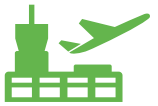
THOUGHTFOCUS

Architecting .NET Applications

- By Aniket Inge

Introduction / Course Agenda



Principles of architecture

Goals of architecture (*-ities)

Central theme of application architectures

Architecture types

Future

Q & A

About Me

Aniket Inge

Associate Technical Architect

Payments & Loyalty Solutions Group

Architecting applications in .NET, Java and JavaScript

THOUGHTFOCUS

Principles of architecture

Principles of architecture

- **Single Responsibility Principle (SRP)**

  Every component/module in the architecture of the system must exist for single reason only & should be as loosely coupled as possible with other components/modules of the system

- **Separation of Concerns (SOC)**

  It is a pattern of separating application into distinct sections such that, each section addresses a separate concern.

- **Open Close Principle (OCP)**

  The system should be “Open” for extension, but “Closed” for modification

- **Law of Demeter (LoD)**

  The components of the system should only have limited knowledge of other components & must talk only to it’s immediate “friends” using loosely coupled interfaces

- **Don’t Repeat Yourself (DRY)**

  No two components in the system should have similar responsibilities/reason for existence

- **You Ain’t Gonna Need It (YAGNI)**

  Don’t add redundant components in the system. Avoid BDUF (Big Design Upfront)

- **Keep it Simple, Stupid (KISS)**

  The system as a whole should be easy to comprehend, reason about and as simple as possible

THOUGHTFOCUS

Goals of architecture (*-ities)

Goals of architecture

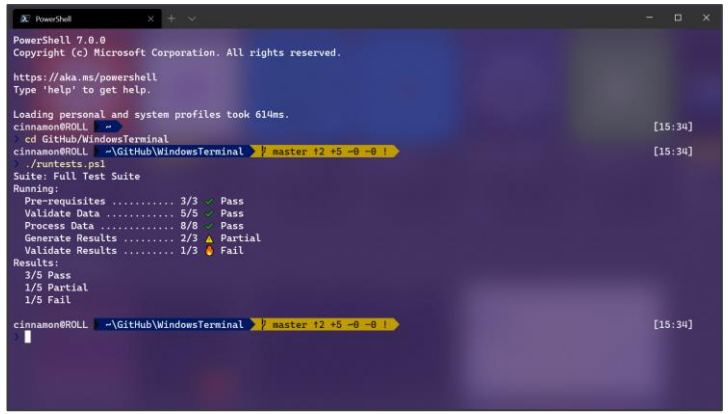
- Substitutability
- Scalability
- Evolvability
- Testability
- Availability
- Reliability
- Security
- Independent Deploy-ability (Upgradability)
- Performance
- Auditability

THOUGHTFOCUS

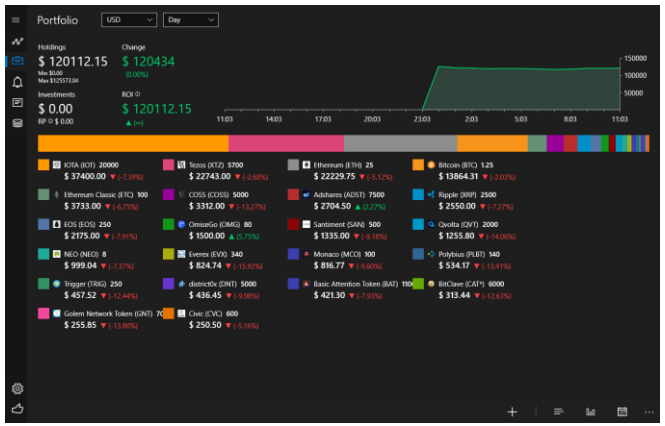
Central theme of application architecture

