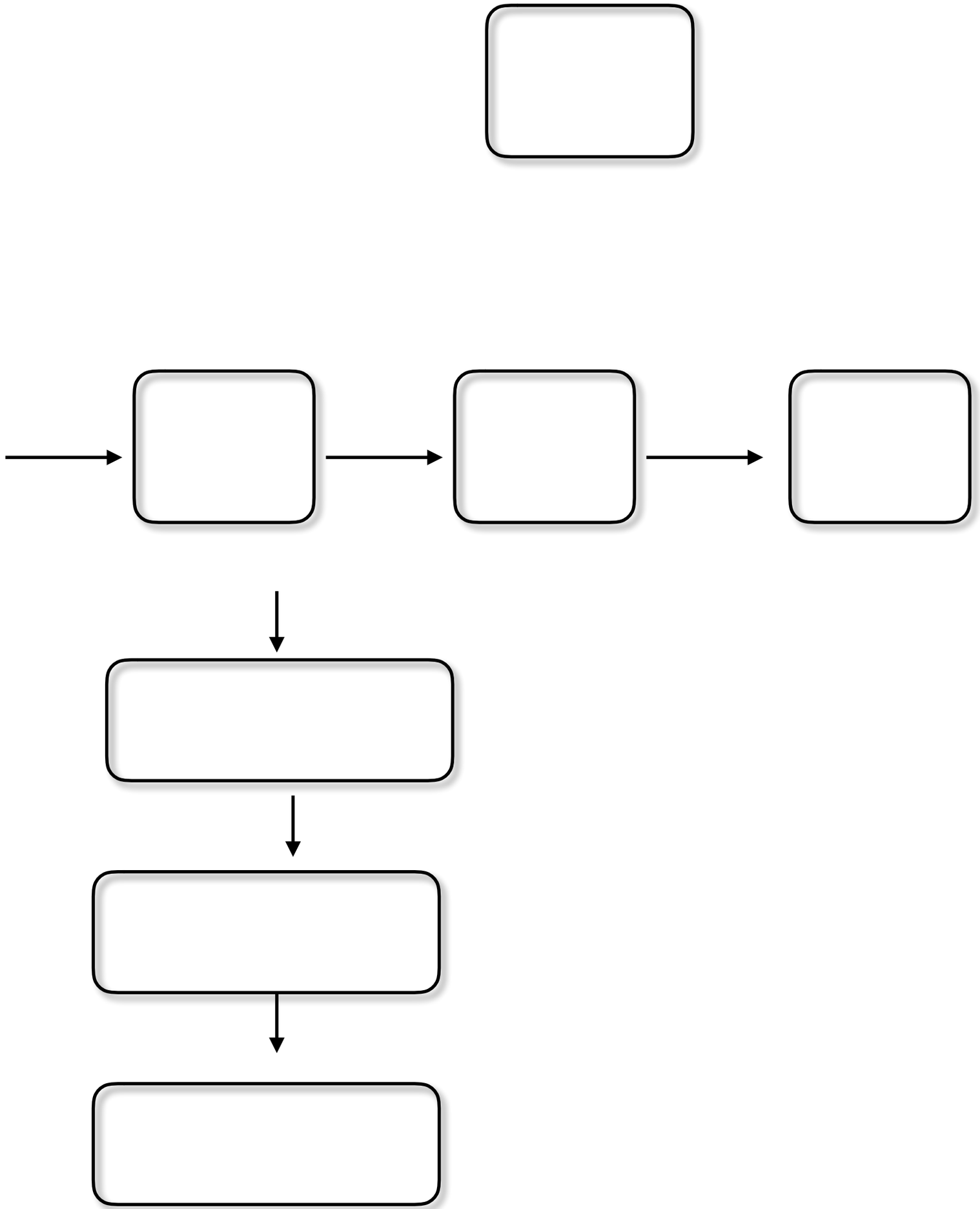
Domain Layer

DAL **(CRUD)**

T3

Data base Layer

Technology

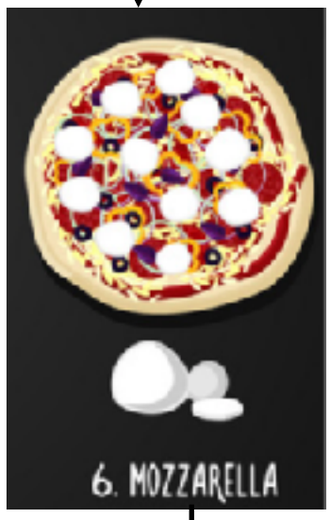



1. DOUGH


2. SAUCE


3. CHEESE


4. TOPPINGS


6. MOZZARELLA


1


5

Invoke logic()

Layer 1

f1(int) f2() f3(int,int)

Layer 2

f3(int,int)

f1(bool) f4()

f2(double)

Layer 3

f1() f4()

f2(int) f3() f5(int)

Layer 4

f1(int) f2()

Process Data()

Pipe

Filter 1

f1(int)

Filter 2

f2(int)

Filter 3

f3(int)

* Can swap filters

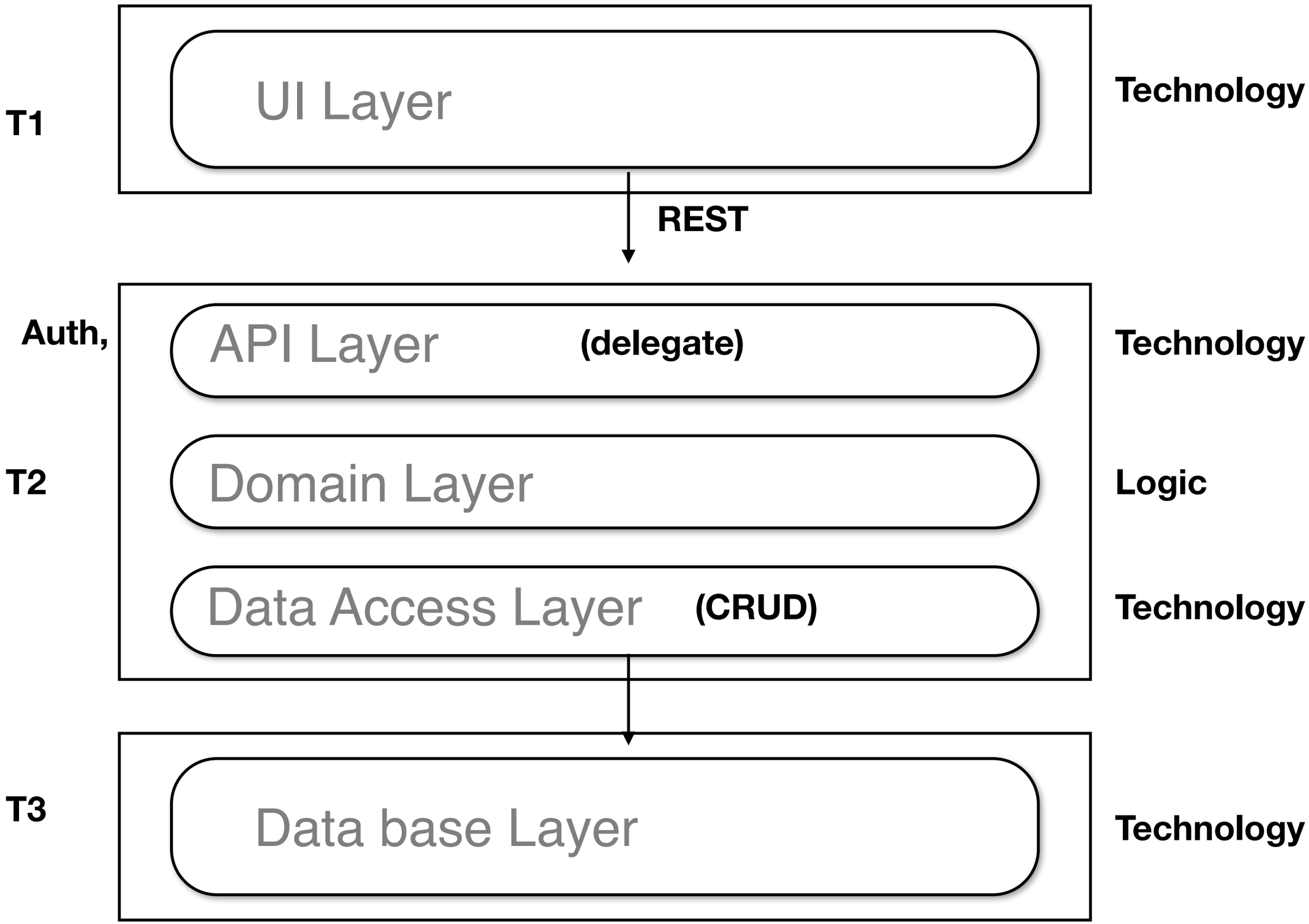
Filter 4

f4(int)

Data

Design #1

Logical View



Design #2

T1

validation, formatting, Routing, State mgmt

View Layer — **Technology**

Controller Layer — **Logic**

Model Layer — **Technology**

REST

API Layer — **Technology**

T2

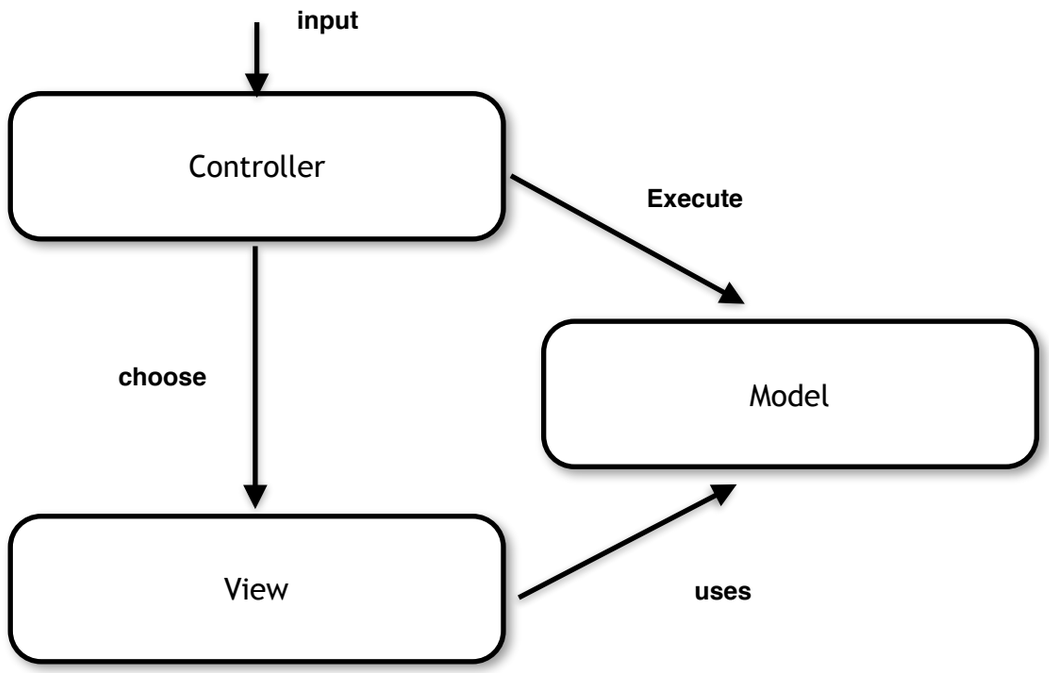
Domain Layer — **Logic**

Data Access Layer — **Technology**

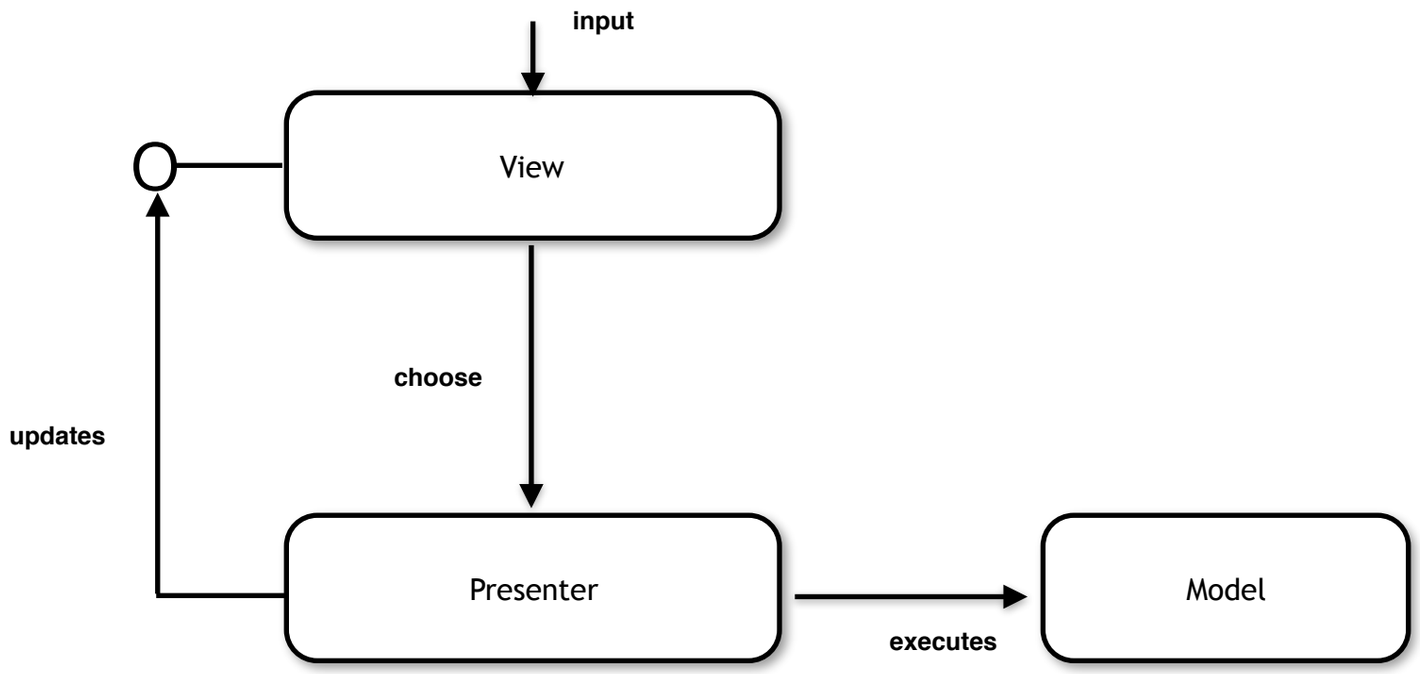
T3

Data base Layer — **Technology**

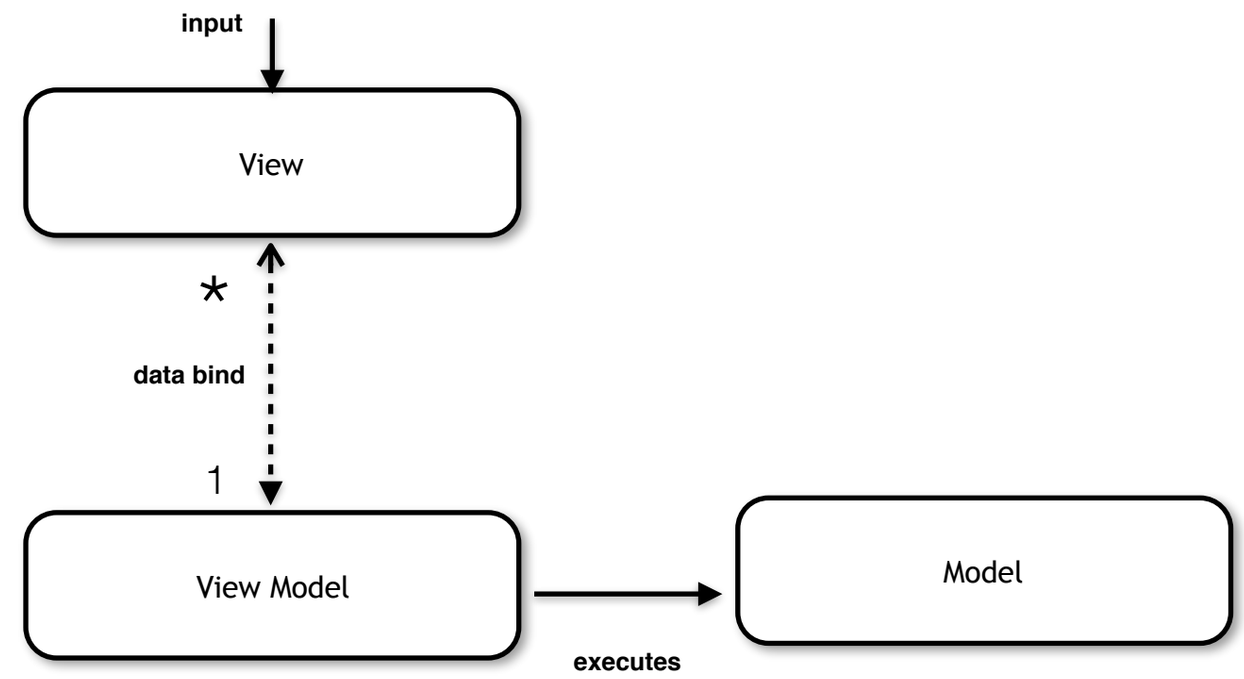
MVC 2



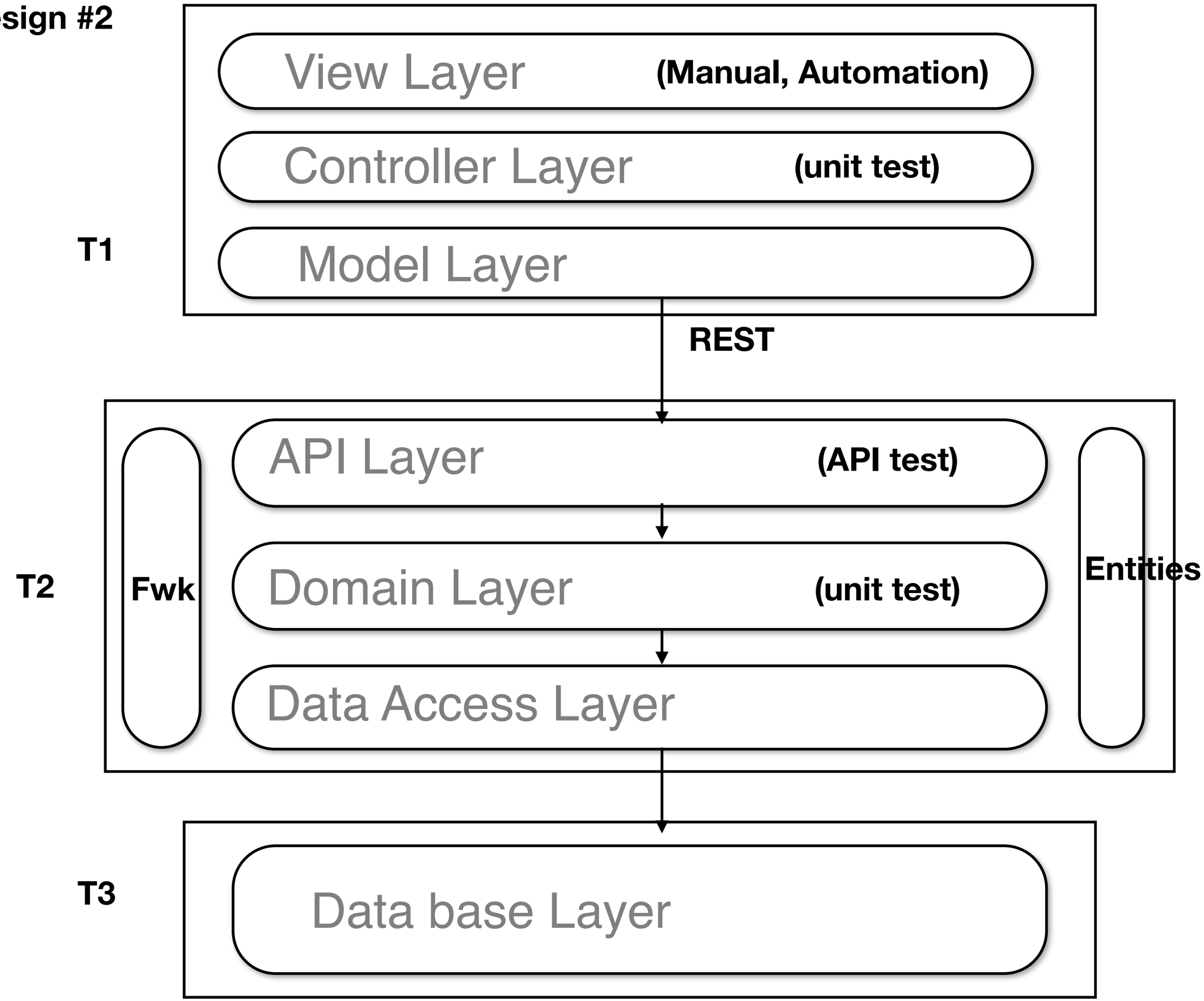
mvp



MVVM



Design #2


View Layer
(Manual, Automation)
Controller Layer
(unit test)
Model Layer
T1
REST
API Layer
(API test)
Domain Layer
(unit test)
Data Access Layer
Fwk
Entities
T2
Data base Layer
T3

