# Microkernel Architecture

Software Architecture

Richard Thomas

March 4, 2024

So far...

Simplicity – Monolith, Pipeline Modularlity – Layered, Pipeline

# Definition 1. Extensibility

Features or extensions can be easily added to the software over its lifespan.

How easy is it to extend *Monolith*, *Layered* or *Pipeline*?

How easy is it to extend *Monolith*, *Layered* or *Pipeline*?

Answer

Monolith – Everything in one container



How easy is it to extend *Monolith*, *Layered* or Pipeline?

Answer

Monolith – Everything in one container Layered – Typically all layers







How easy is it to extend *Monolith*, *Layered* or *Pipeline*?

### Answer

Monolith – Everything in one container Layered – Typically all layers Pipeline – Create a new filter







# Definition 2. Interoperability

Software can easily share information and exchange data with internal components and other systems.

What about interoperability?

What about interoperability?

Answer

Monolith – Everything in one container

• Internal 🥮 External 🤏

What about interoperability?

### Answer

Monolith – Everything in one container



• Internal 🧼 External 👄



What about interoperability?

### Answer

Monolith – Everything in one container



Layered – Nearest Neighbour





Pipeline – Standard Interface



What if I want simplicity, extensibility and interoperability?

What if I want simplicity, extensibility and interoperability?

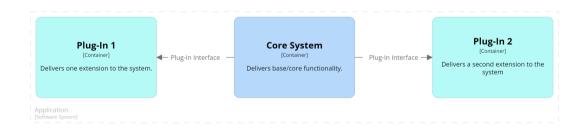
Answer

Consider Microkernel Architecture

# $\S\ Microkernel\ Architecture$

### Definition 3. Microkernel Architecture

Core system providing interfaces that allow plug-ins to extend its functionality.



# Definition 4. Registry

and how to access them.

Tracks which plug-ins are available to the core system

Loading Plug-ins

Static Loading when application starts

Dynamic Loading as needed at run-time

Registry designed for the selected strategy

Can you think of a *microkernel architecture*?

Can you think of a *microkernel architecture*?

Answer

Web Browser?

# Definition 5. Independent Plug-in Principle

Plug-ins should be independent, with no dependencies on other plug-ins. The only dependency on the core system is through the plug-in interface.

# Definition 6. Standard Interface Principle

There should be a single interface that defines how the core system uses plug-ins.

Does a plug-in architecture equate to a microkernel architecture?

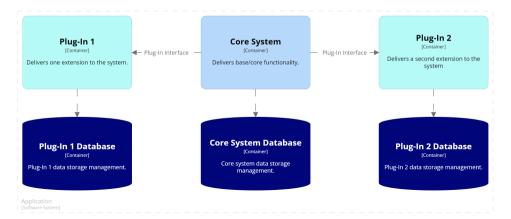
Does a plug-in architecture equate to a microkernel architecture?

Answer

What about *IntelliJ*?

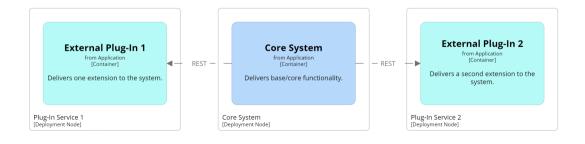
# Plug-ins with Separate Databases

- Plug-ins cannot access core system data
  - Core system may pass data to the plug-in
- Plug-ins may have their own persistent data

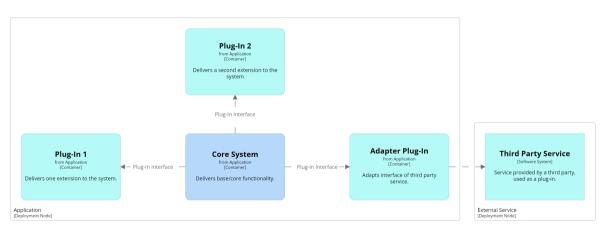


# Plug-ins as External Services

- Need communication protocol
- Registry records communication contract
  - e.g. URL of the REST endpoint & data passed to it



# Adapting Non-Conforming Interfaces

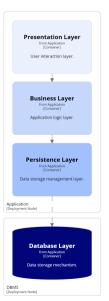


# § Technical & Domain Partitioning

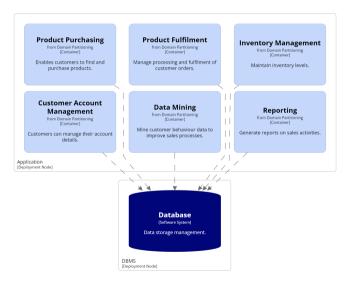
# Technical Partitioning



# Technical Partitioning



# Domain Partitioning



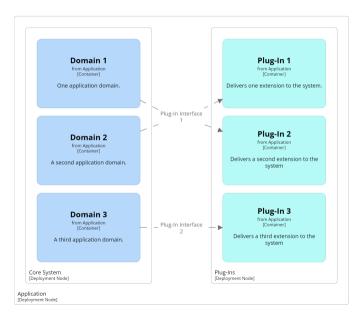
Is the microkernel architecture suited to technical or domain partitioning?

Is the microkernel architecture suited to technical or domain partitioning?

### Answer

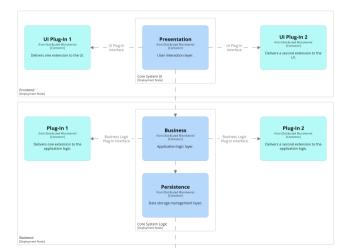
Core system can be partitioned either way.