Central theme / Application Types

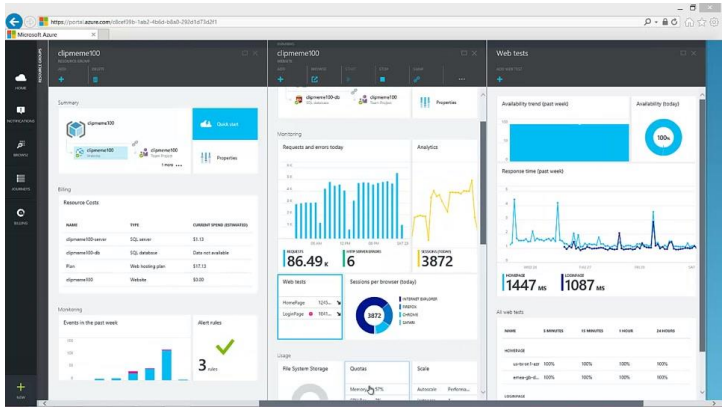
Terminal Applications



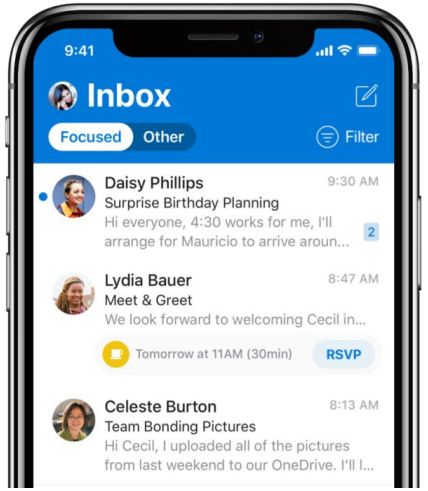
Desktop Applications



Web Applications

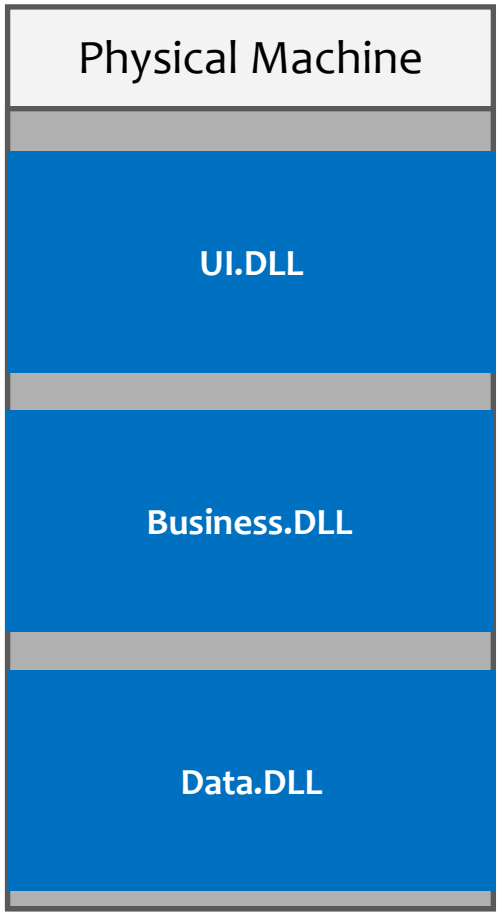


Mobile Applications

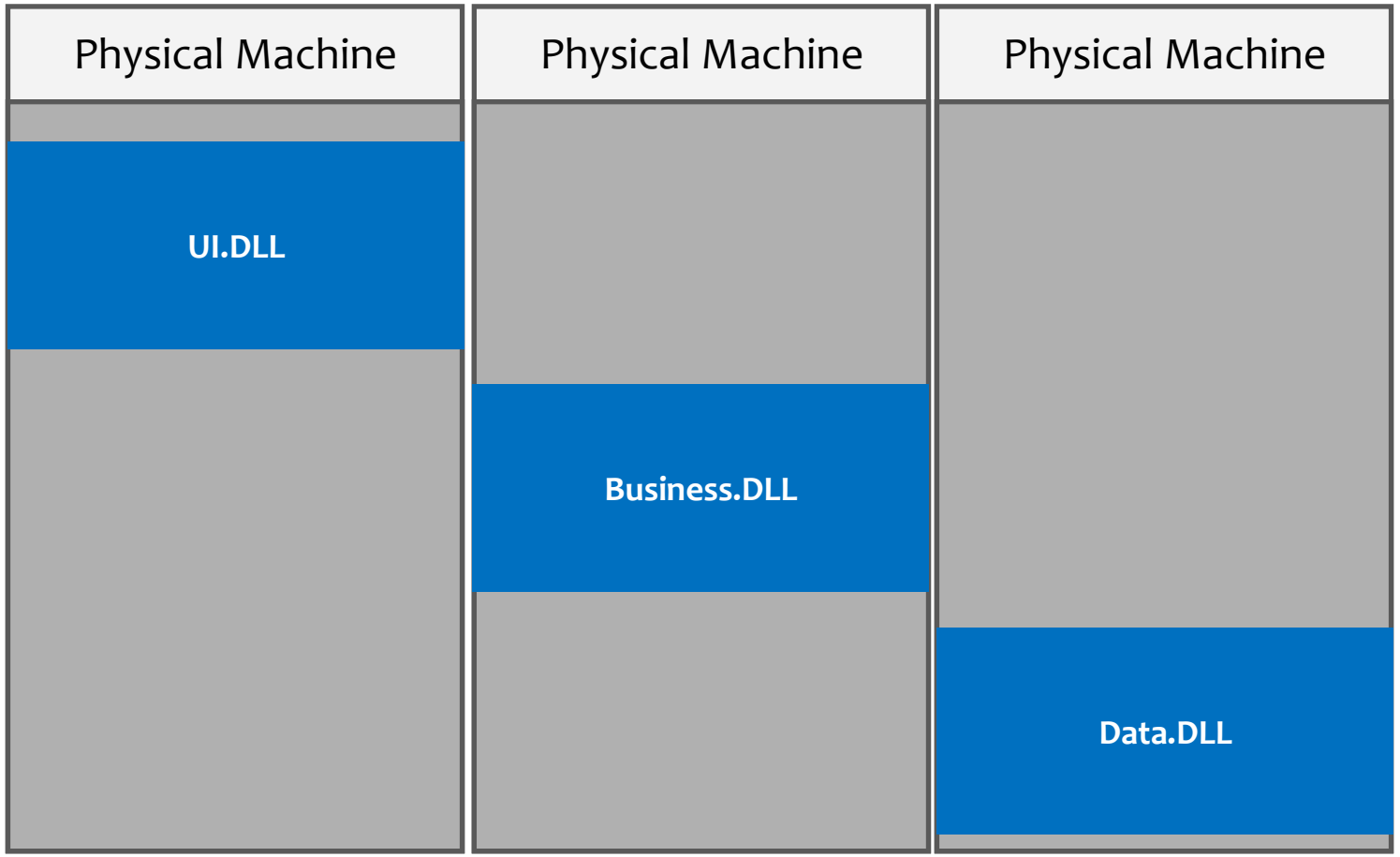


Central theme / Layers and Tiers

1 Tier, 3 Layer Architecture

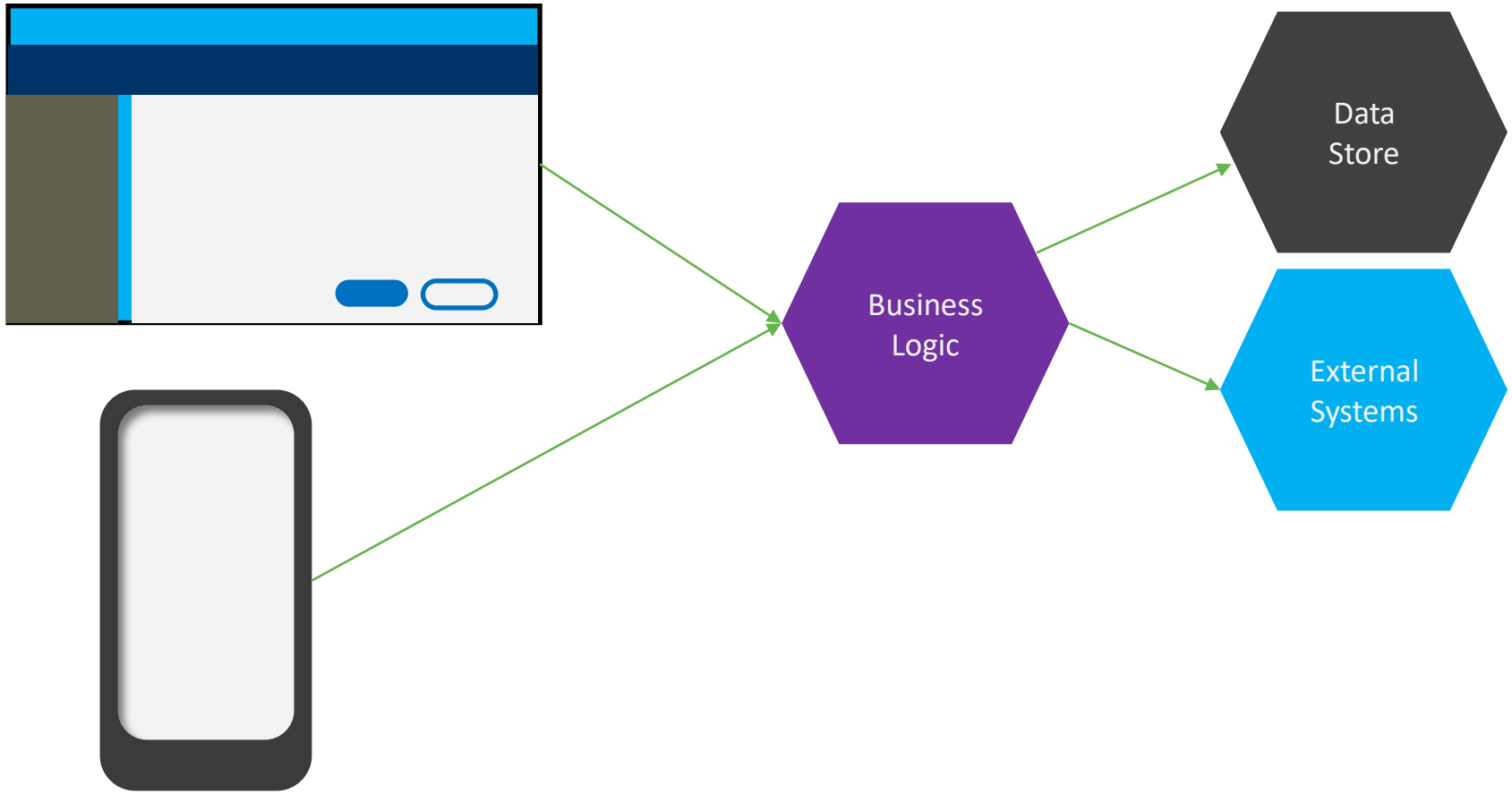


3 Tier, 3 Layer Architecture



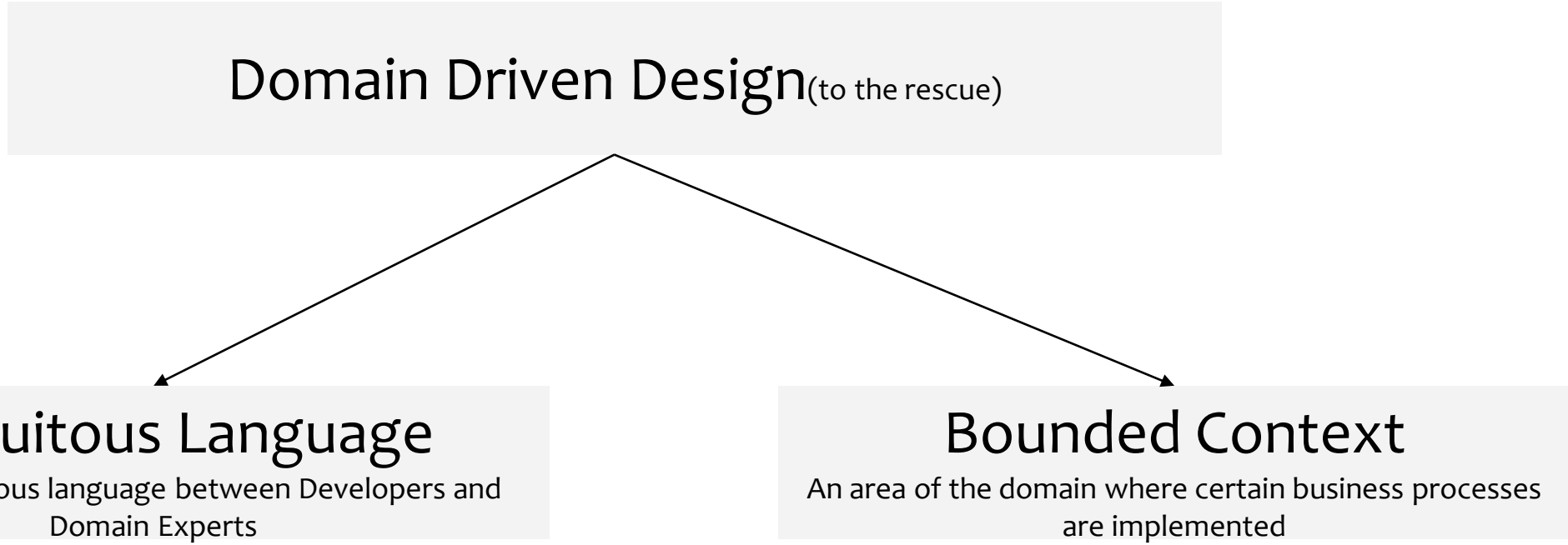
Central theme / User Interface

User Interface is just an interface between User and the Business Logic



Central theme / Business Layer (or Tier)

- Business layer is purely code, and must remain testable and extensible