Design #3

T1

View Layer

Controller Layer

Model Layer

Technology

Domain Logic

Technology

REST

T2

API Layer (delegate)

1 to 1

Application Layer/ Facade Layer

(CRUD)
Data Access Layer

(logic)
Domain Layer

API Technology

Cross Cutting Concern Technology

Domain Logic

Technology

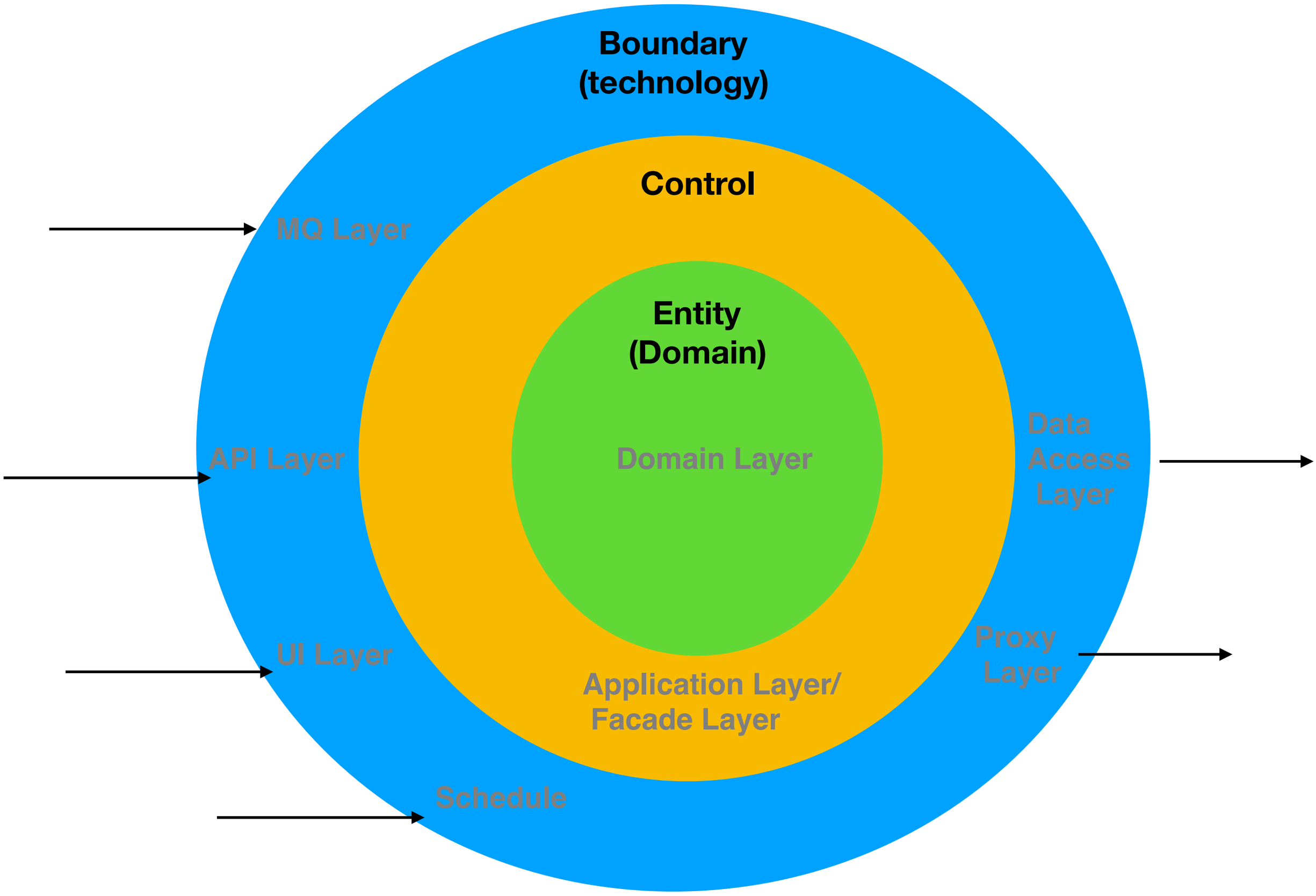
flow, log, Exception Handling, Authorization, Caching

T3

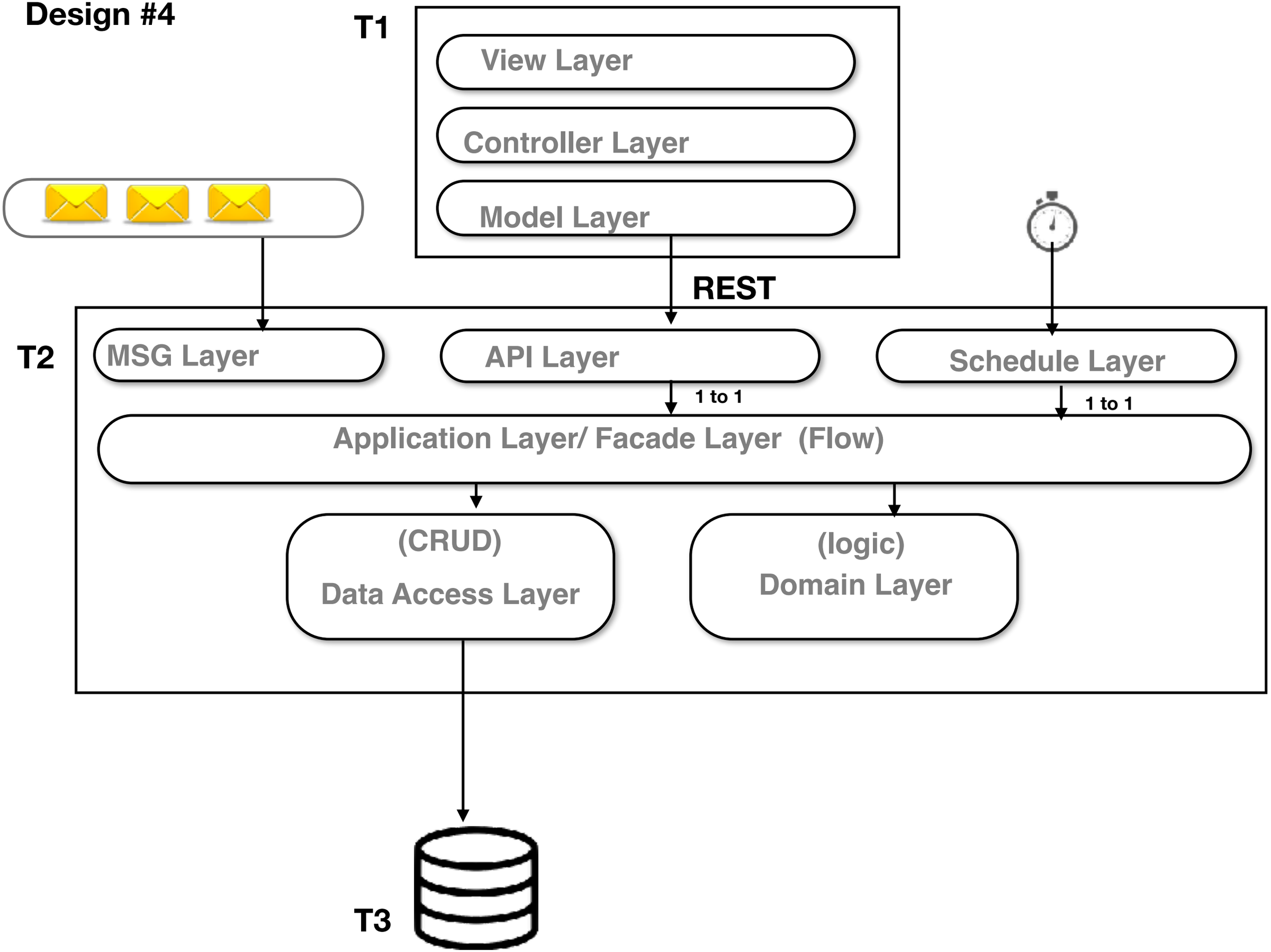
Data base Layer

Technology

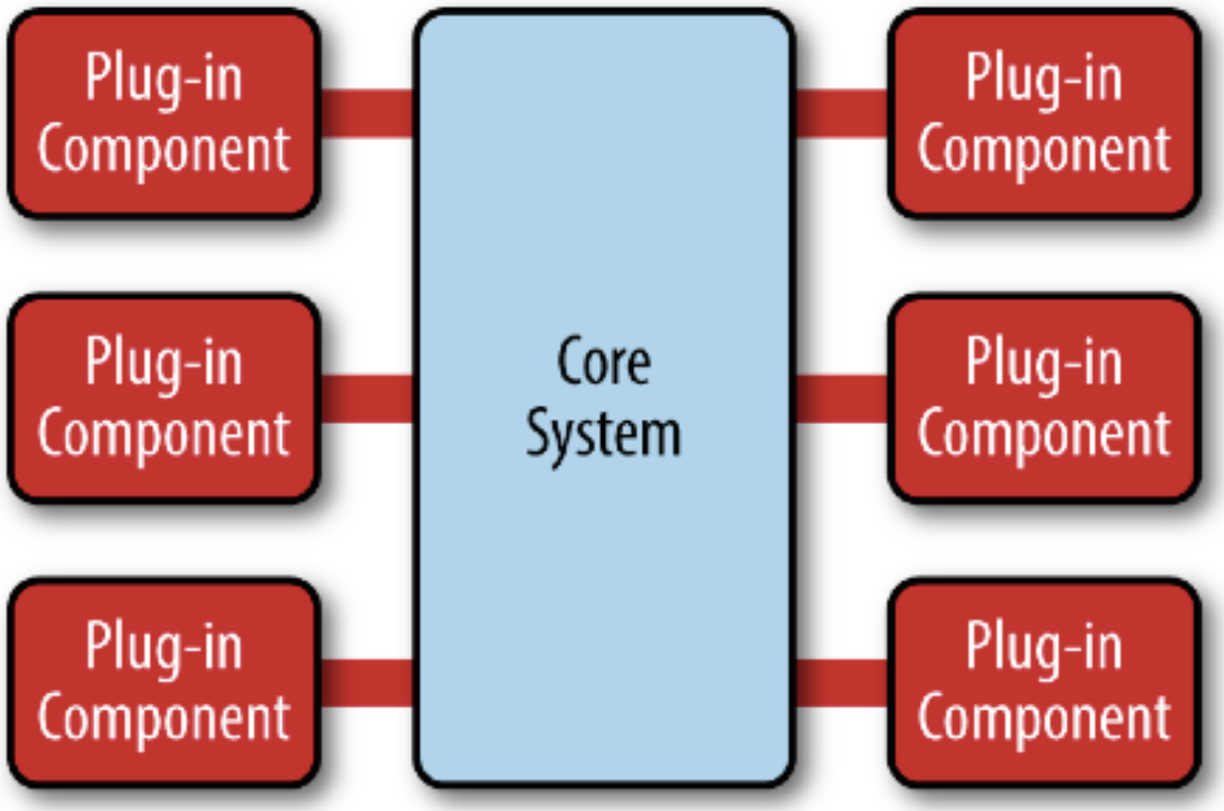
Hexagonal Arch



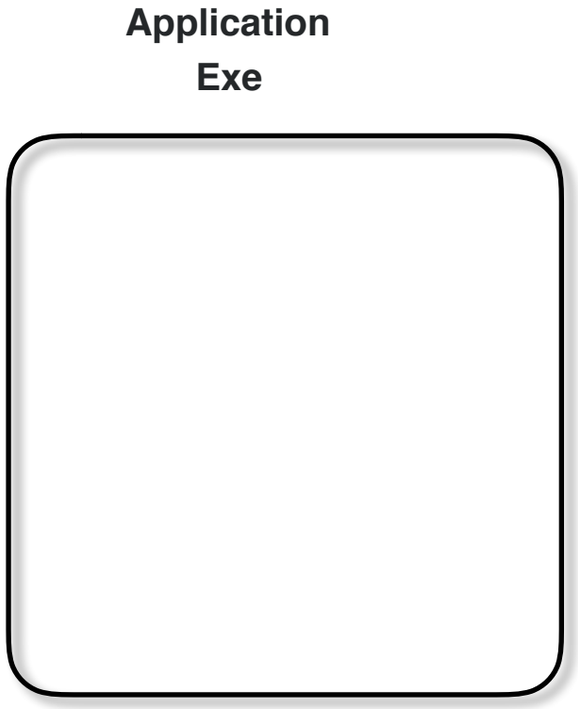
Design #4


T1
View Layer
Controller Layer
Model Layer
REST
T2
MSG Layer
API Layer
Schedule Layer
1 to 1
1 to 1
Application Layer/ Facade Layer (Flow)
(CRUD)
Data Access Layer
(logic)
Domain Layer
T3