# Domain Standard Interfaces



# Distributed Microkernel

- Partitions in the core system can be distributed
  - Technical or domain partitions
  - Plug-ins could also be distributed



# $\S\ Media\ Server \ Example$

What types of systems could use a *microkernel* architecture?

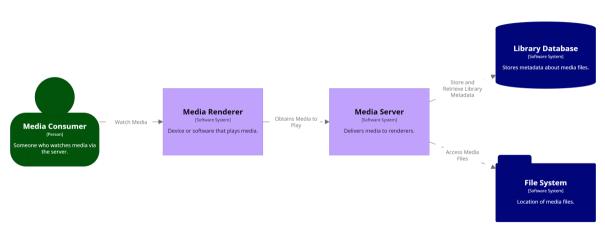
### Question

What types of systems could use a *microkernel* architecture?

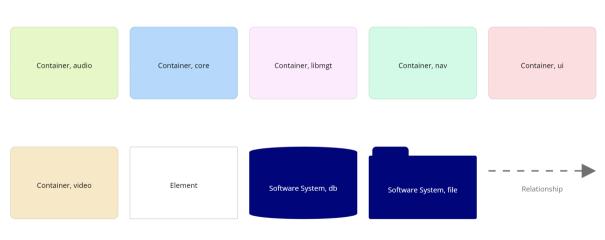
#### Answer

- Social media aggregator
- IoT management & processing
- Media server

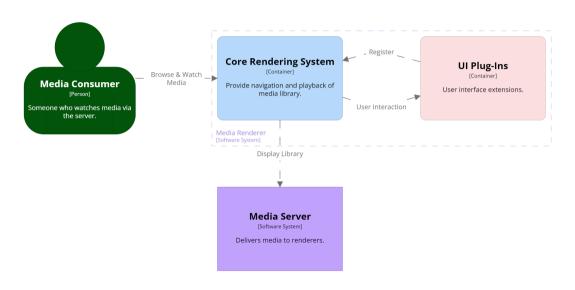
## Media Server & Renderer



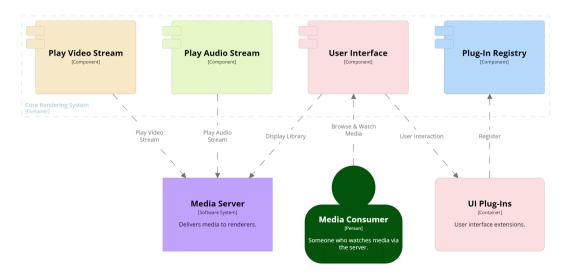
# Domain Colour Key



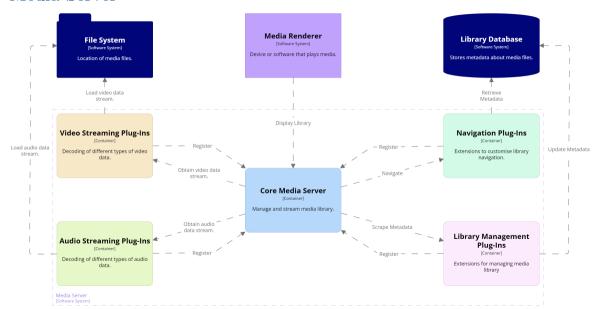
#### Media Renderer



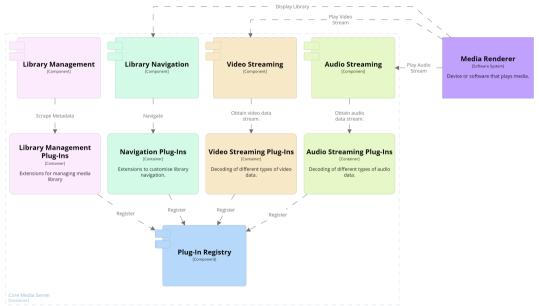
# Core Rendering System Components



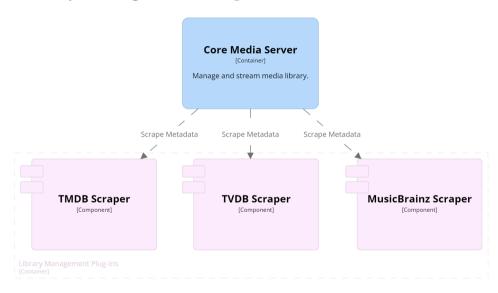
#### Media Server



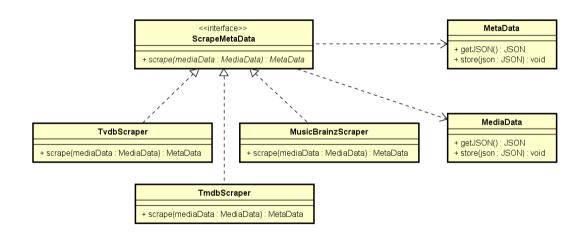
## Core Media Server Components



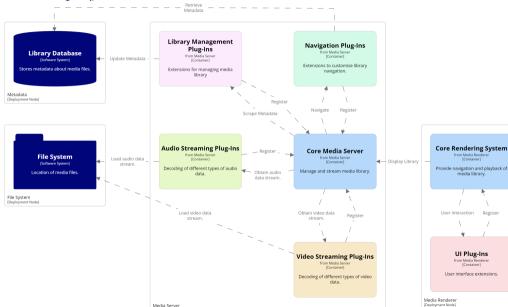
## Media Library Management Components



## Scrape Metadata Interface



# System Deployment



# § Conclusion

# Microkernel Pros & Cons

Simplicity Core system & Plug-in interface Extensibility Plug-ins Interoperability Plug-ins Scalability Reliability