- Business layer is the “core” or the “brain” of the application
- Business layer interfaces with most of the external components (called the “Infrastructure”)
- Depending upon the application’s business domain, this layer can get complicated



THOUGHTFOCUS

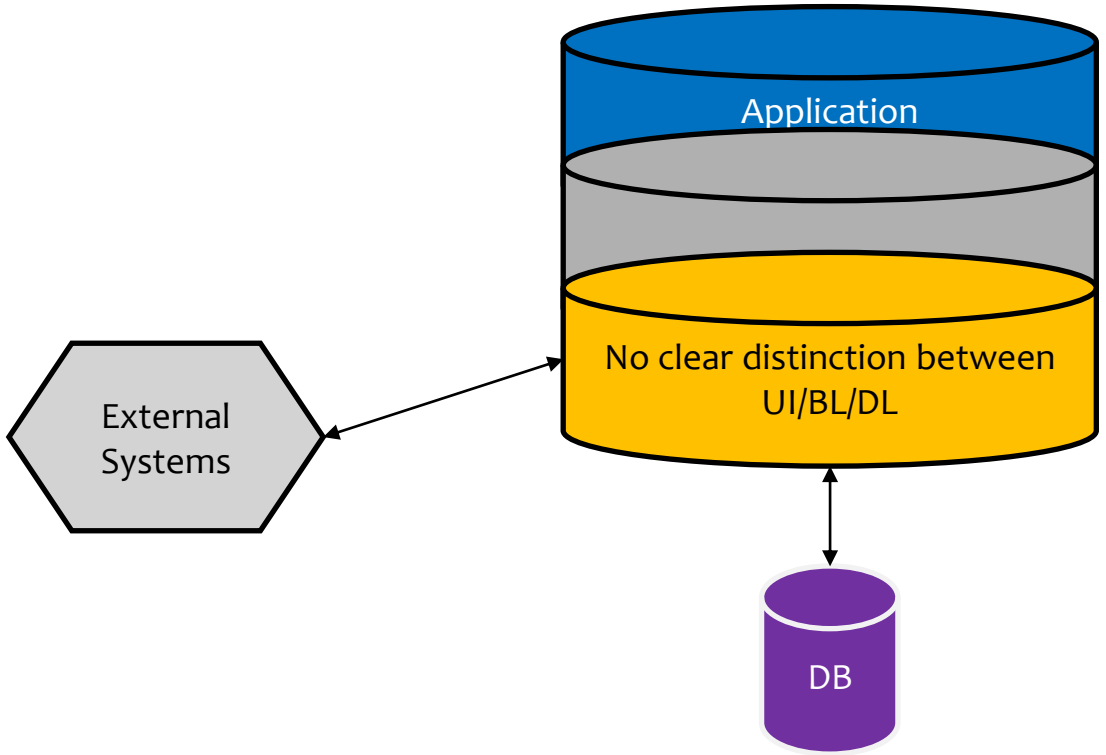
Architecture Types

Architecture Types / Index

- Monolith Architecture
- Client-Server Architecture
- Three-Layer/Tier Architecture
- N-Tier Architecture
- Microkernel Architecture
- Hexagonal Architecture
- Clean Architecture with DDD
- Event Driven Architecture
- Multi-App Architecture with Message Bus
- Multi-App Architecture with Pub-Sub
- Microservices Architecture