Microkernel Architecture

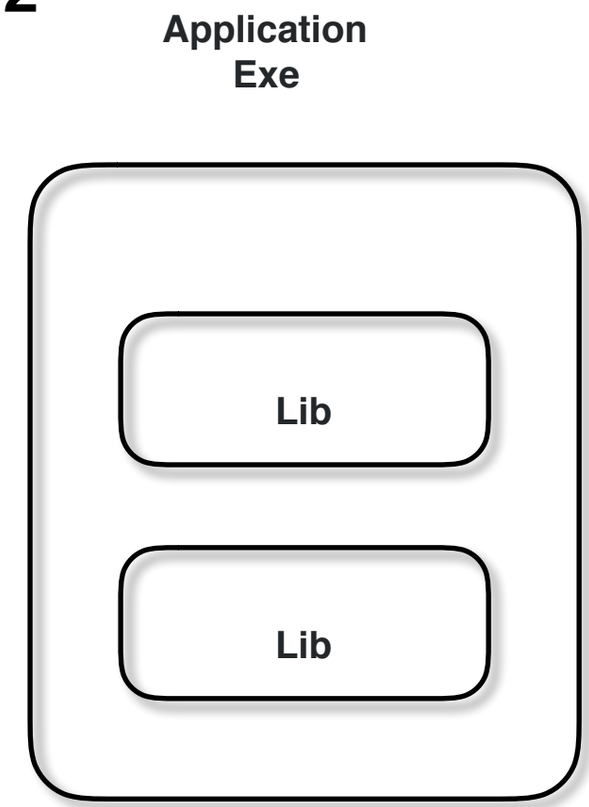


1



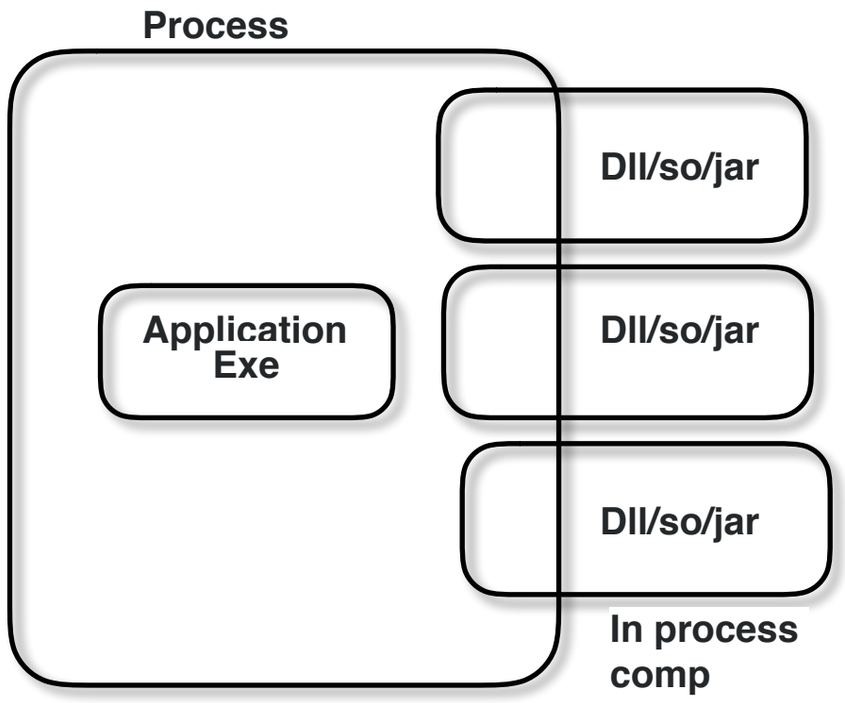
Compile time
monolithic

2



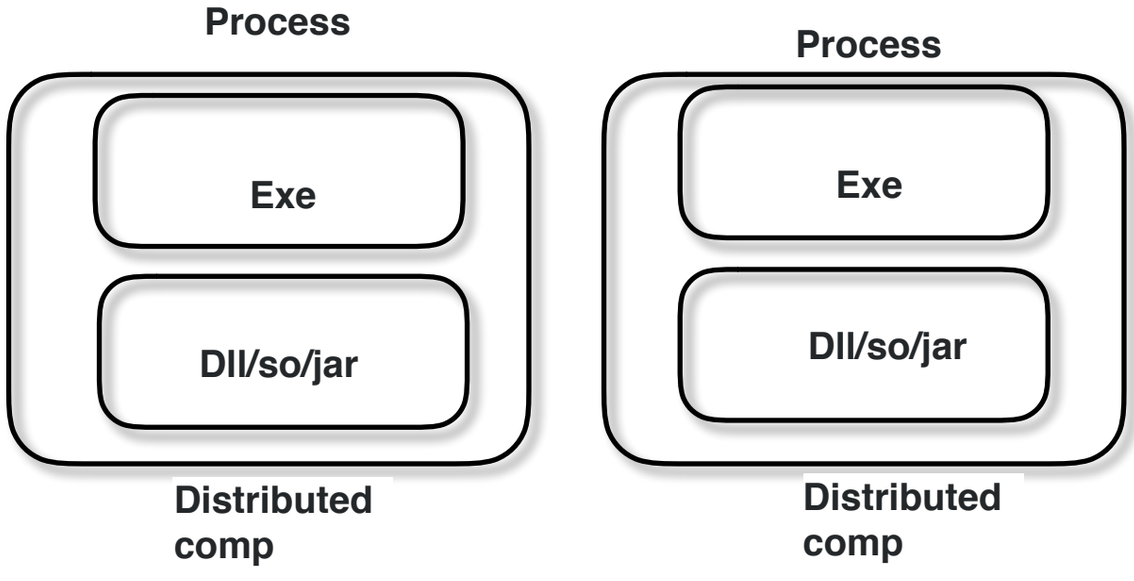
Link time
monolithic

3



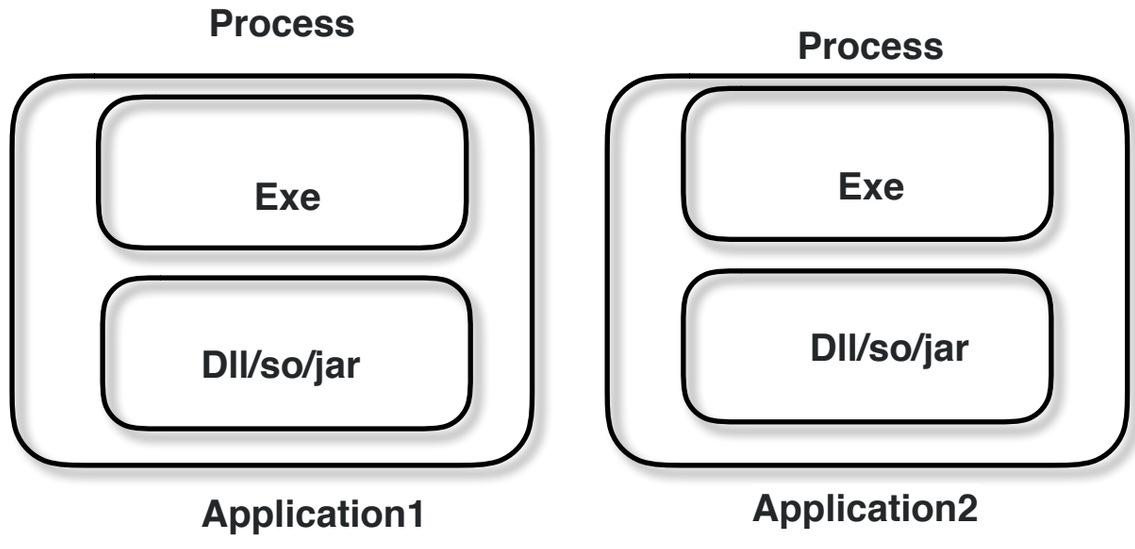
Runtime
monolithic

4

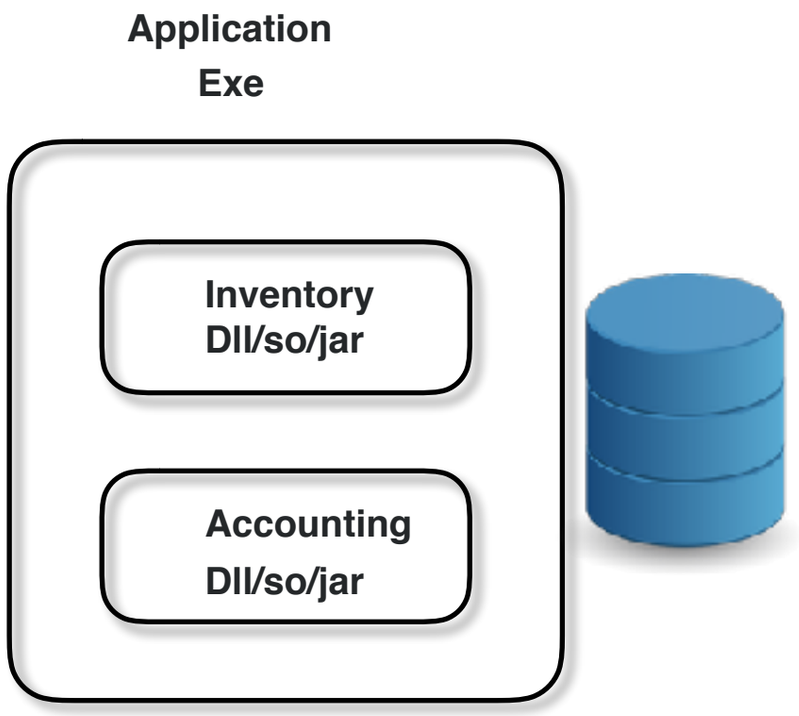


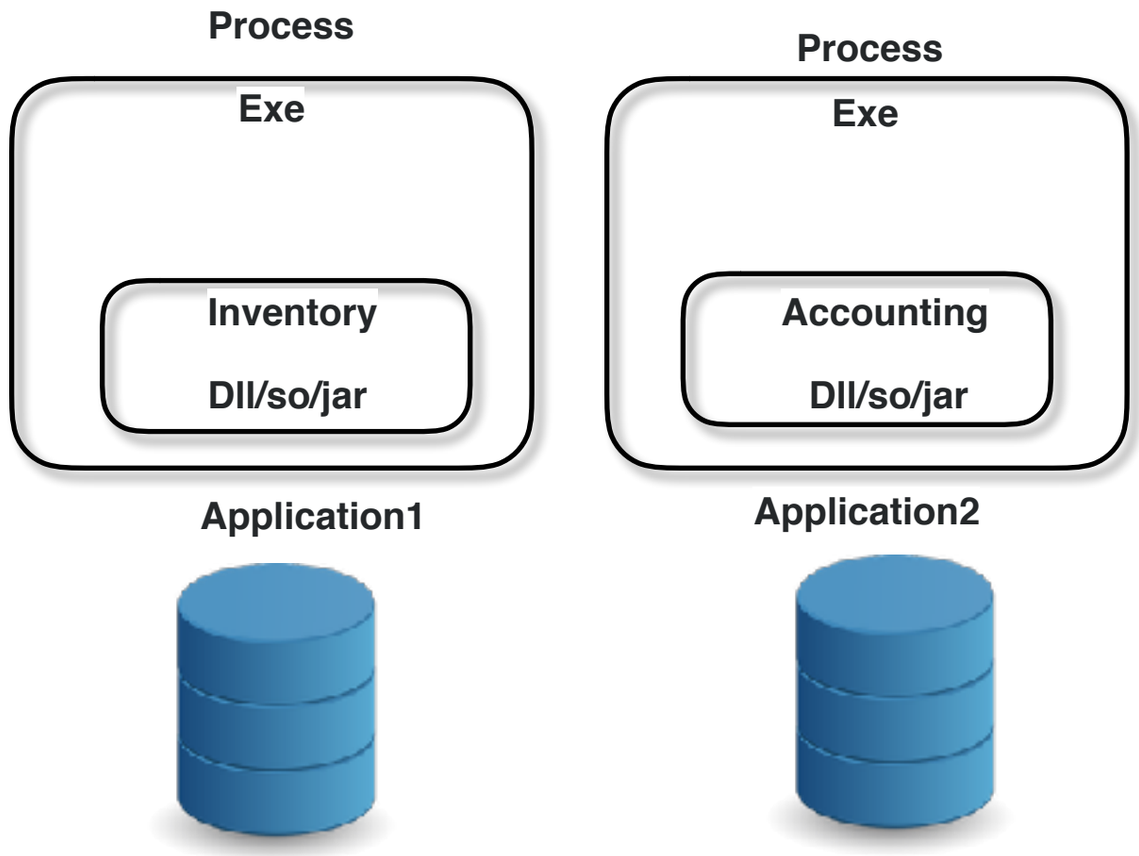
Distributed Application

5



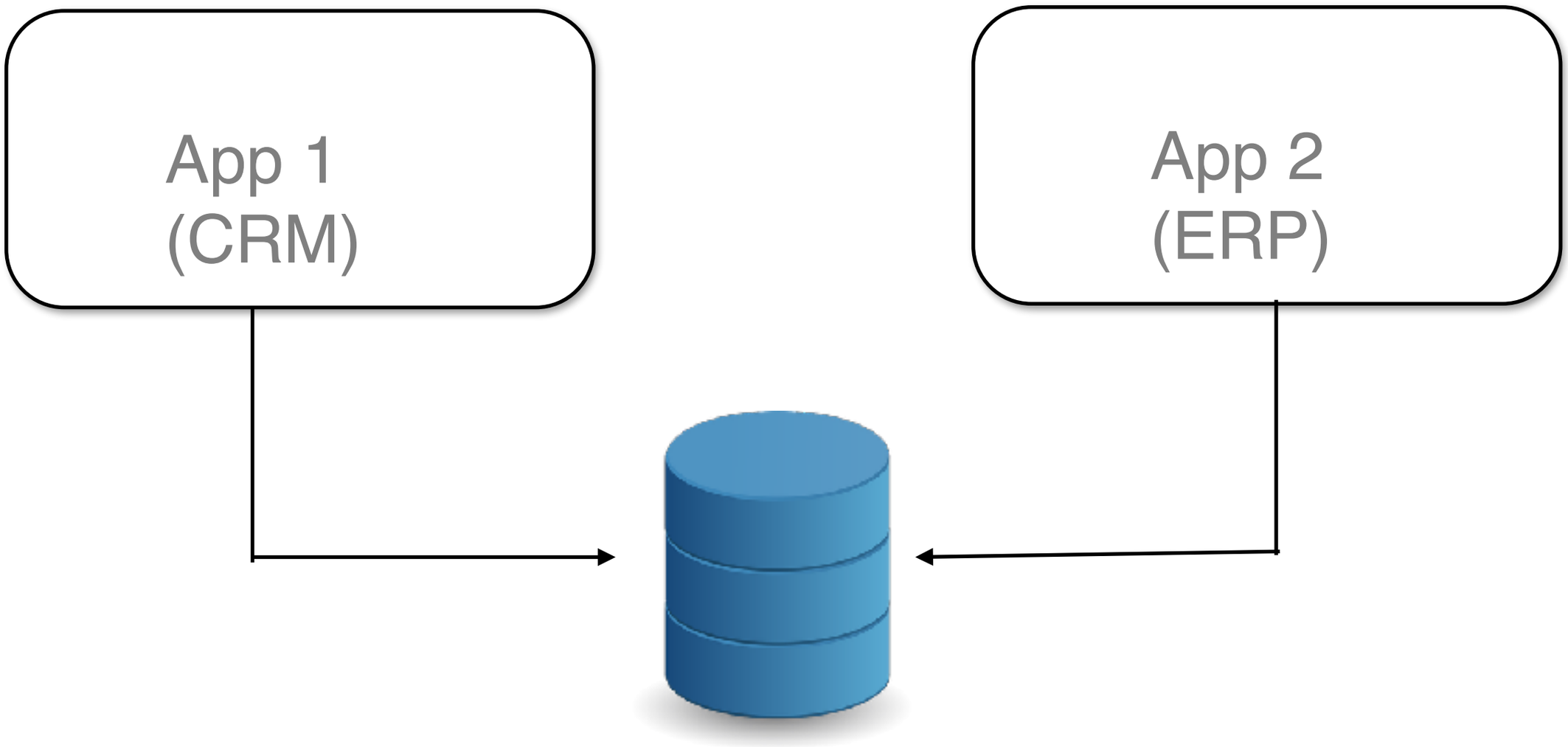
Micro Application


Application
Exe
Inventory
Dll/so/jar
Accounting
Dll/so/jar

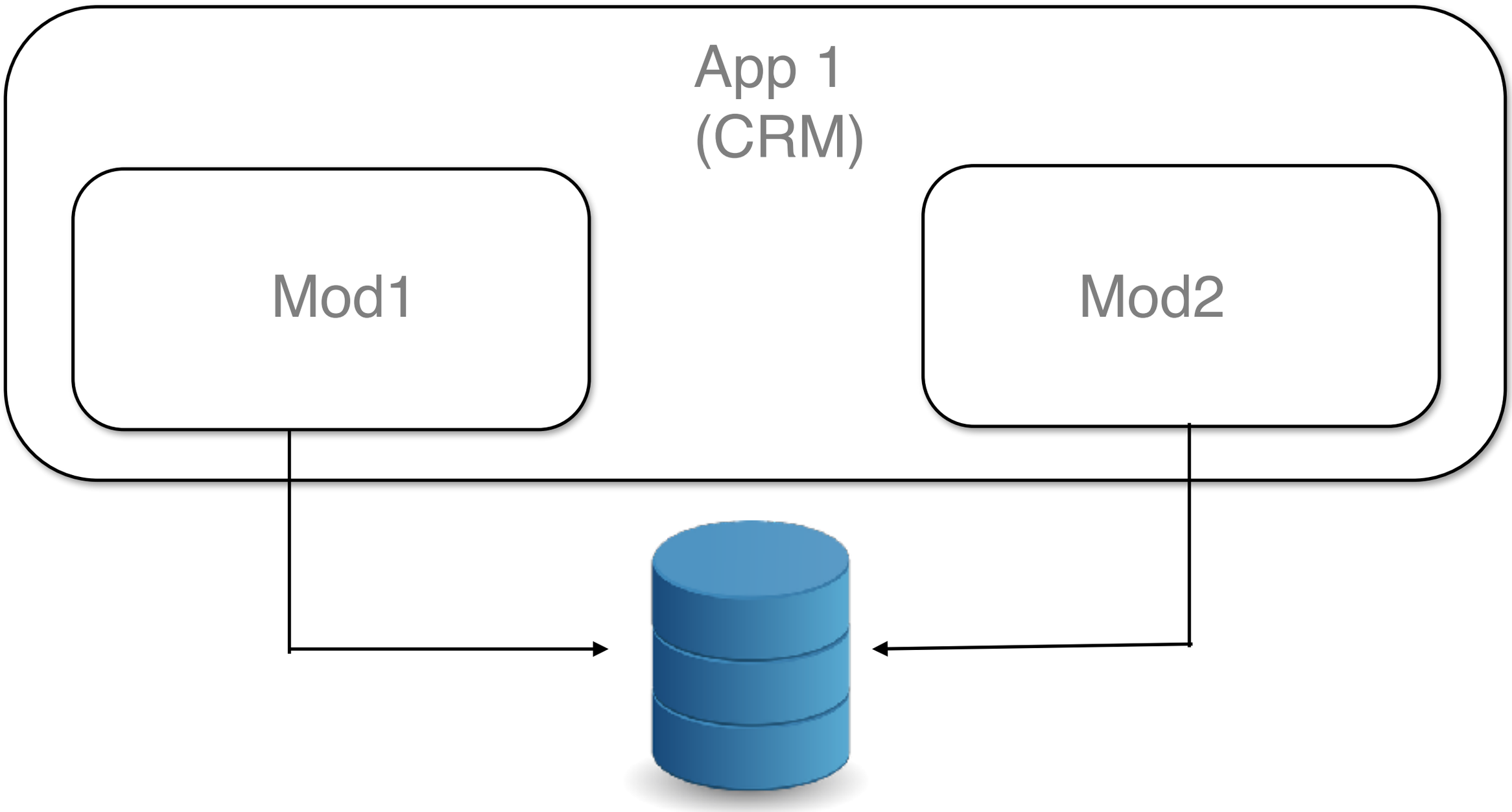

Process
Exe
Inventory
Dll/so/jar
Process
Exe
Accounting
Dll/so/jar
Application1
Application2

	Distributed Component	Micro Application	W	Score
Database / Storage	Shared	Not Shared	2	
Infra (Hosting)	Shared	Not Shared	3	
Sorce Control	Shared	Not Shared	2	
CI/CD (Build Server)	Shared	Not Shared	3	
Fun Requirements	Shared	Not Shared	1	
SCRUM Team / Sprint	Shared	Not Shared	1	
Test Cases	Shared	Not Shared	1	
Architecture	Shared	Not Shared	1	
Technology Stack / Fwks	Shared	Not Shared	1	