Architecture Types / Monolith

- Architecture where UI and Business Logic with Data Store are tightly coupled.
- Change to either one of the component will require complete re-deployment
- Requires rigorous unit testing, functional testing, systems testing, end-to-end testing before deployment
- High chances of breaking, if done incorrectly
- Typically waterfall model followed
- Low reusability and low substitutability
- Highly efficient (high performance), but not scalable horizontally [that is, not easily scalable across physical machines]

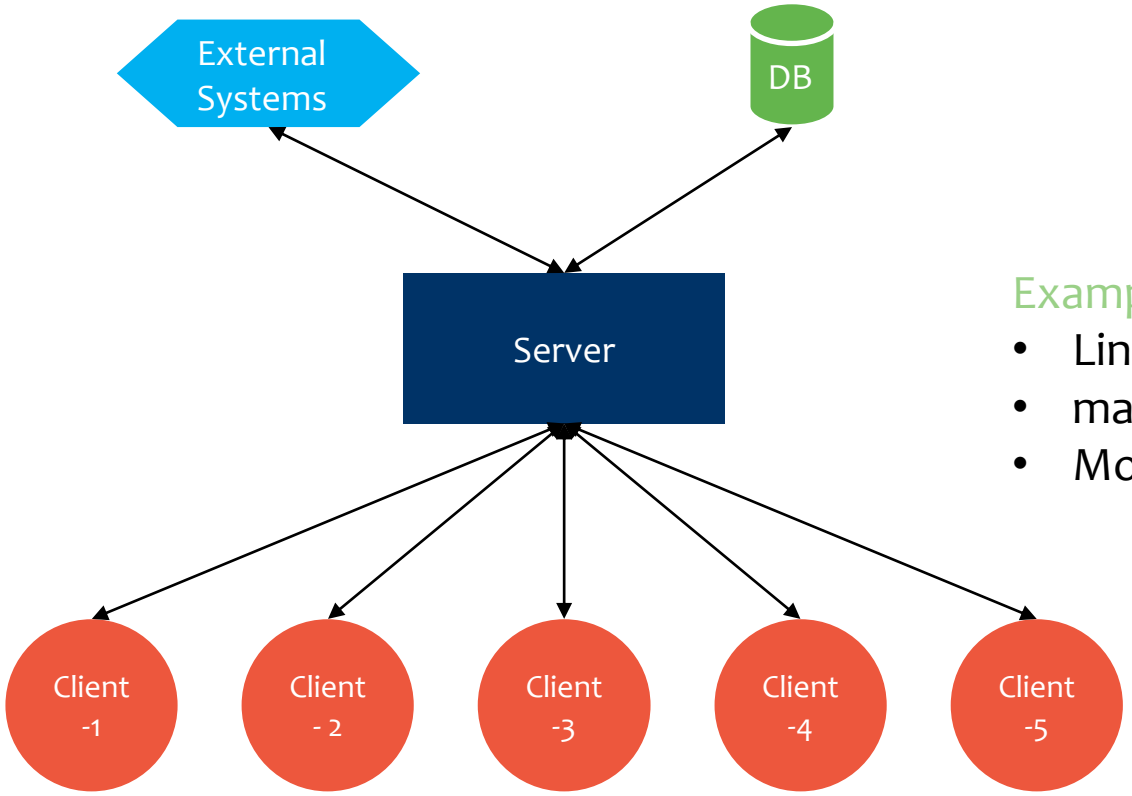


Example:

Most flight systems at NASA during 1970s-1990s … especially the Shuttle programs

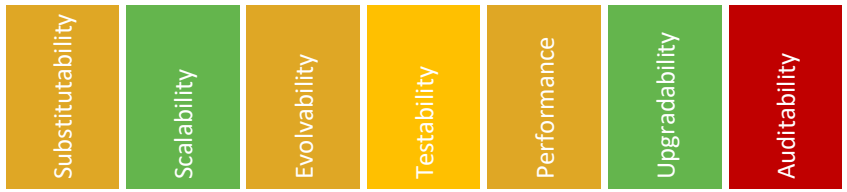
Architecture Types / Client-Server

- Two-Layer separation of concerns
- Highly scalable & efficient architecture
- Mostly achieved through OS sockets and networking, but can also be achieved via named-pipes & mailslots
- Improved testability and upgradability due to substitutability. (☺)



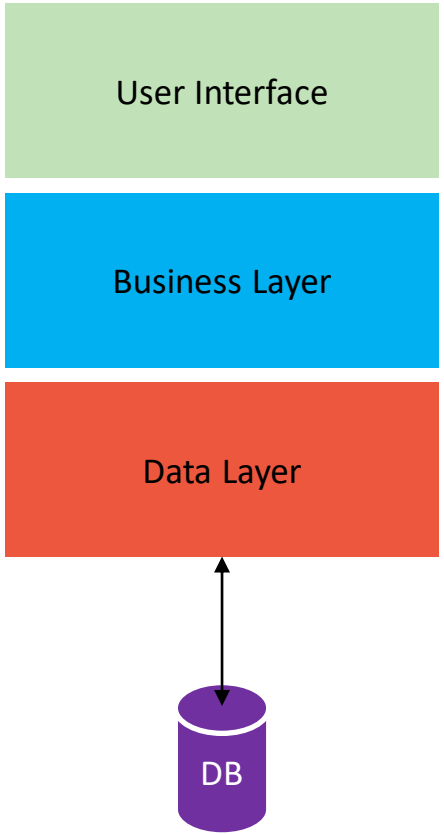
Examples:

- Linux's X.Org server, for X Windows System
- macOS's Quartz Display Engine
- Most MMORPG games



Architecture Types / Three Layer/Tier Architecture

- Enhanced separation of concerns
- User Interface Layer, Business Layer and Data Layer independently scalable
- Highly testable (but not 100% Unit-Test Automatable), maintainable and upgradable architecture, especially with Tiering
- Tiering typically achieved through WCF services