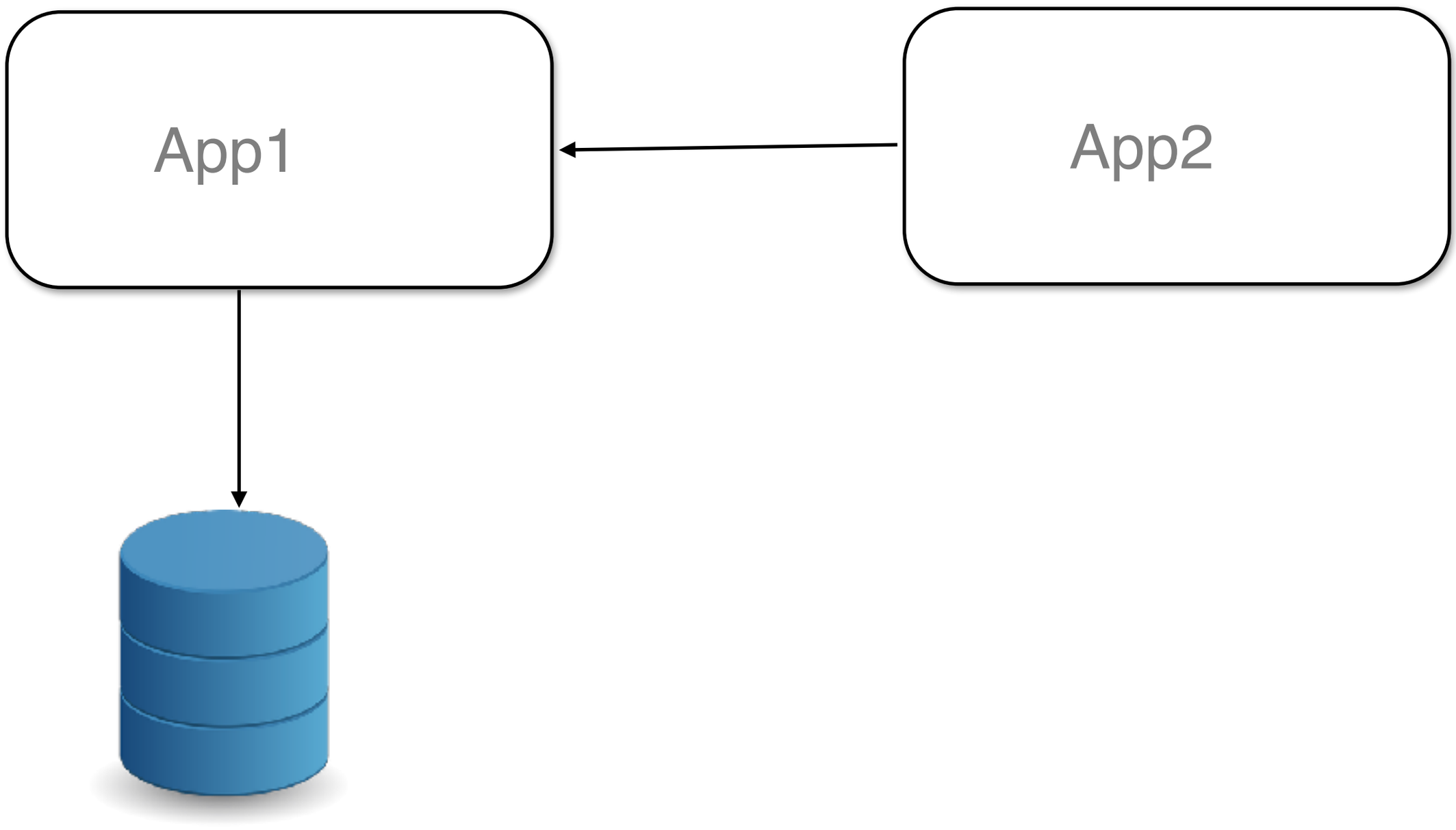
1. Database



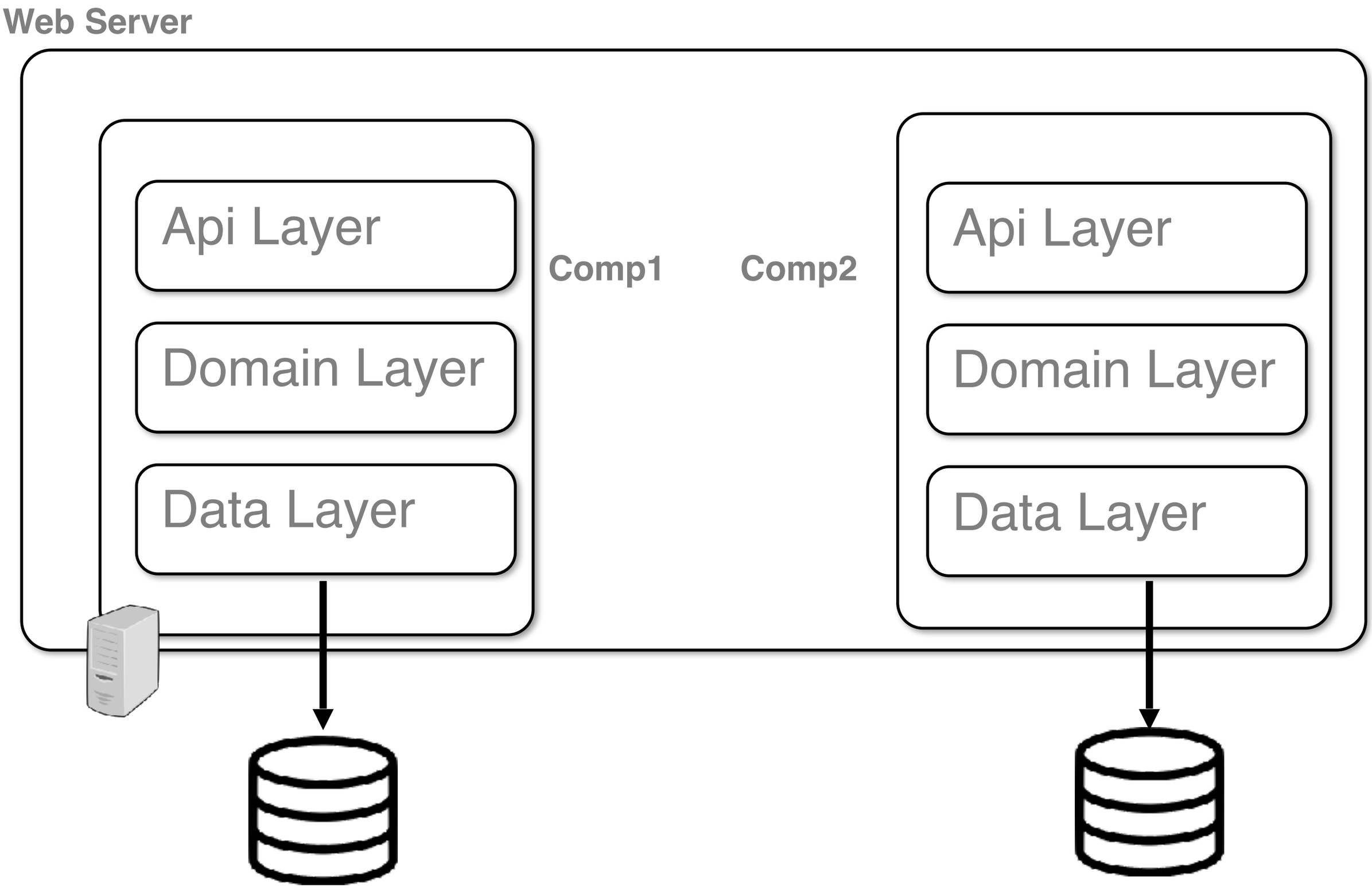
1. Database



1. Database



2. Infrastructure



1

UI App - SPA

Web Server

API Layer

Domain Layer
<<feature1>>

Data Layer

Angular (SPA)

2

Browser

Web Server

UI Layer -
Server pages

Domain Layer
<<feature2>>

Data Layer

JSP, Servlet, JSF (Server pages)

3

Desktop

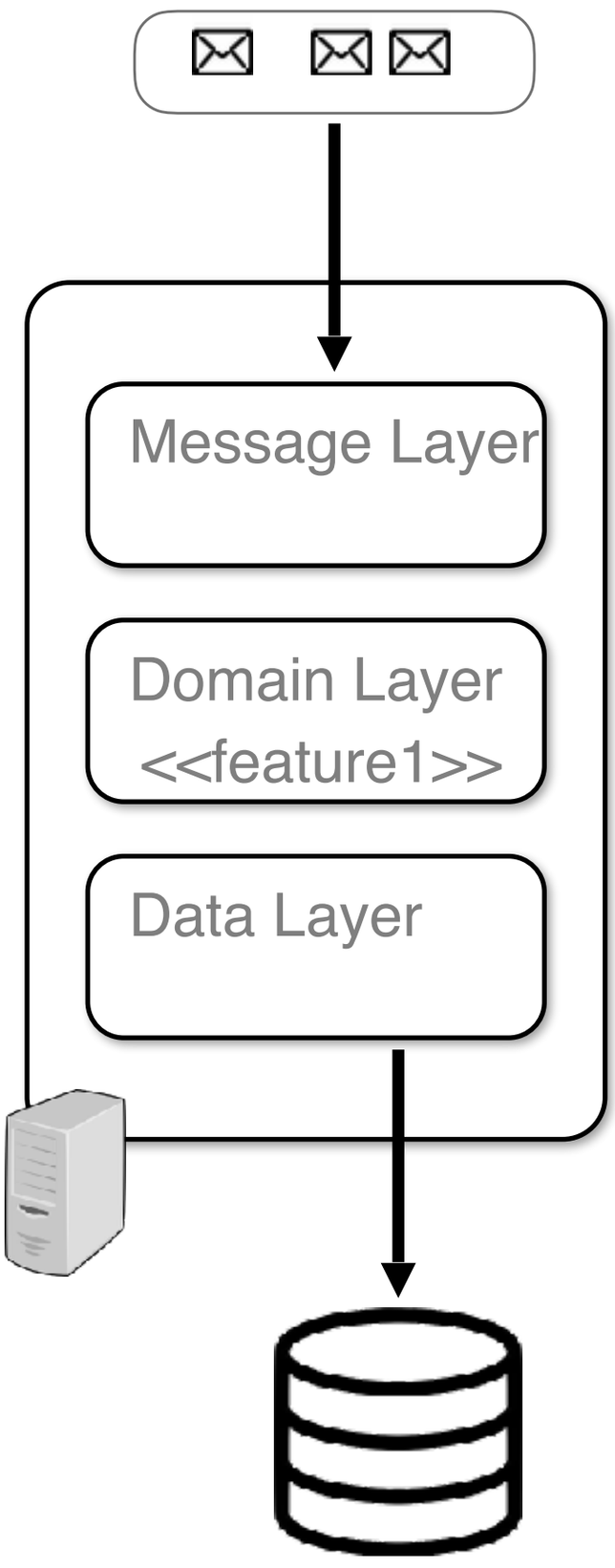
Native UI Layer

Domain Layer
<<feature3>>

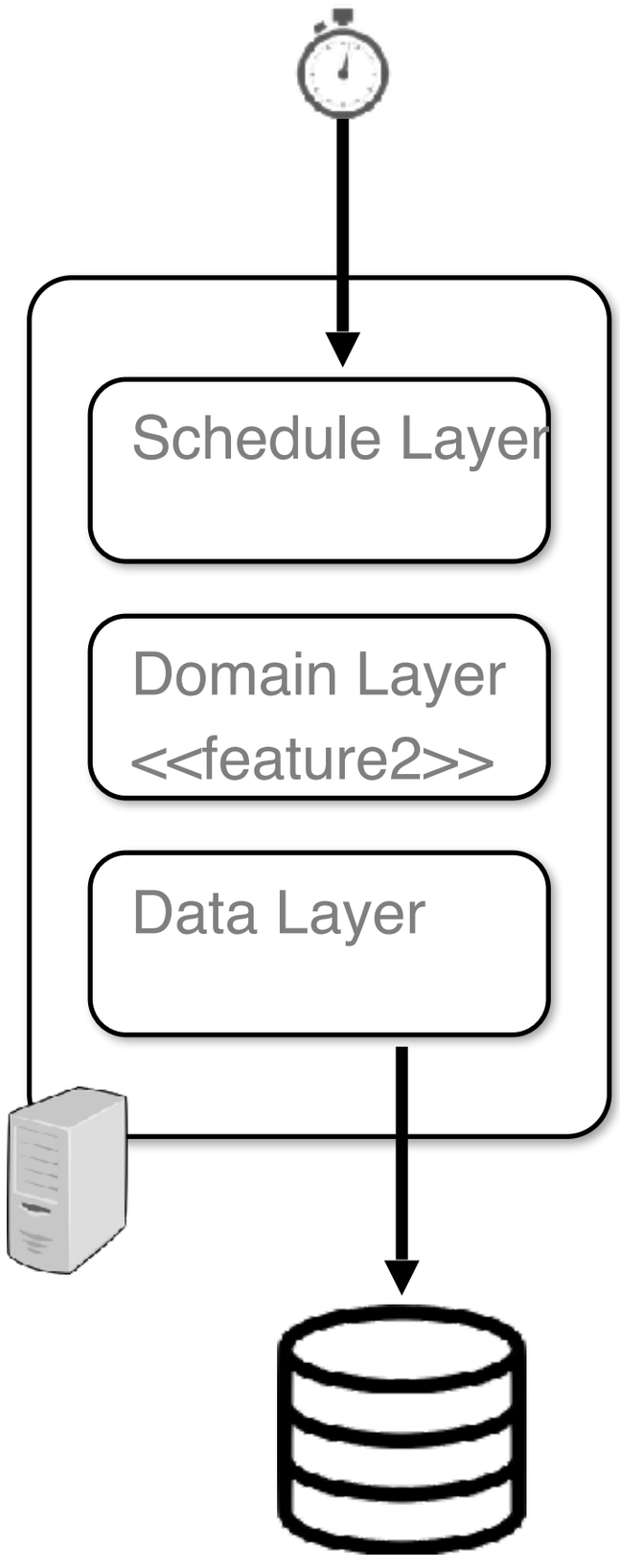
Data Layer

Swing, Android, Ios (Native)