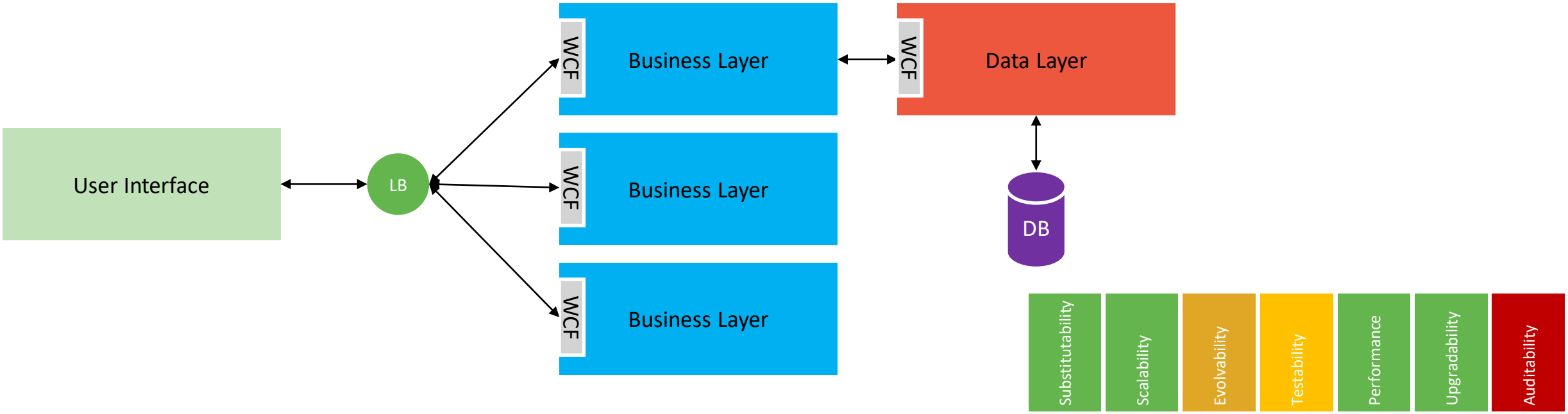


Examples:

- Most Web Applications & Desktop Applications
- Pre-Modern Mobile Applications

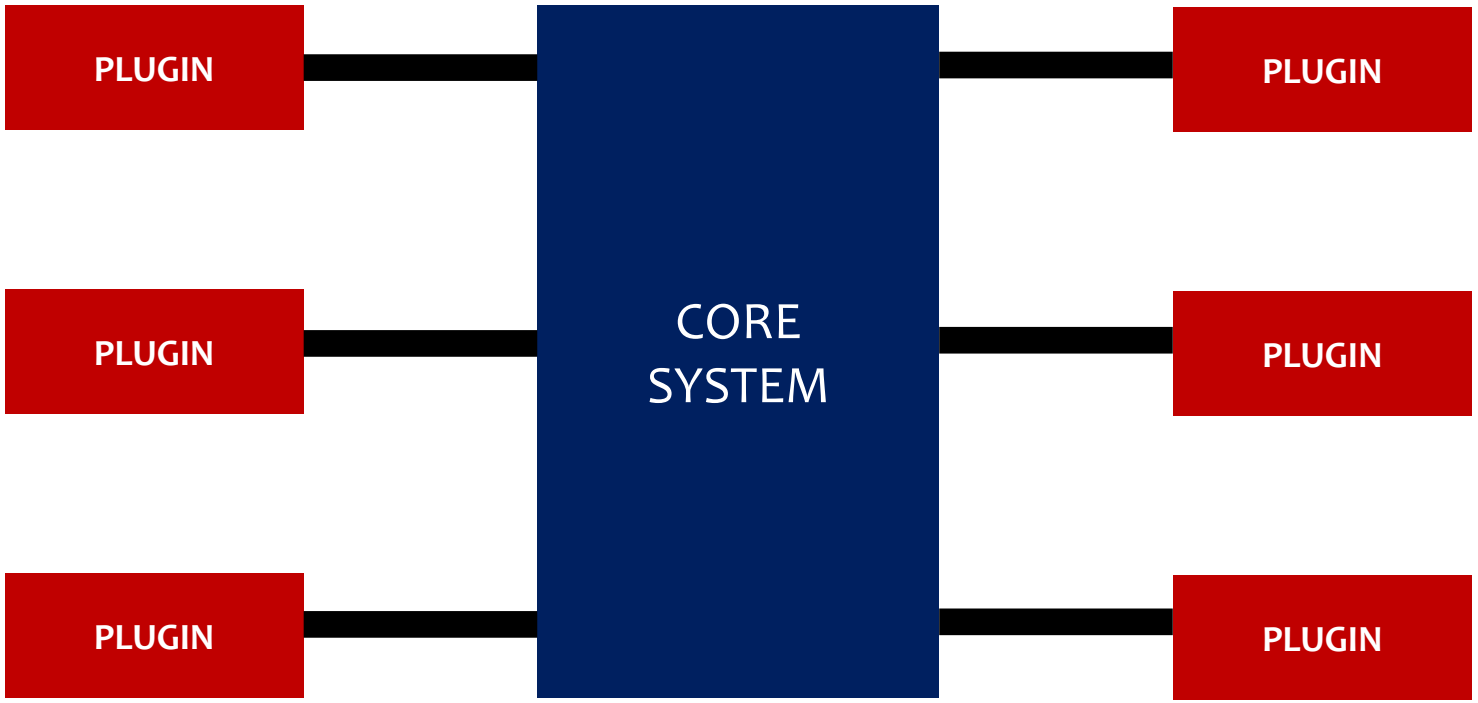
Architecture Types / N-Tier Architecture

- N-Tier Applications extend the concept of 3-Tier Architecture and allow vertical and horizontal scaling
- The multiple tiers are scaled horizontally using round-robin based resource managed load balancers
- Using Cloud technologies, these tiers can be auto-scaled vertically to handle peak loads
- Independent development possible, code for each tier can be written by separate team with focused technical knowledge
- For example, data tier can be managed and programmed by Data experts while UI and UX experts can simultaneously work on the UI tier



Architecture Types / Microkernel Architecture

- In Microkernel architecture, application logic is divided between two types of architecture components:
  - Core System
  - Plugin Modules
- Provides extensibility, flexibility and isolation of application features & custom processing
- Core system is usually the general business logic without custom code for special cases, special rules or complex conditional processing
- Core system can exist without the additional plugin modules, but the plugin modules can be hot-loaded without downtime

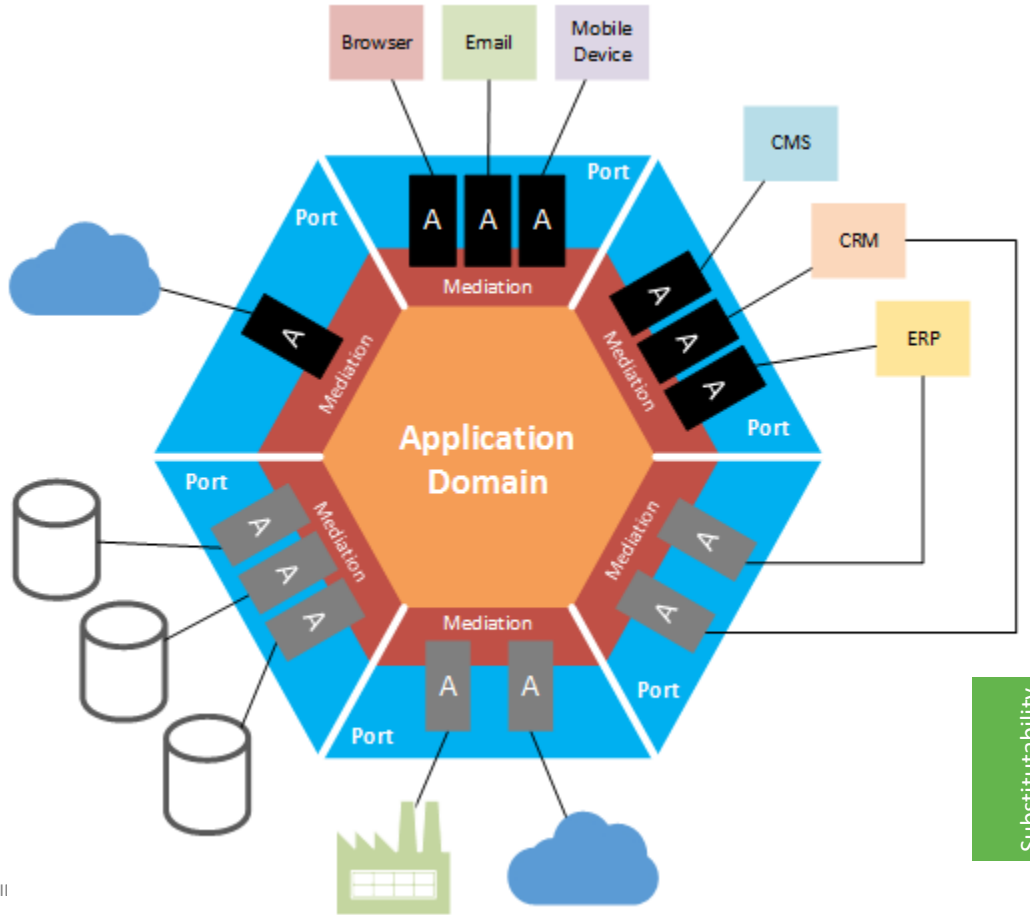


Examples:

- Most Operating System kernels
- Adobe Photoshop, Illustrator etc.
- Autodesk Maya, 3DSMax
- C# Roslyn Compiler
- Mozilla Firefox & Google Chrome
- Game Engines

Architecture Types / Hexagonal Architecture

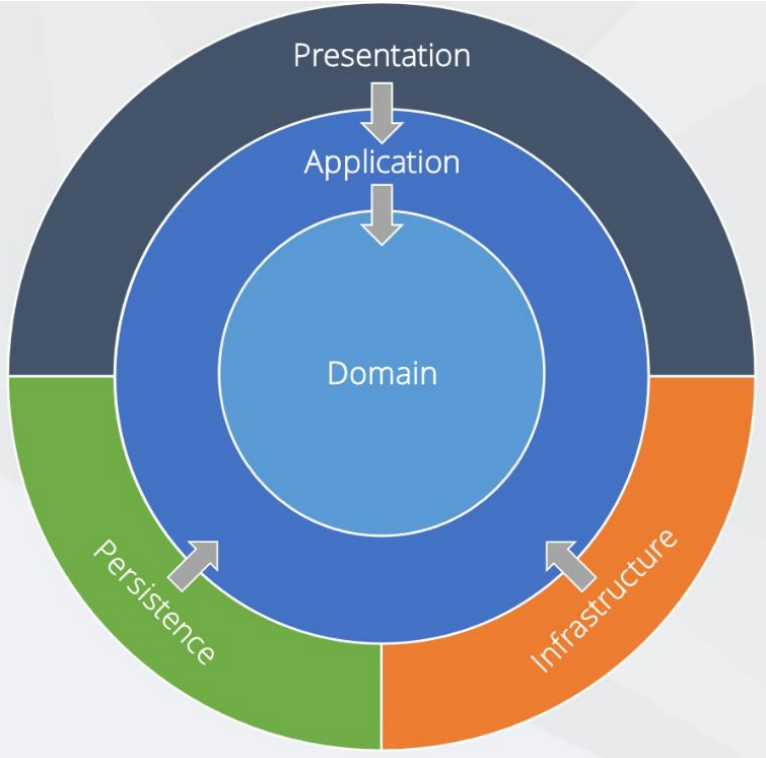
- Major goal of Hexagonal architecture is to separate the input and output boundaries from the core business logic of the application
- Improved testability as all the inputs to the business layer and the outputs from the business layer are isolated behind clean, well defined interfaces
- “Fakes” and “Mocks” can be used during development & unit testing – in case the production database is not up
- Development strategies such as TDD can be used – providing high quality code



Substitutability
Scalability
Evolvability
Testability
Performance
Upgradability
Auditability

Architecture Types / Clean Architecture with DDD

- Clean Architecture was proposed by Robert Cecil Martin
- It is independent of frameworks
- Independent of UI
- Independent of the database (does not even assume a database)
- All the dependencies direct inwards
- Highly testable