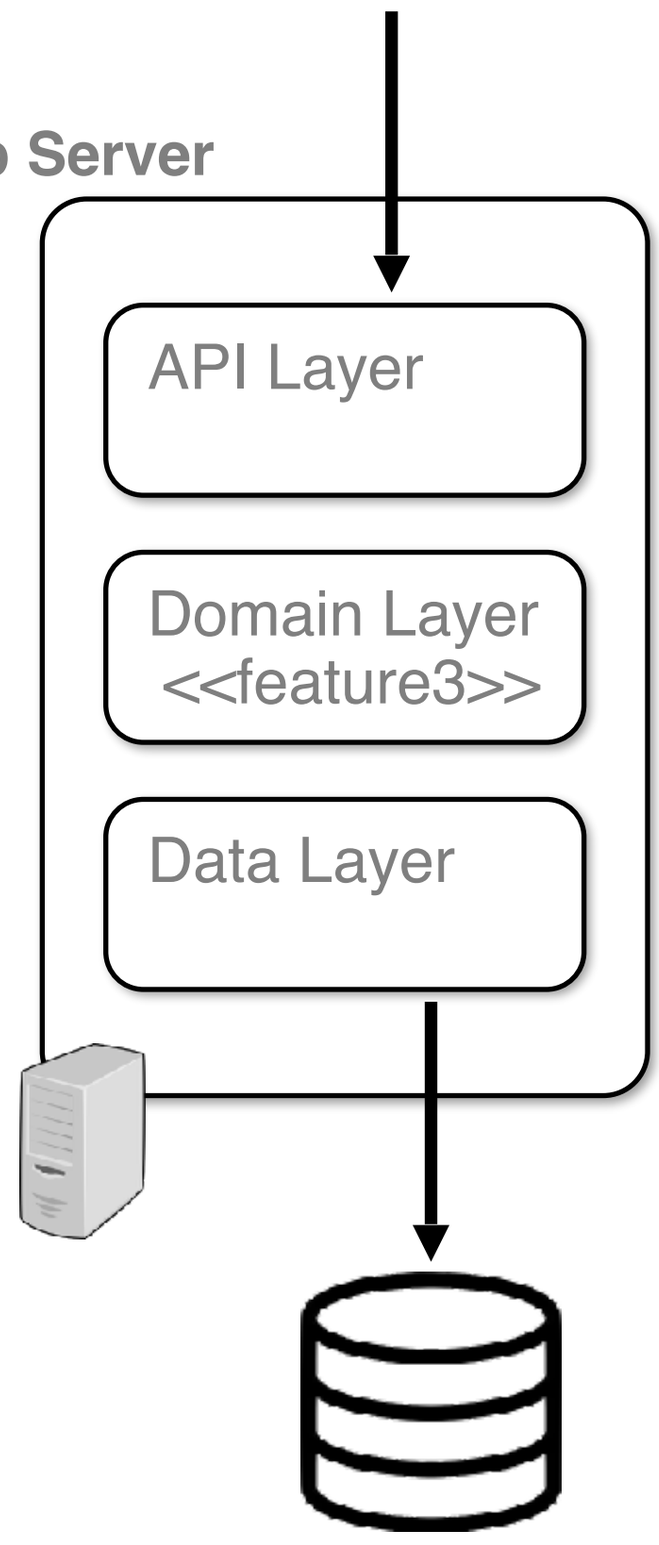
1


Message Layer
Domain Layer
<<feature1>>
Data Layer

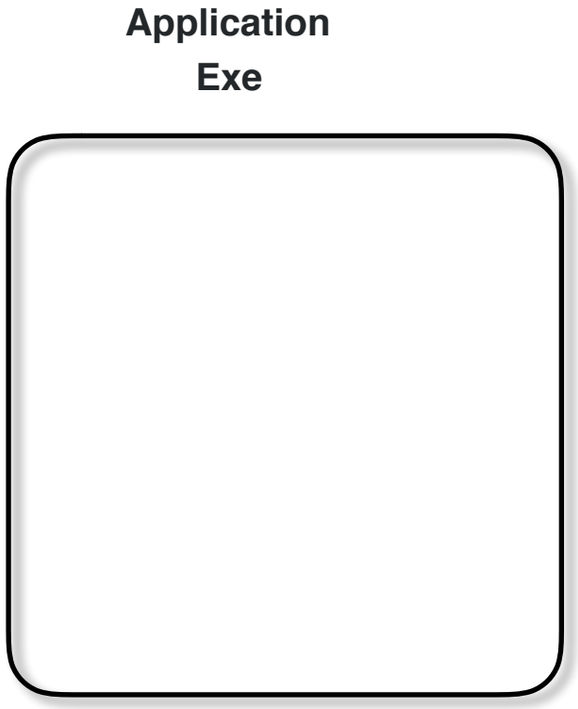
2


Schedule Layer
Domain Layer
<<feature2>>
Data Layer

3

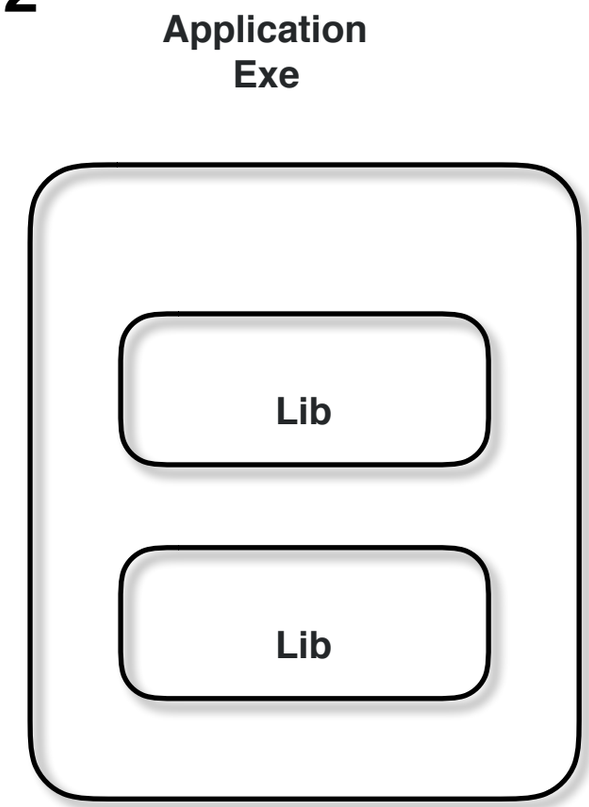

Web Server
API Layer
Domain Layer
<<feature3>>
Data Layer

1



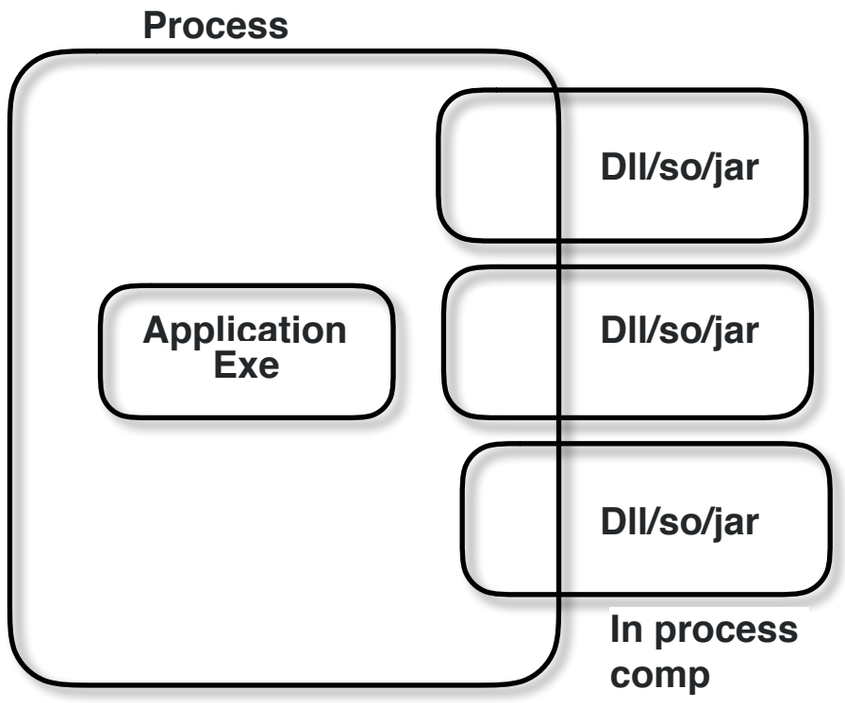
Compile time
monolithic

2



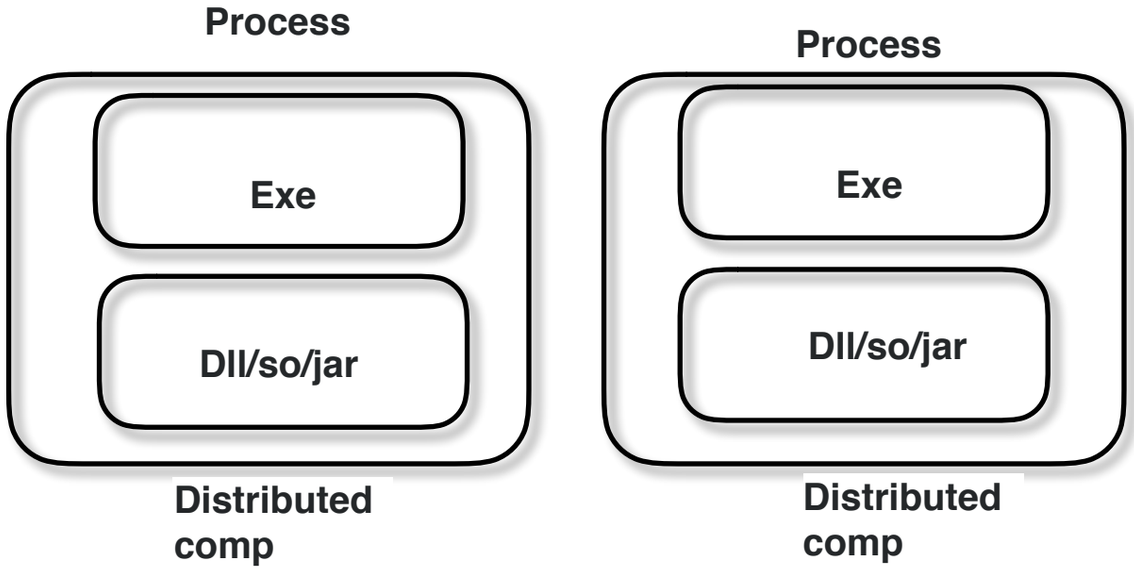
Link time
monolithic

3



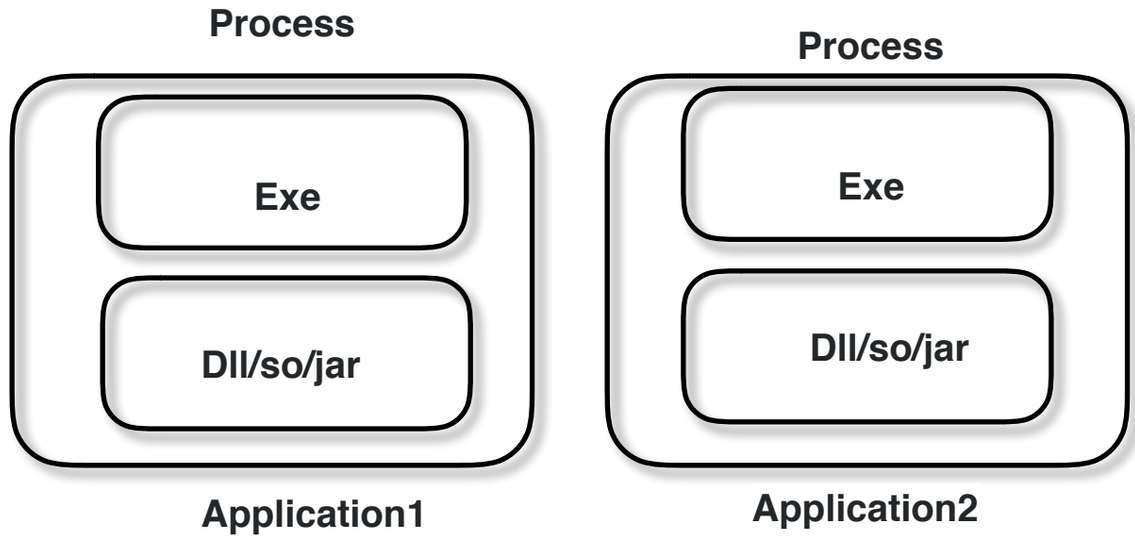
Runtime
monolithic

4



Distributed Application

5

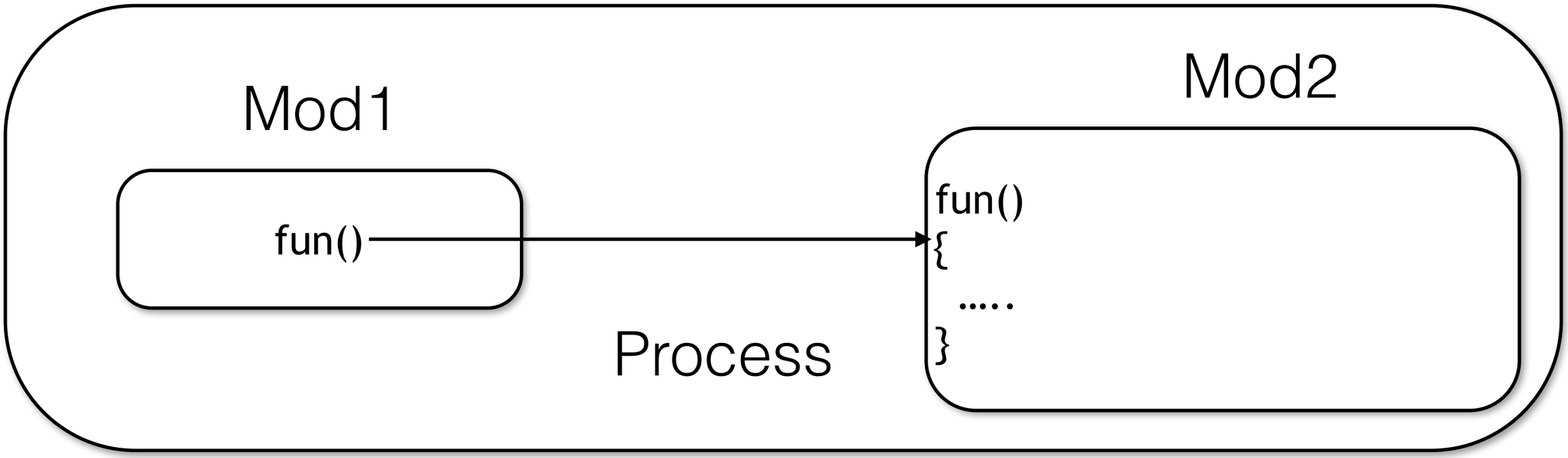


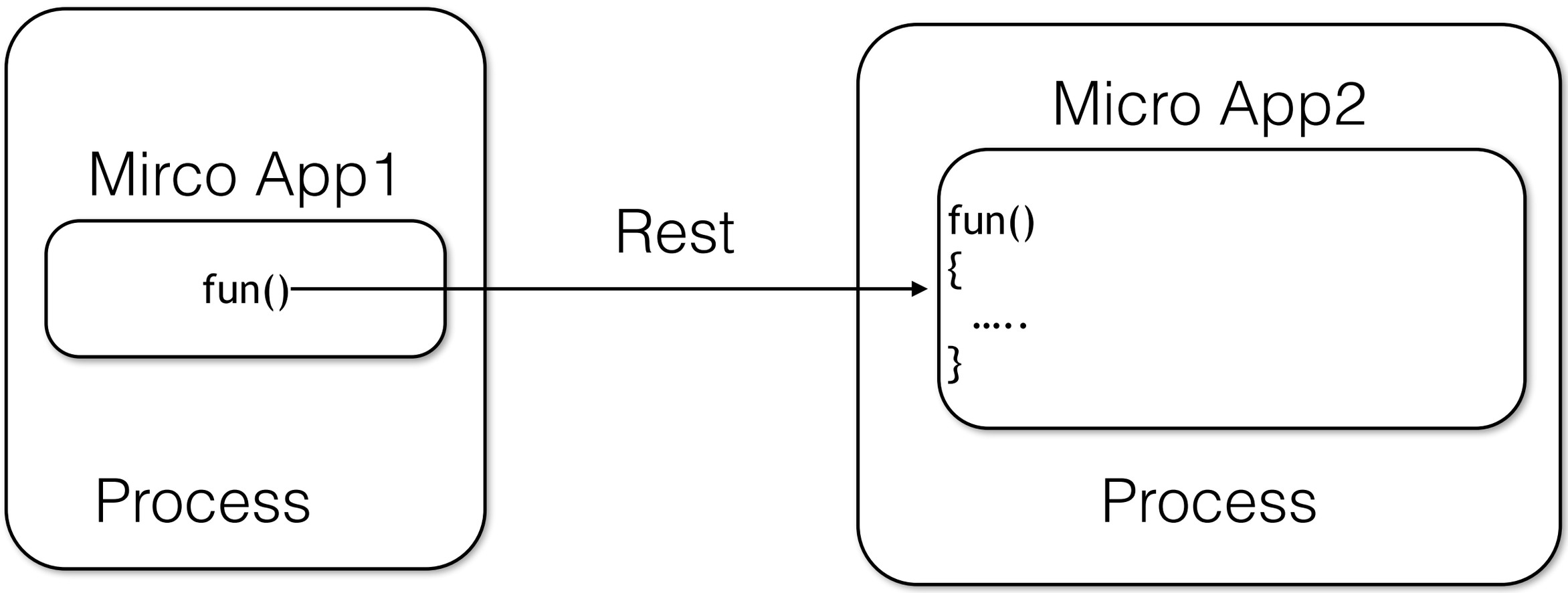
Micro Application

	Pros/ Cons	Solution
Development time	- -	
Learning Curve	- -	
Resource Performance (CPU, Memory, I/O)	- -	
Db Transaction Management	- -	
Views / Report / Dash board/ join	- -	
Infra Cost	- -	Lambda, Azure Functions, Containers
Deployment effort	- -	
Debugging, Error Handling,	-	Distributed tracing (Jaeger)
Integration Test	- -	
debug/ error Log Mgmt	- - C	Logg agg (ELK, EFK, Splunk, …)
Config Mgmt	- - C	
Authentication	- - C	OAth2, OpenID connect, ..
Authorization	- - C	Claim Based
Audit Log mgmt	- - C	
Monitoring / Alerting	- - C	Kibana, grafana, Prometheus, ..
Data Security and Privacy	- -	
Build Pipeline (CI)	- -	
Agile Architecture (Agility to change)	++ +	
Feature Shipping (Agility to ship)/ CD	+ + +	
Scalability (volume - request, data,	+ +	
Resilence	+ +	
Availability	+ +	
Ability to do Polygot	+ +	
Maintainability (easy to change)	+ + +	

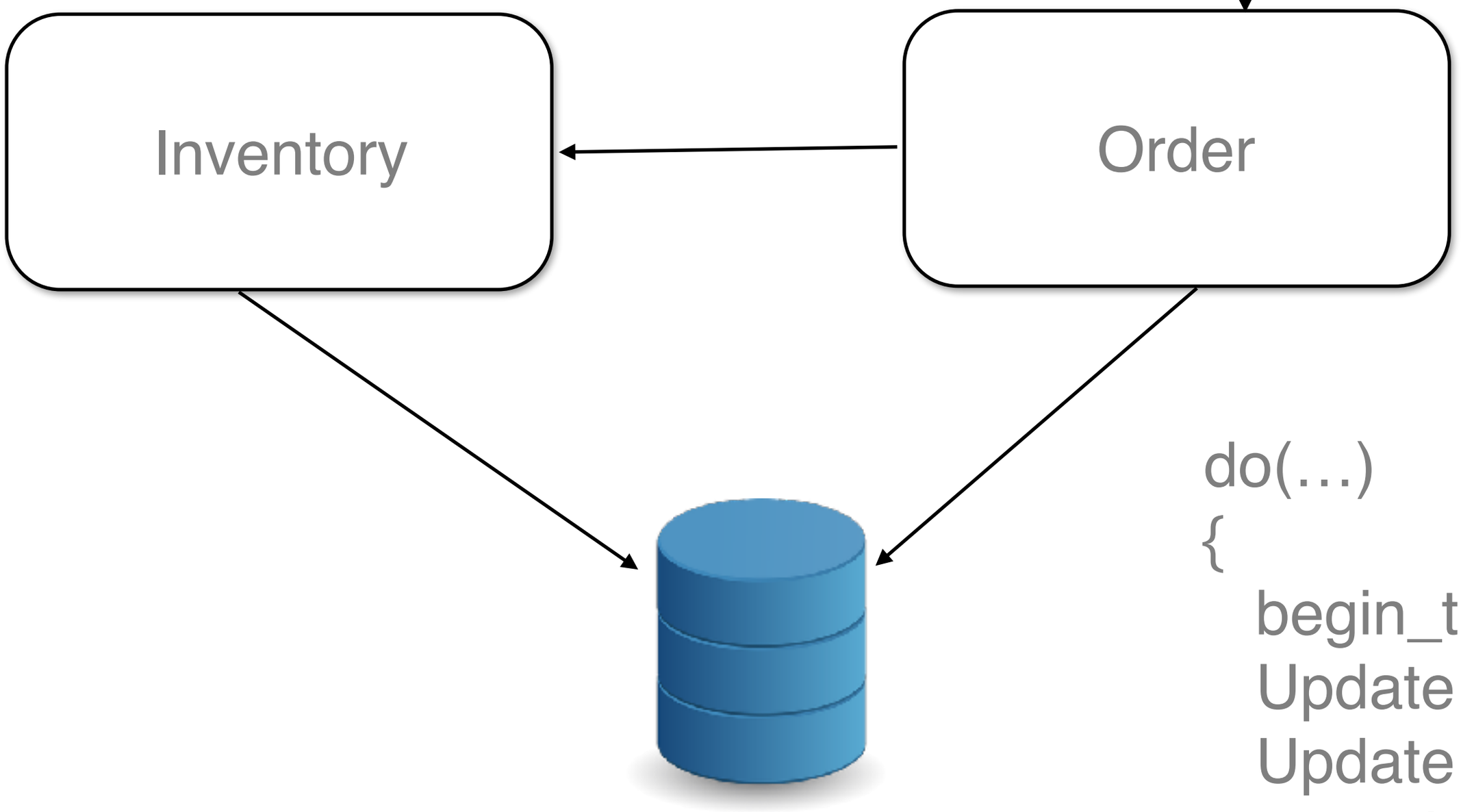
	CPU Cycles
a + b	3 cpu
Fun call	10 cpu cycles
Create Thread	200,000 cpu cycles
Destroy thread	100,000 cpu cycles
Write file	10,00,000
API Call	20,00,000
Db call	45,00,000

4. Development time



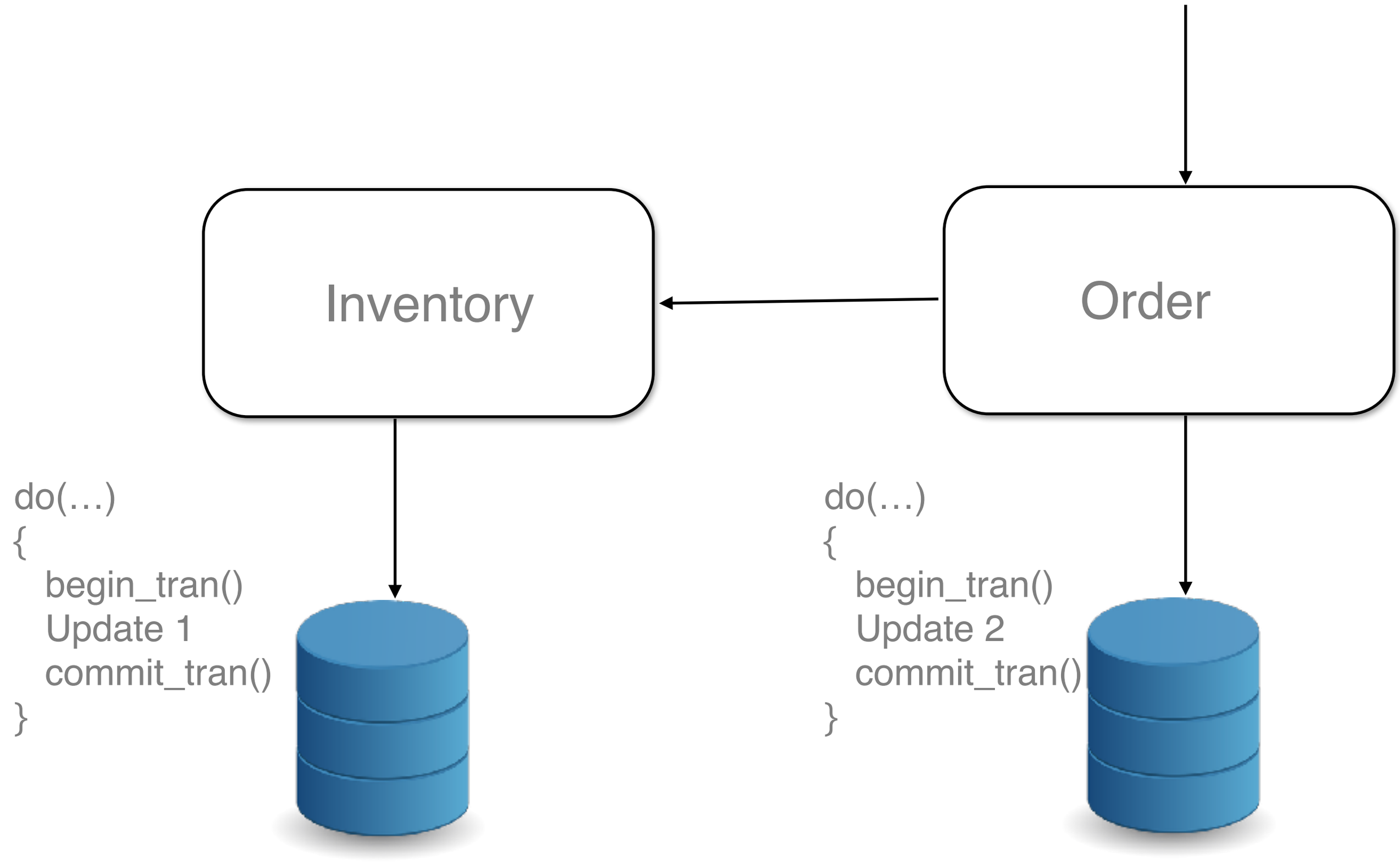


2. Db transaction



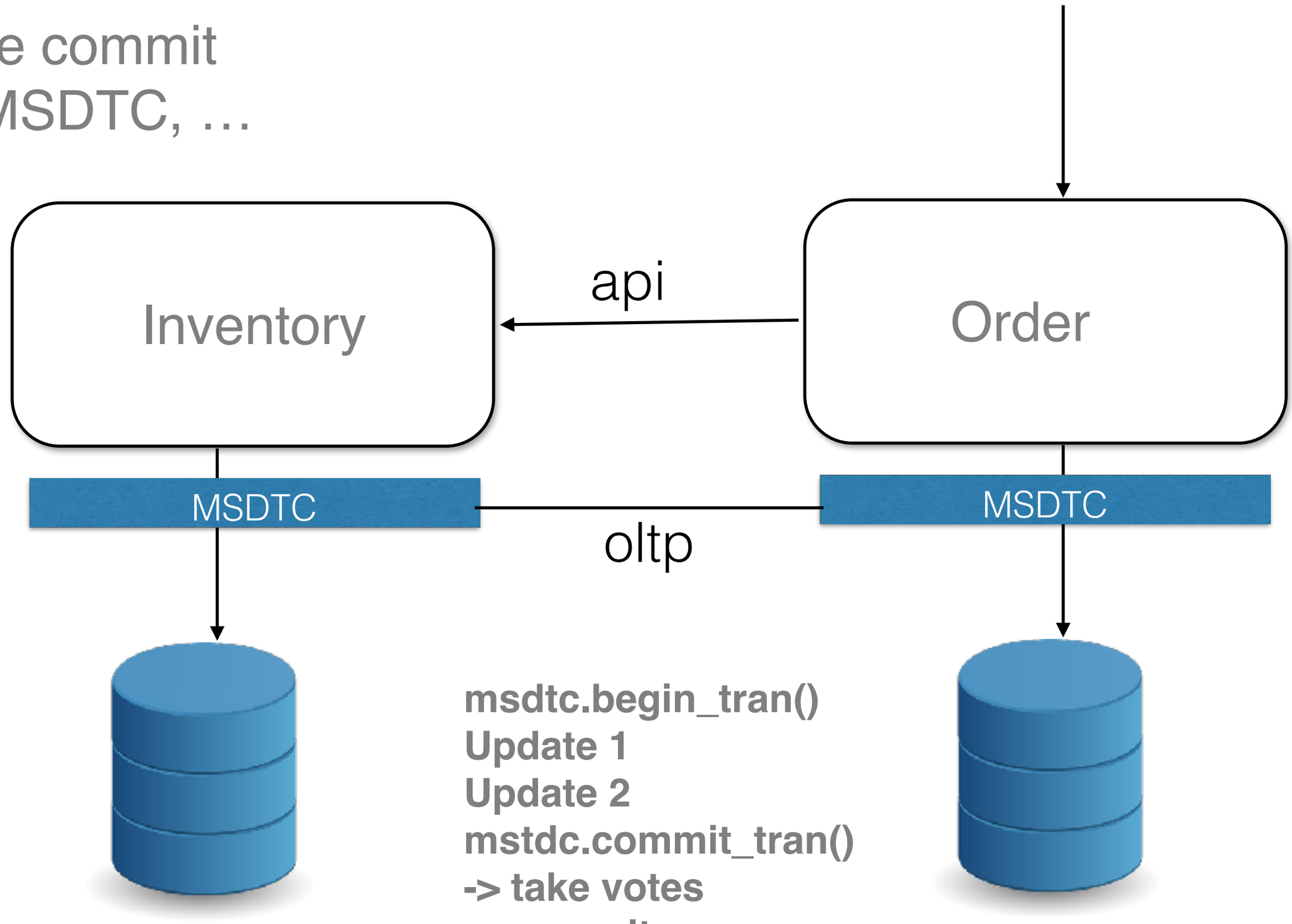

```
do(...)
{
   begin_tran()
   Update 1
   Update 2
   commit_tran()
}
```