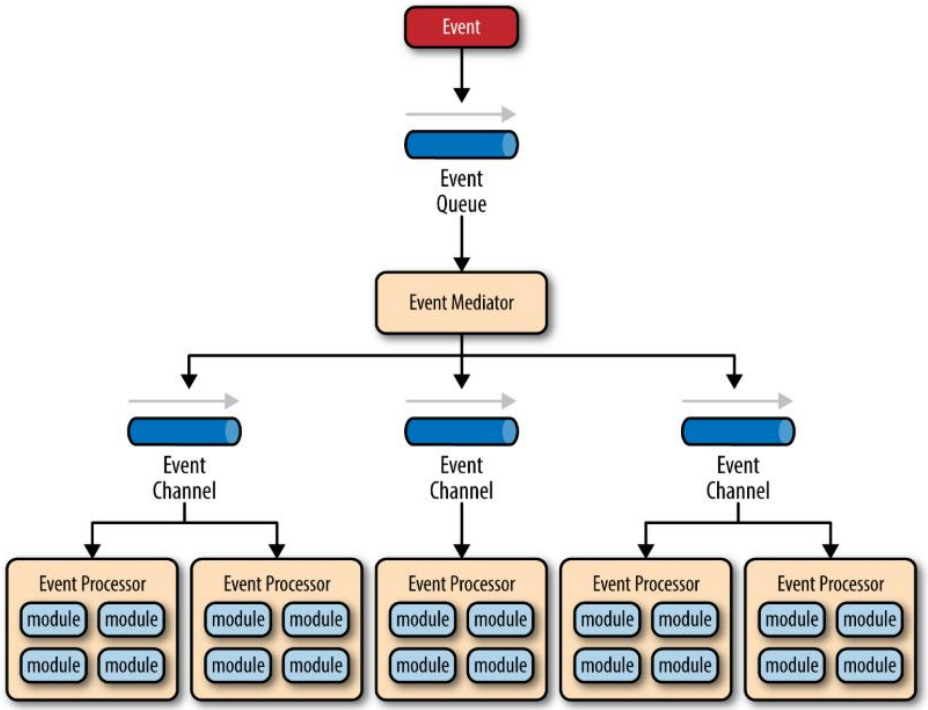


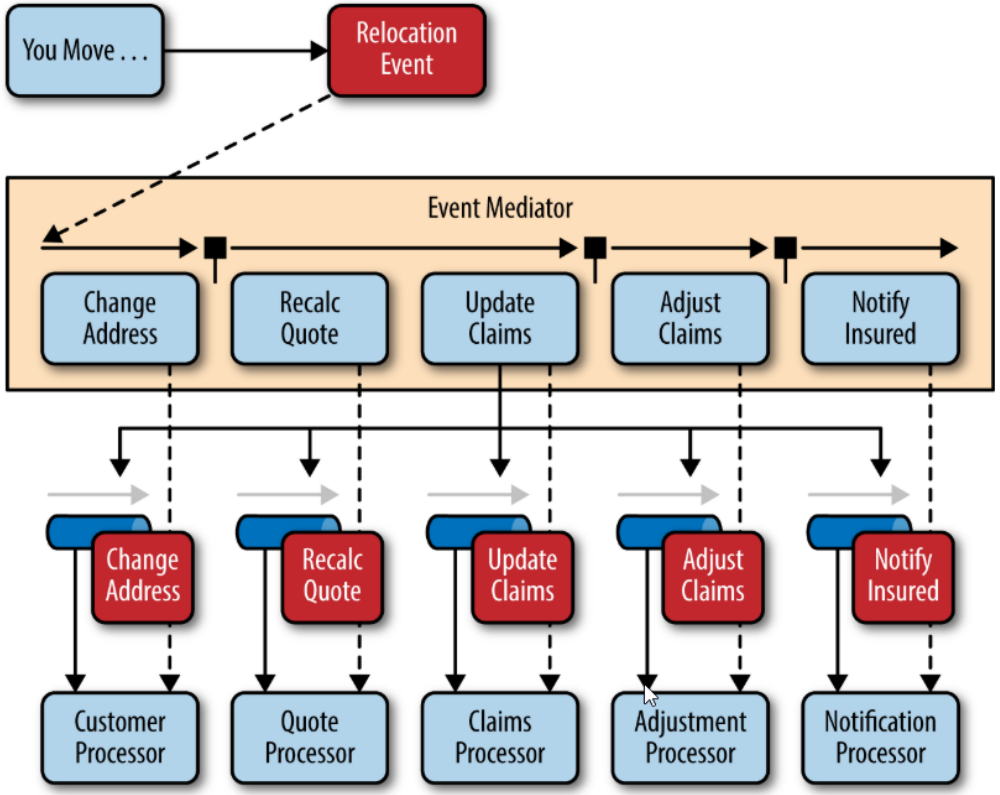
Examples:

- Fitness Test Suite

Architecture Types / Event Driven Architecture

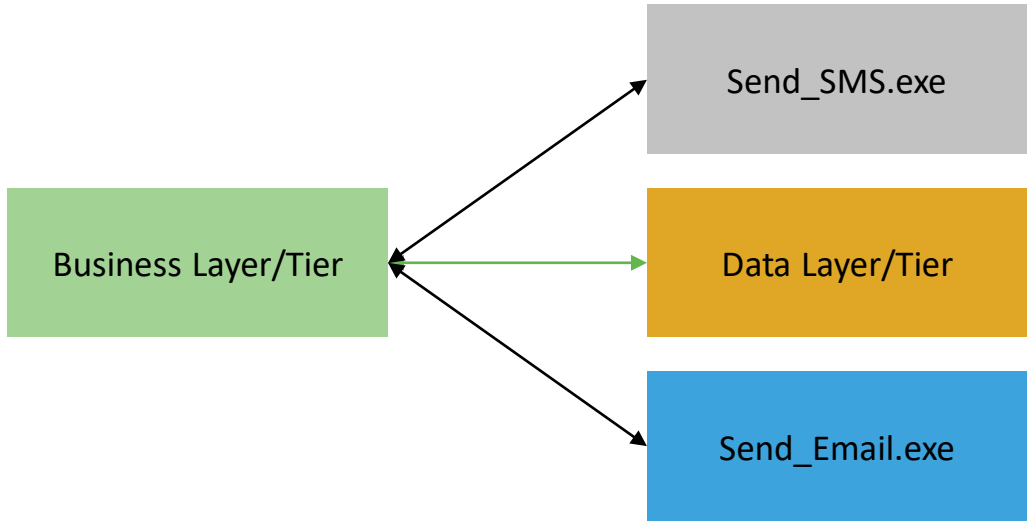
- Popular distributed asynchronous architecture pattern used to produce highly scalable applications
- Can be used for small applications as well as large, complex ones
- The architecture is made of highly decoupled, single-purpose event processing components that receive and process events
- Consists of two major topologies: “Mediator” and “Broker” making it highly **Auditable**
- Mediator topology is used when you want to orchestrate multiple steps for a single event through a single mediator
- Broker topology is used when you want to chain events together without the use of a single central mediator





Architecture Types / Multi-App Architecture

- Multiple single-purpose applications are used to orchestrate a business flow
- E.g. A business process registers an order in the database, sends invoice to the user over email using a separate process connected via some form of communication medium (PubSub or Message Bus), sends a SMS to the user confirming the order in another process that is also connected via the communication medium
- The independent processes can scale, can reside on different machines with different process-address spaces and if the independent processes crash, they do not take the entire system down
- Allows for asynchronous processing that does not clog the business flow
- Inherently improves performance(responsiveness) of the system