2. Db transaction



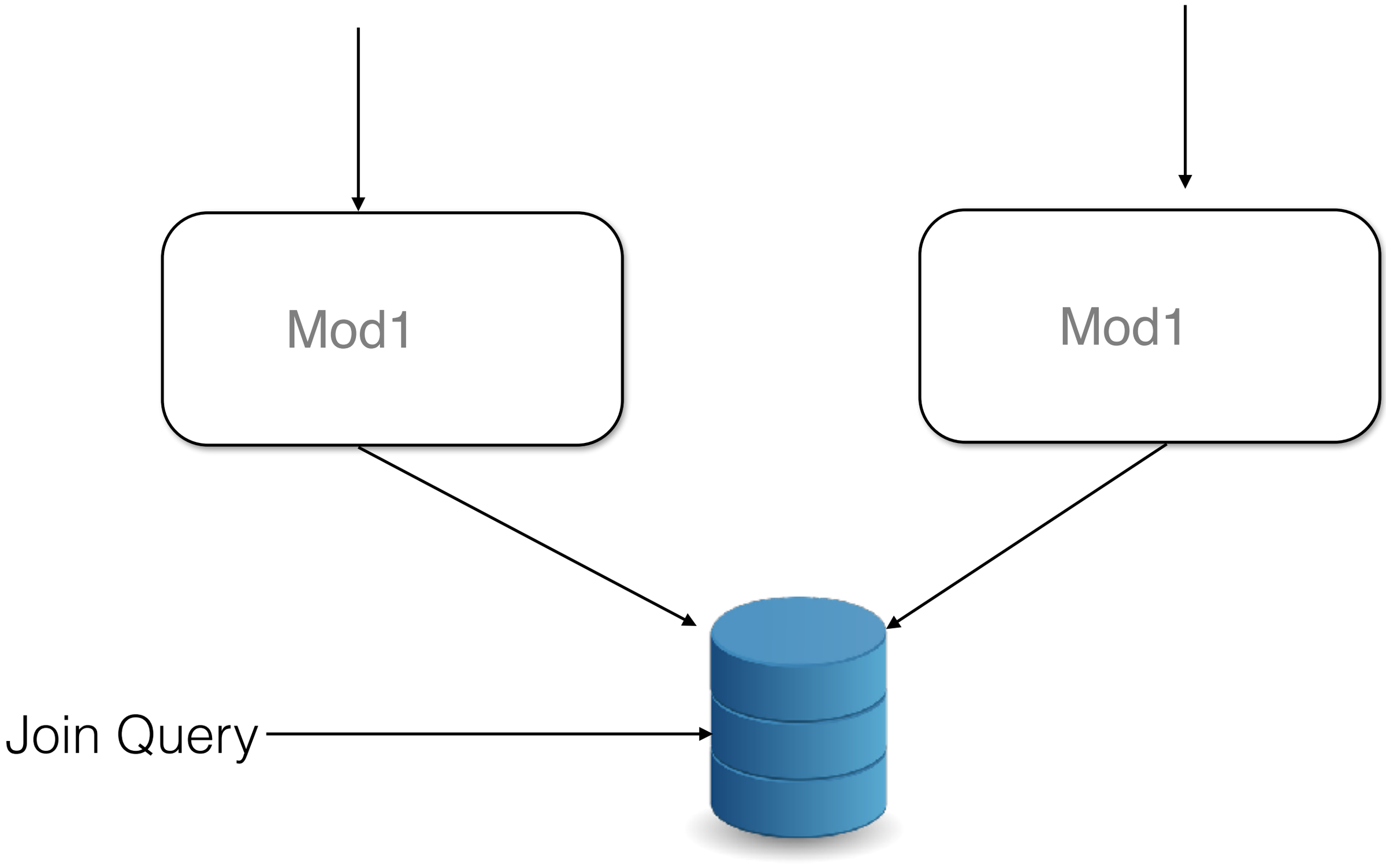
2. Db transaction

2 phase commit
JTX , MSDTC, …

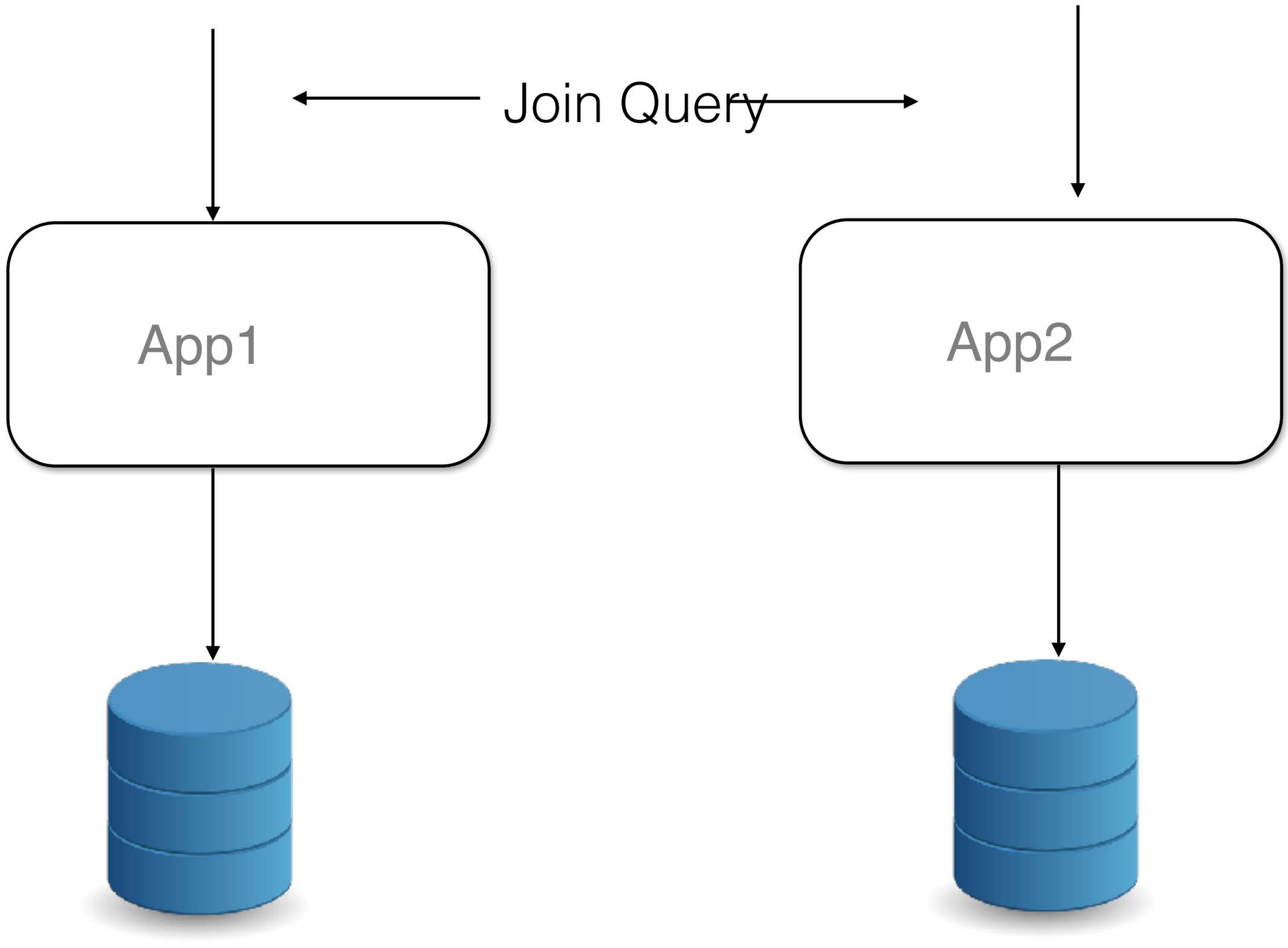


msdtc.begin_tran()
Update 1
Update 2
mstdc.commit_tran()
-> take votes
-> commit

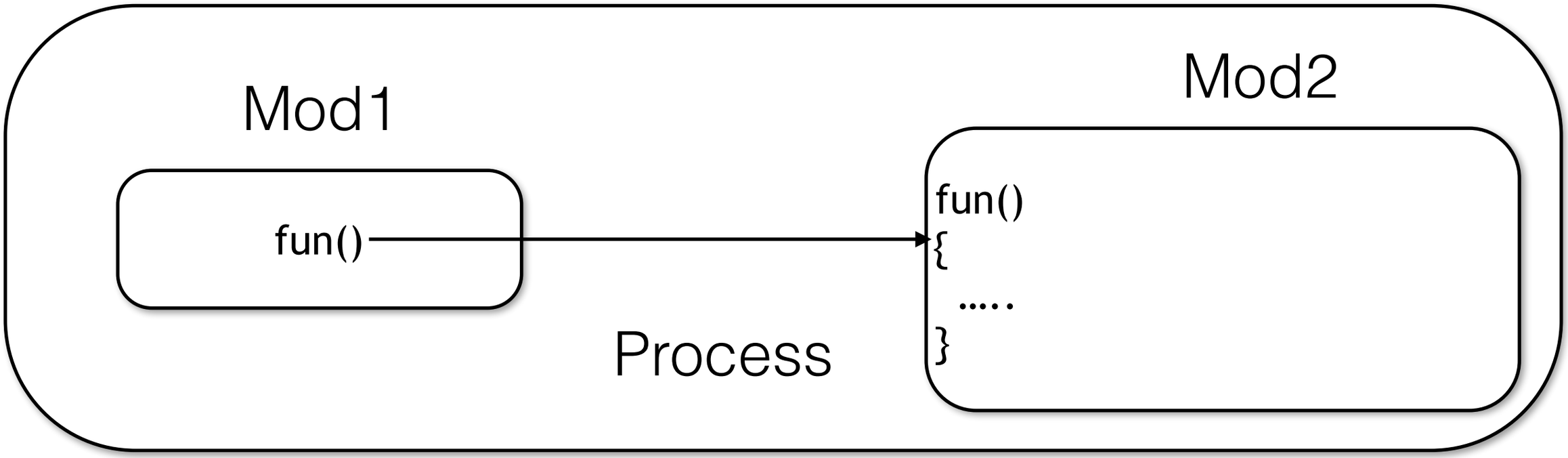
3. Db query

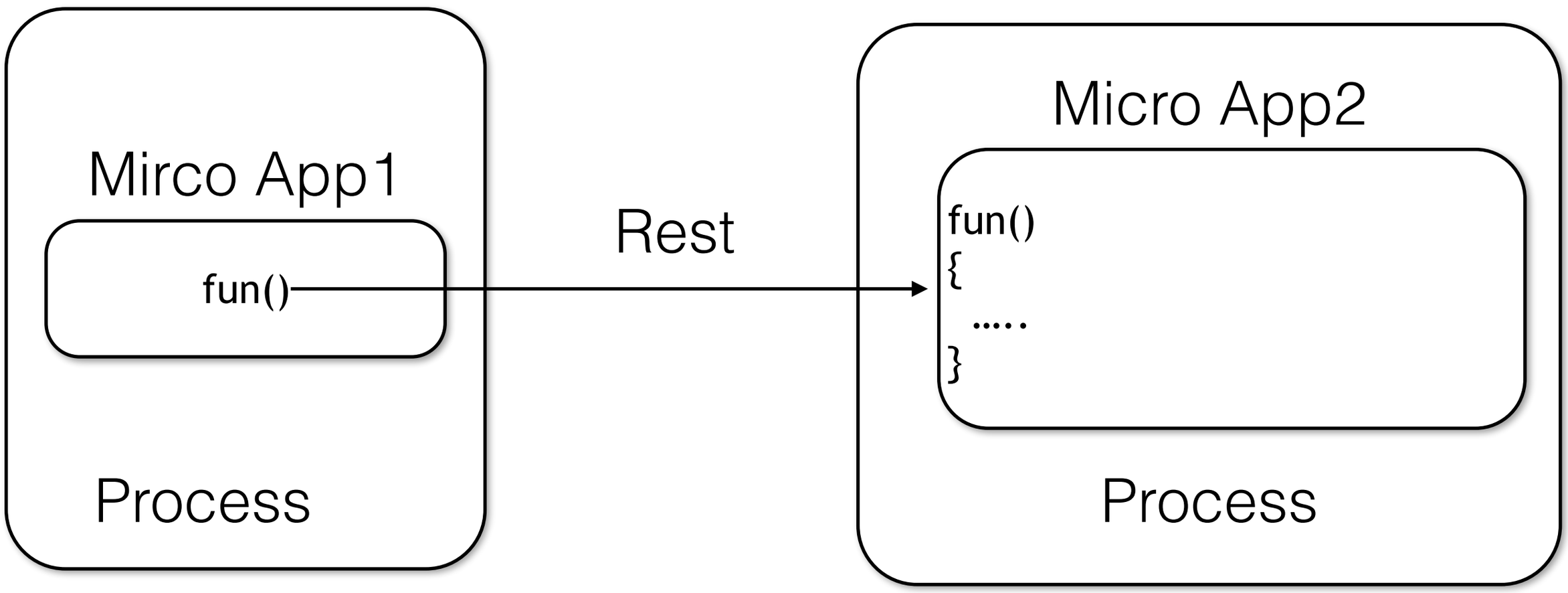


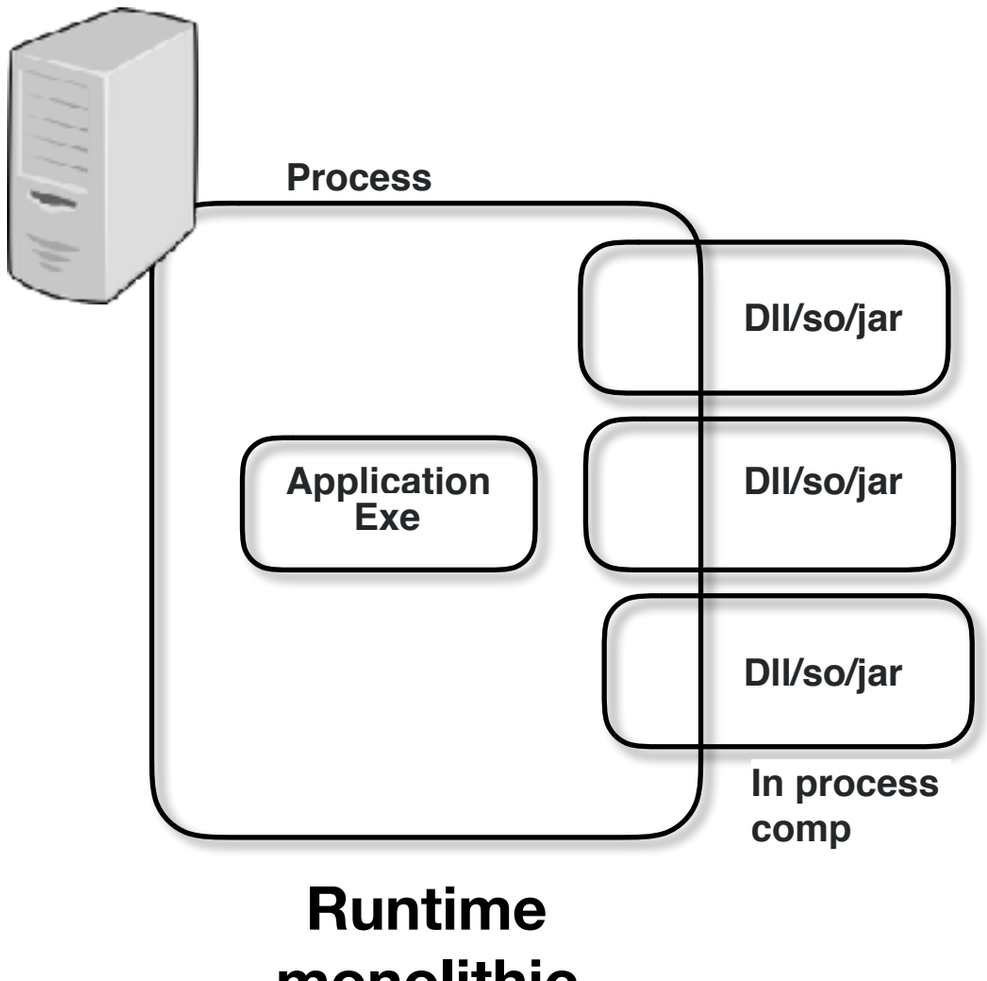
3. Query



4. Development time

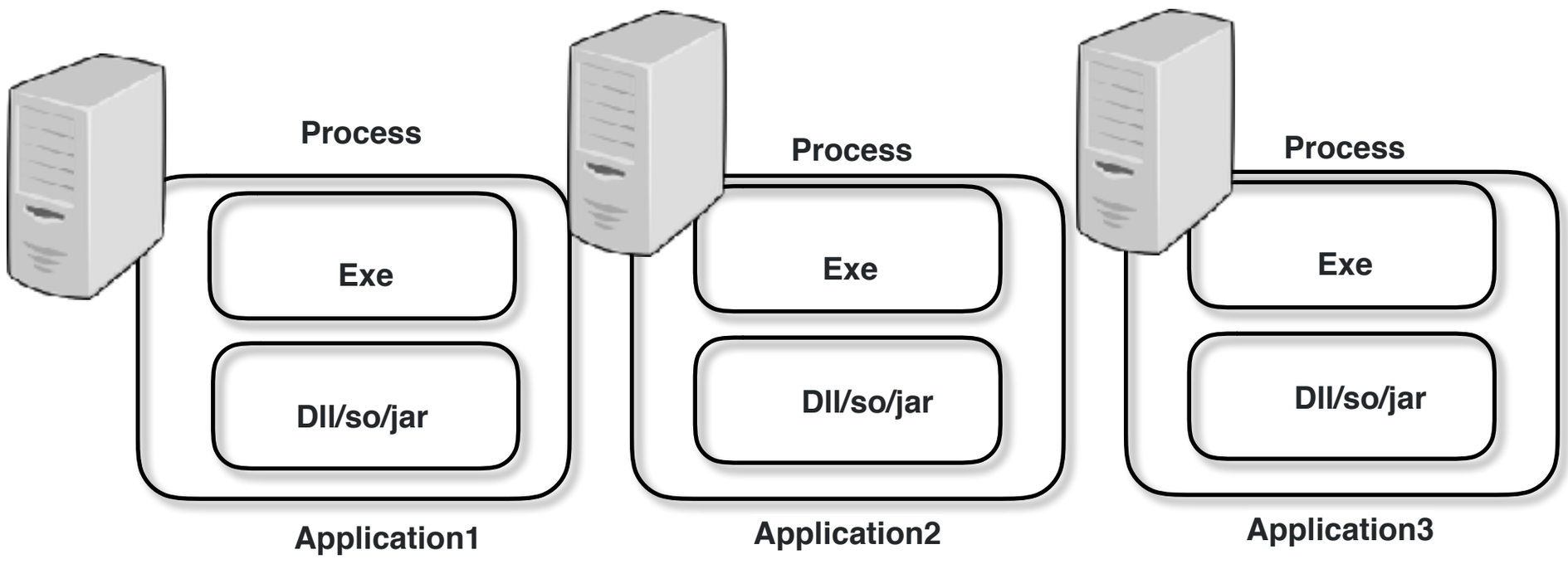






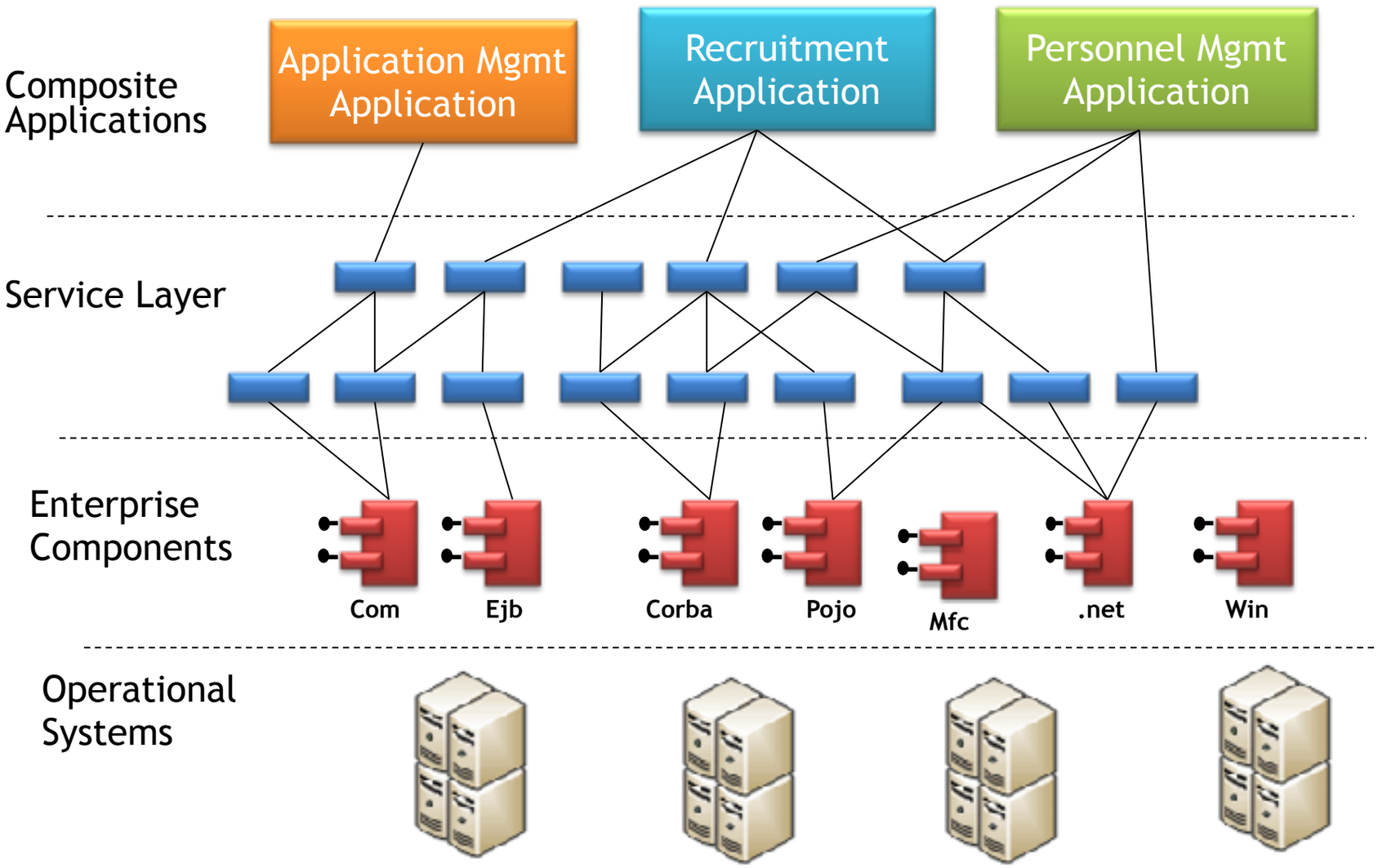
Runtime

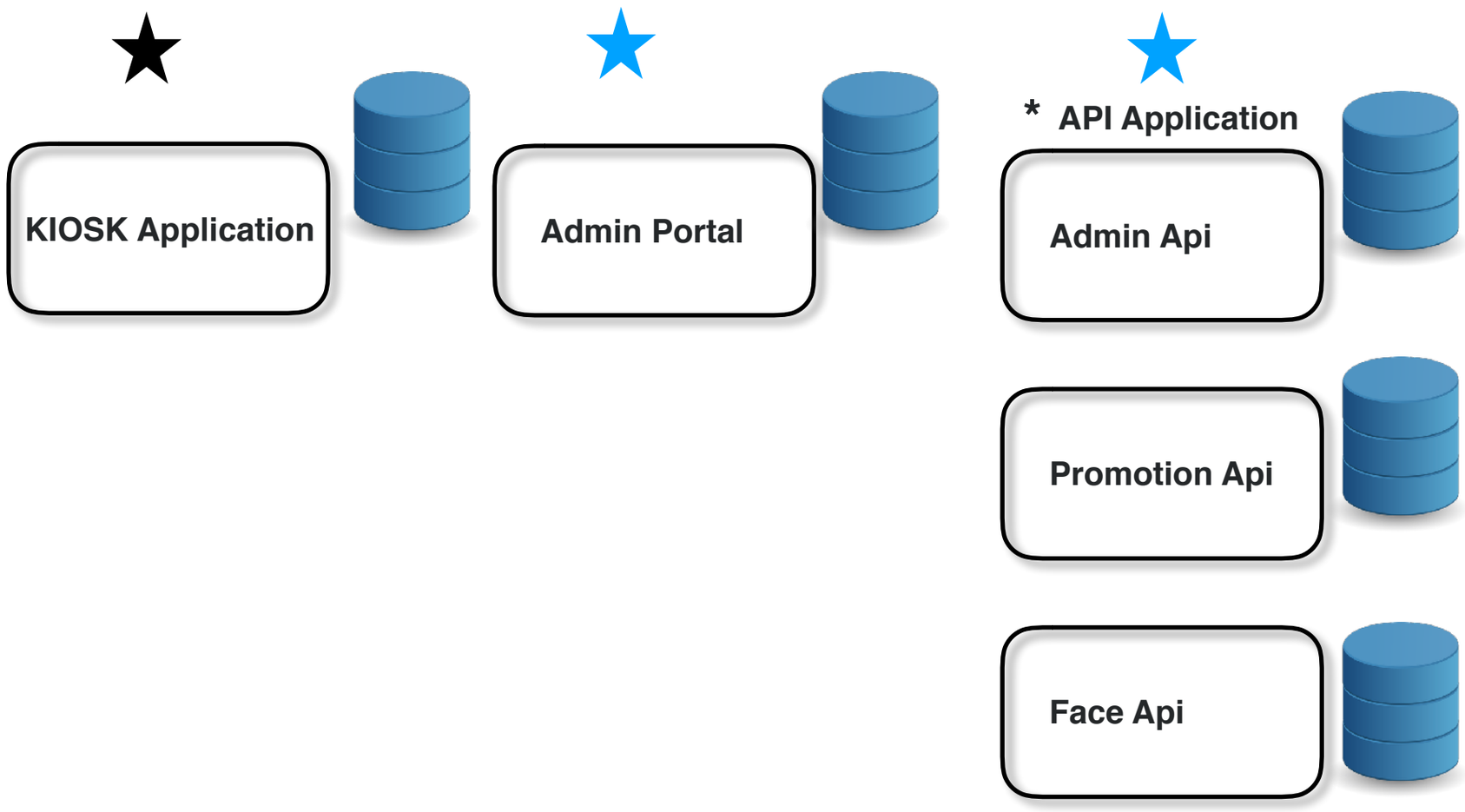
monolithic



Micro Application

SOA


Composite Applications
Application Mgmt Application
Recruitment Application
Personnel Mgmt Application
Service Layer
Enterprise Components
Com
Ejb
Corba
Pojo
Mfc
.net
Win
Operational Systems


KIOSK Application
Admin Portal
* API Application
Admin Api
Promotion Api
Face Api

< 10 TB

Binary data
(videos, images, audio, ..)

Lob data

Rest API Access
(access anywhere)

Cloud Object Storage

S3, Azure blob

File protocol Support

Cloud File Storage

EFS, Azure Files

Persistent
access from VM

Managed Disk

EBS

Ephemeral (short)
access from VM

Unmanaged Disk

Transactional
(ACID)

RDBMS

Complex
Relationships

Graph

Semi Structured
Data

Document

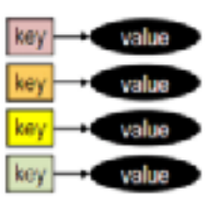
Simple
structure

Persistent

Key Value Store

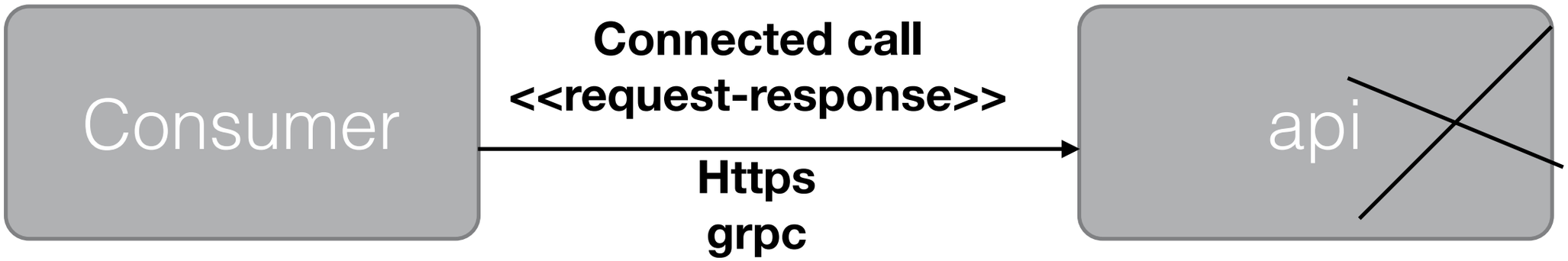
Transient

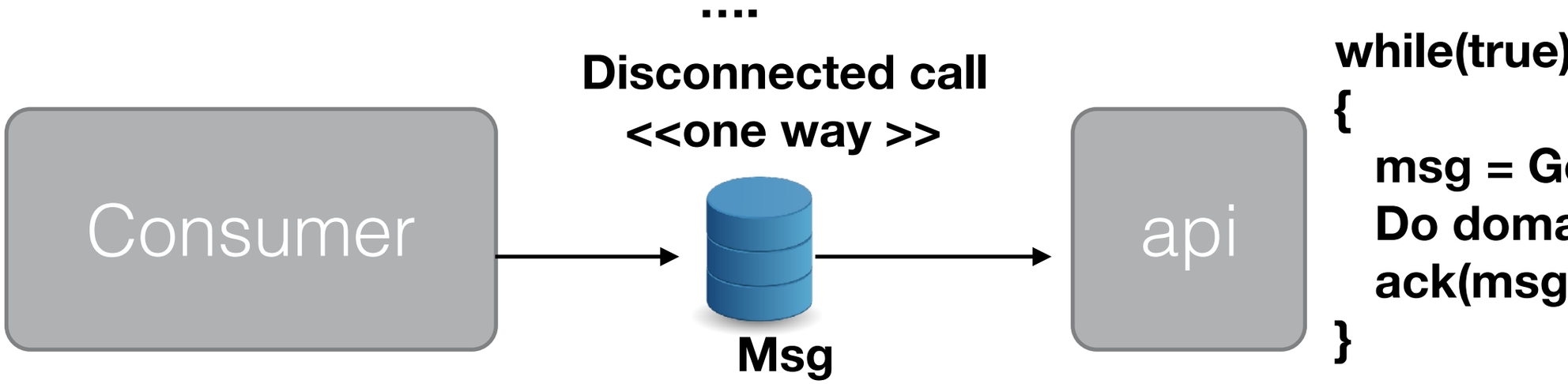
Key Value Cache

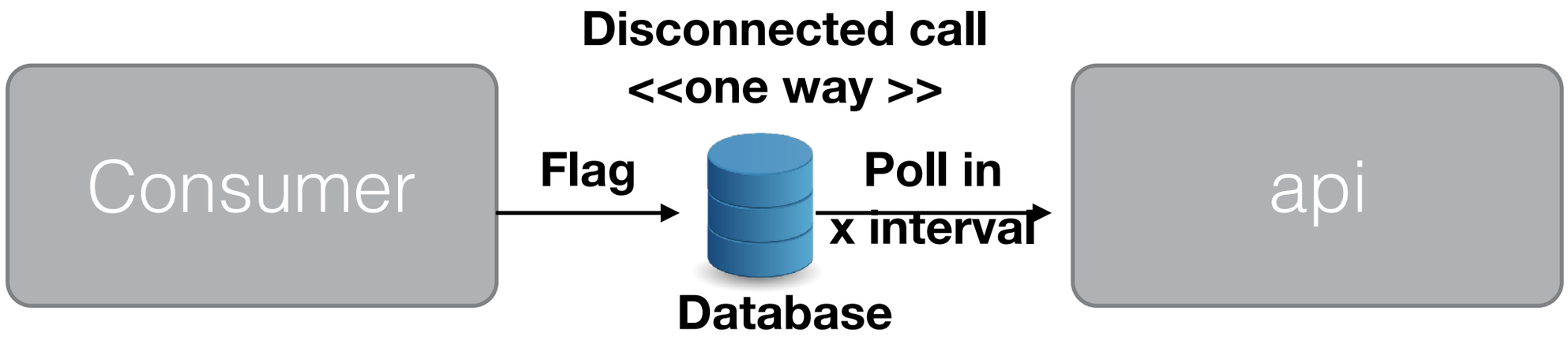


Logical view - todo
Application decomposition

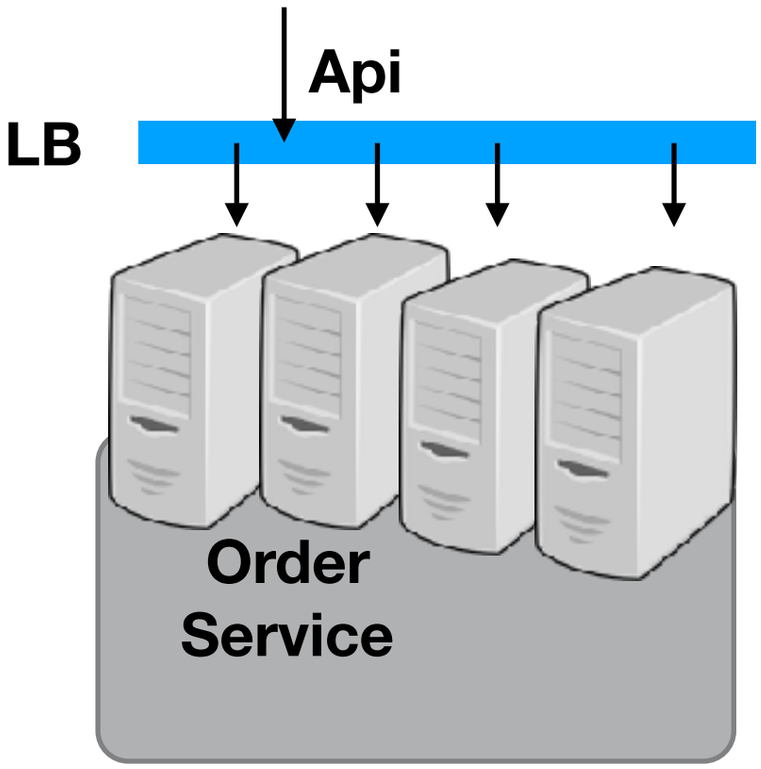
Logical view
Communication







API call



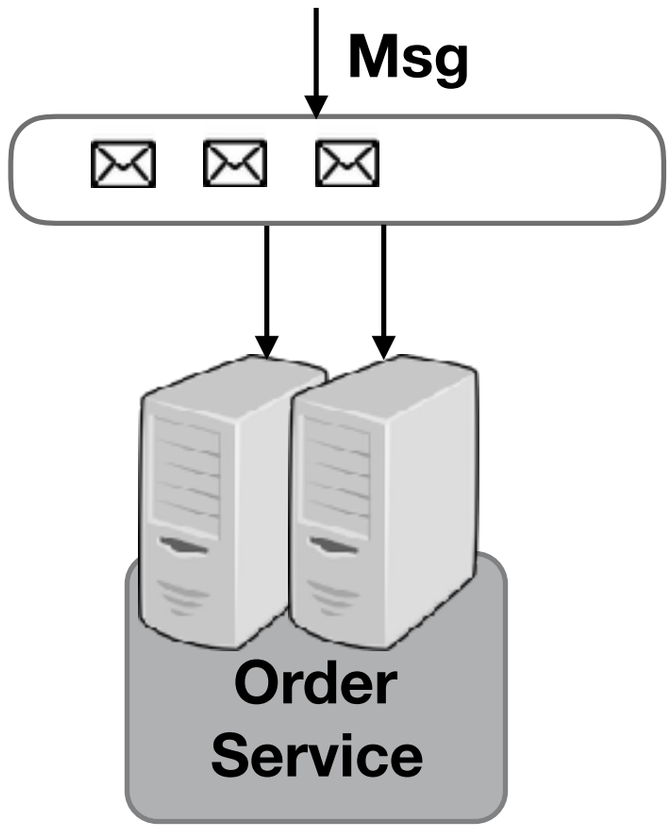
immediate Consistency

Less Scalable
no of server’s depends on peak load

less reliable
Will not remember last execution context
After recovery during a crash

loss of data
Because of throttle

Messaging



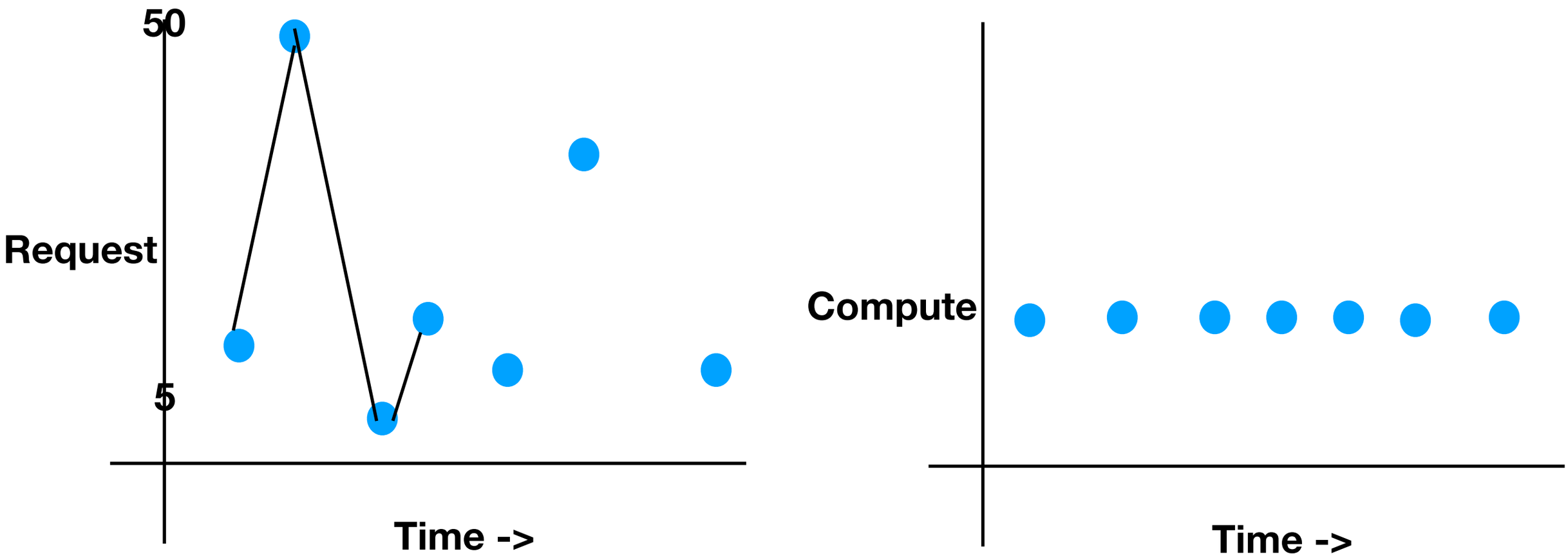
Eventual Consistency

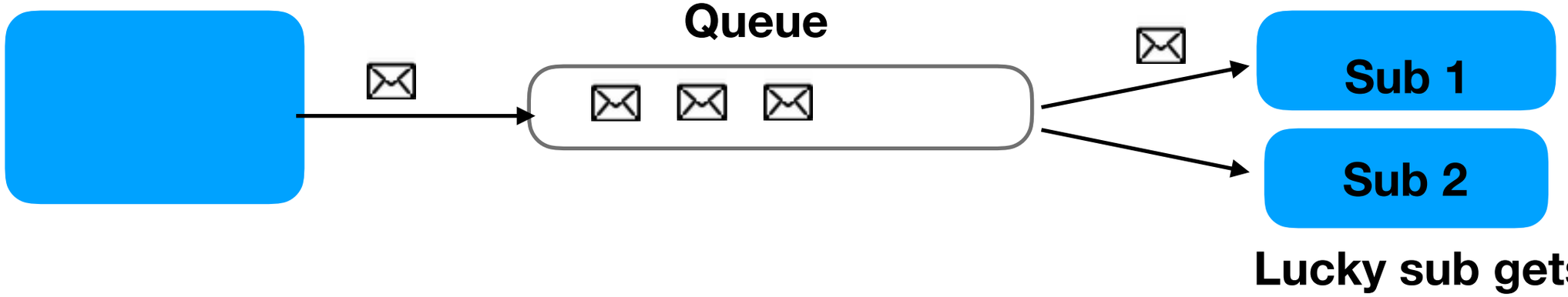
more scalable
no of server’s depends on Avg load
(Load Leveling)

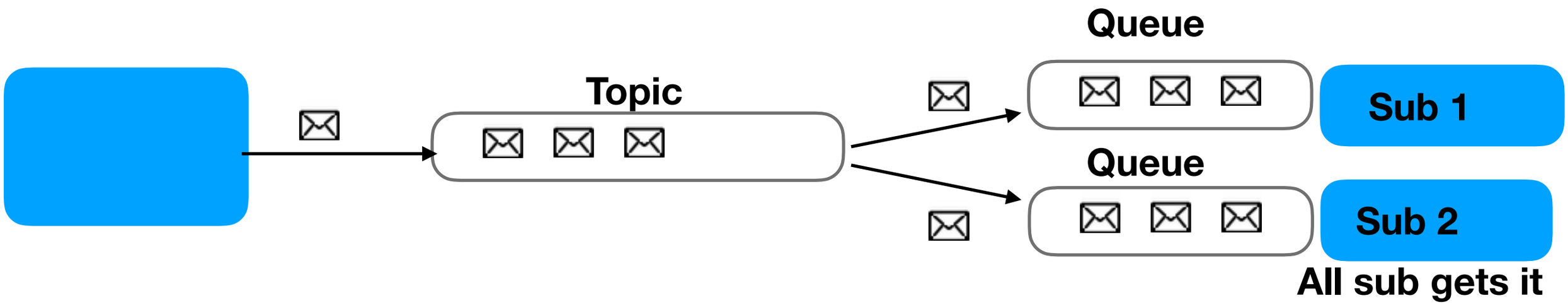
more reliable
If message are not ACK, messages
reappear in Queue

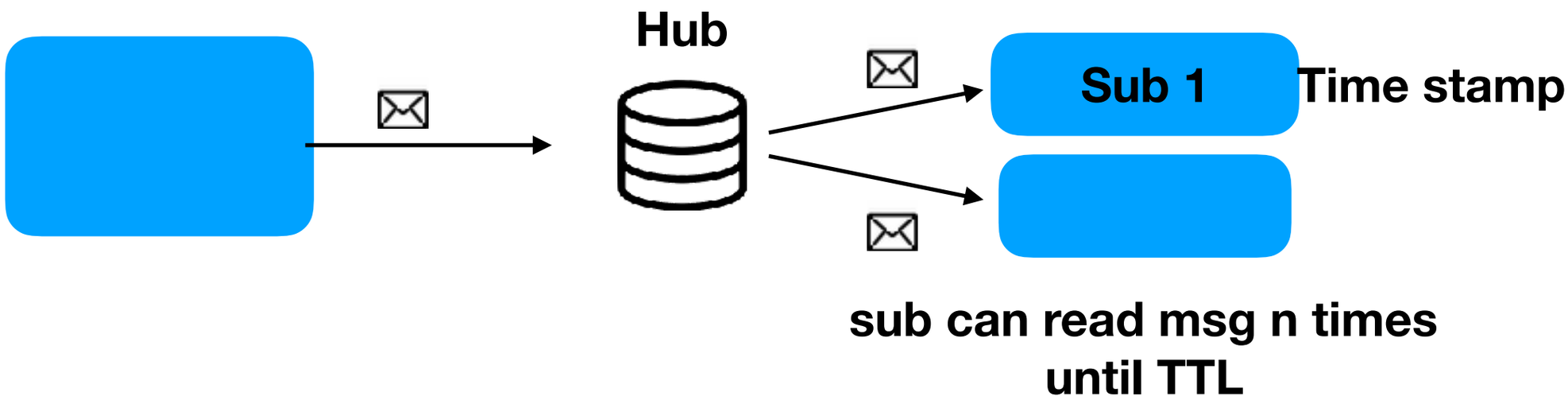
no loss of data
Gets queued

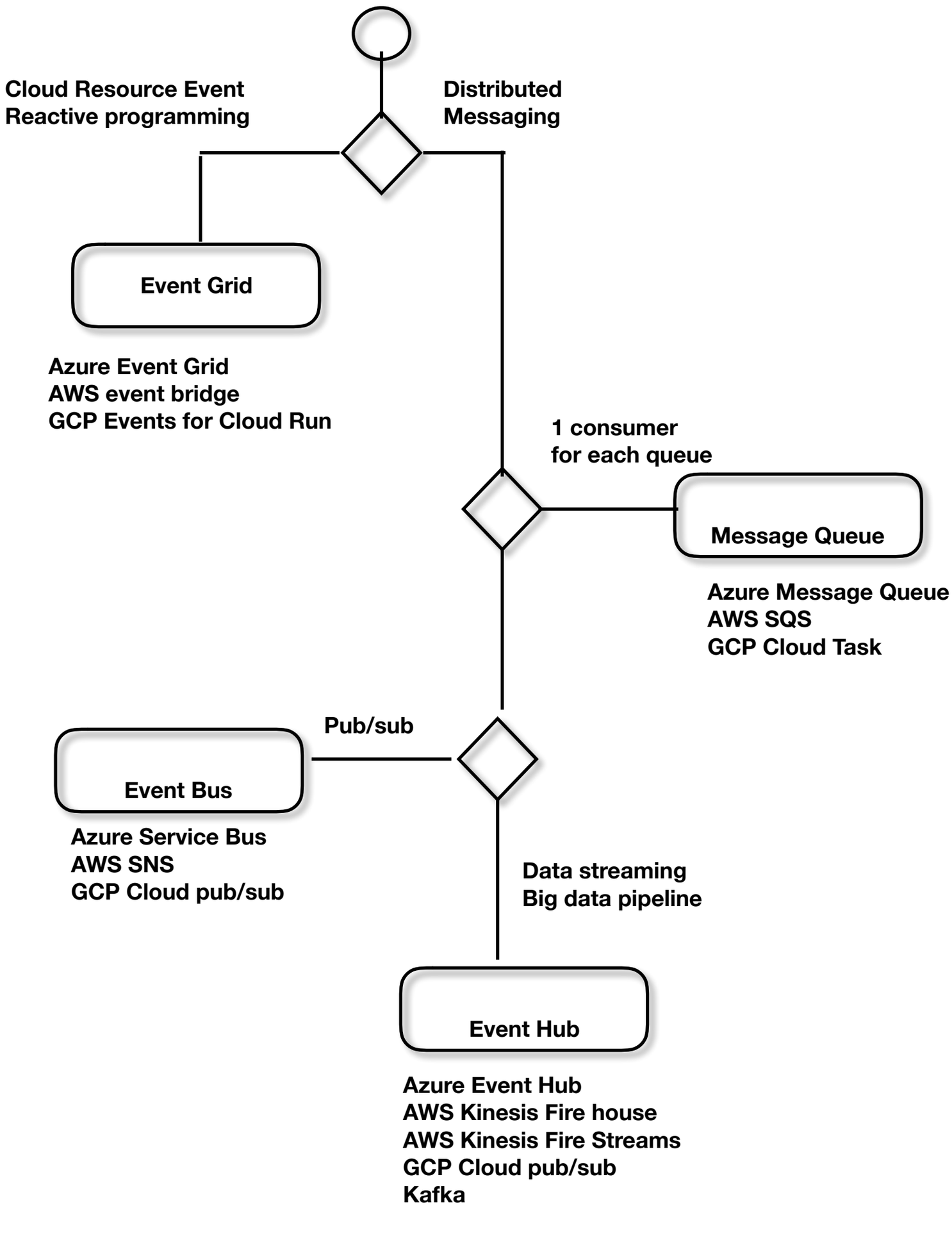
Load Leveling




Queue
Sub 1
Sub 2
Lucky sub gets it


Topic
Queue
Queue
Sub 1
Sub 2
All sub gets it


Hub
Sub 1
Time stamp
sub can read msg n times
until TTL


Cloud Resource Event
Reactive programming
Distributed
Messaging
Event Grid
Azure Event Grid
AWS event bridge
GCP Events for Cloud Run
1 consumer
for each queue
Message Queue
Azure Message Queue
AWS SQS
GCP Cloud Task
Pub/sub
Event Bus
Azure Service Bus
AWS SNS
GCP Cloud pub/sub
Data streaming
Big data pipeline
Event Hub
Azure Event Hub
AWS Kinesis Fire house
AWS Kinesis Fire Streams
GCP Cloud pub/sub
Kafka