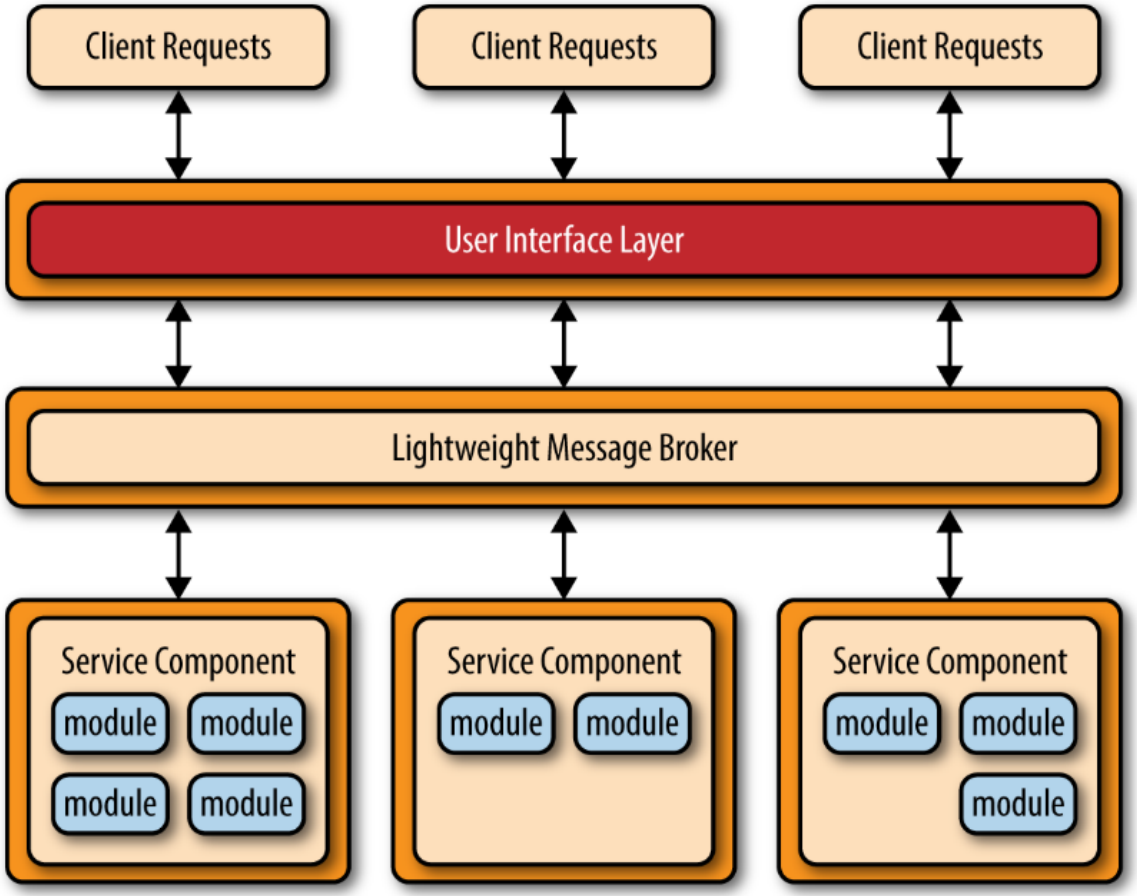


Examples:

- Most E-Commerce Applications
- TF's Beyond Measurements IoT System

Architecture Types / Microservices Architecture

- Microservices are independently deployed, self-contained RESTful Services that divide the application on the concept of "Bounded Context".
- Usually 1 Microservice for 1 bounded context
- Highly scalable, but not inherently performant – due to distributed nature of the services. These can be made performant



Examples:

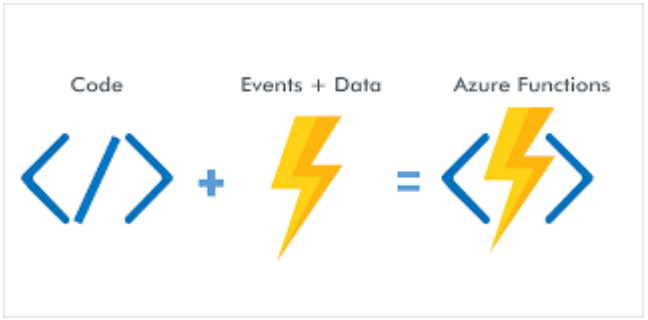
- NetFlix
- Amazon PrimeVideo
- Hotstar

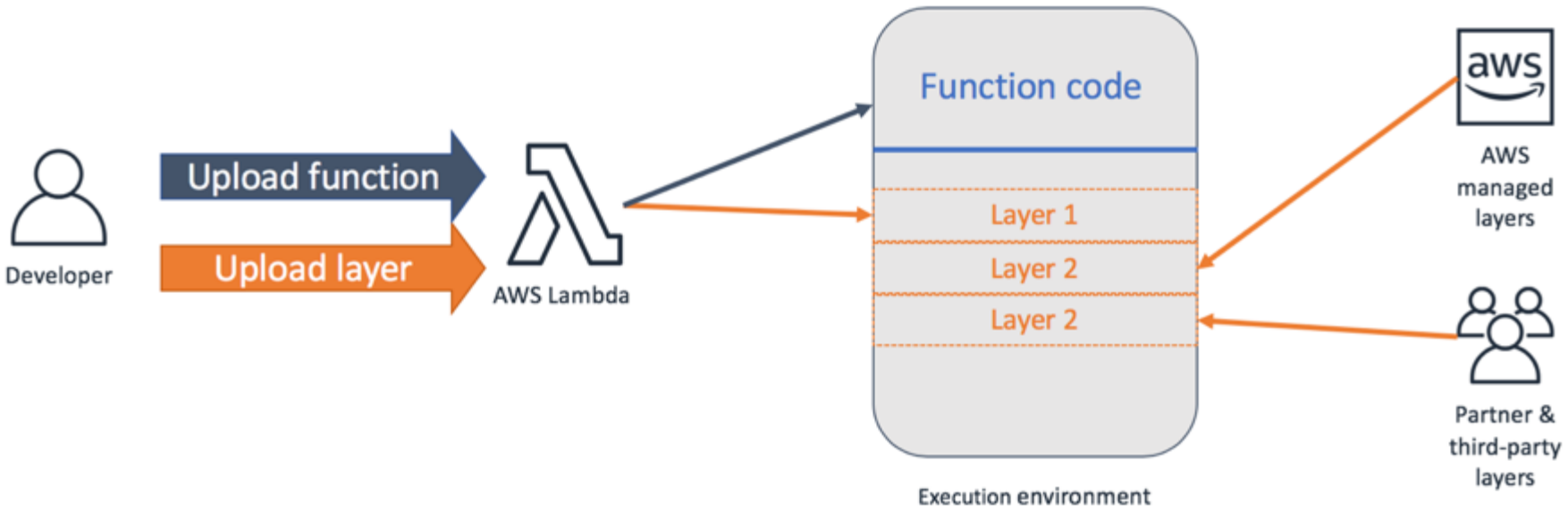
THOUGHTFOCUS

Future

Future

- Two major architectures coming up are “Stream Processing” architectures and “Serverless” architectures
- In Stream Processing architecture, data flows in a stream – business components co-ordinate and react to the data flow, modifying the data as it flows
- In Serverless architectures, applications are hosted by a third-party service, eliminating the need for a server-software, hardware management by the developer
- Application can be broken into individual functions that can be invoked and scaled individually.
- Top two “Serverless” Architecture Providers (Also called Function as a Service (FaaS)) are AWS Lambda and Azure Functions
- FaaS typically use HTTP based API end-points(RESTful) to invoke the functions, and produce output





THOUGHTFOCUS

Q & A?

THANK YOU

Aniket Inge

Associate Technical Architect

+91 99012 26007

aniket.inge@thoughtfocus.com

About ThoughtFocus

ThoughtFocus is a privately held technology consulting and services company serving middle market to large enterprise clients in Professional Services, Manufacturing, Financial Services, Higher Education and Aerospace. Clients look to ThoughtFocus for innovative solutions in product engineering, knowledge process outsourcing, and digital transformation. The company has over 1,900 employees globally and is one of the fastest growing technology services companies. ThoughtFocus is a technology partner and portfolio investment company of Blackstone, a leading private equity firm.