Application Integration Styles

An overview of integration methods for enterprise applications:

- Shared Database
- Remote Procedure Invocation (RPC)
- File Transfer Integration
- Integration via Message Brokers (review)

Understanding strengths, weaknesses, and suitable scenarios for each integration style.

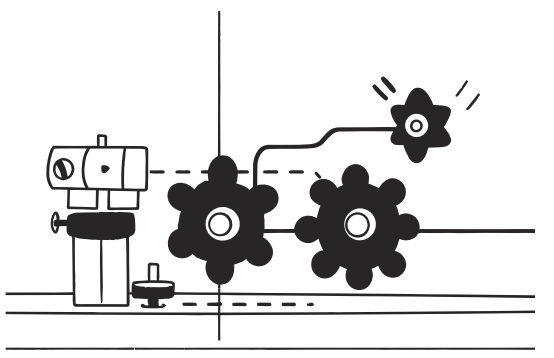
1. Shared Database Integration

Definition:

- Applications integrate via direct access to a single shared database.
- Data consistency achieved through common storage.

Typical Use Case:

- ERP and CRM systems sharing customer data directly.



How Shared Database Integration Works

Mechanics:

- Applications directly query a shared database.
- Shared schema ensures unified data structures.
- Transactions ensure consistency.

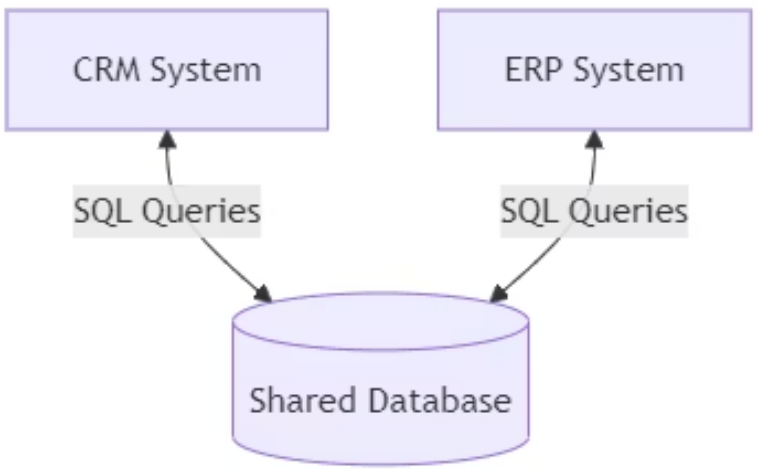
Pros:

- Immediate data consistency
- Simple and direct integration

Cons:

- Tight coupling
- Schema dependency
- Performance bottlenecks due to database contention

Shared Database Integration – Diagram



2. Remote Procedure Invocation (RPC)

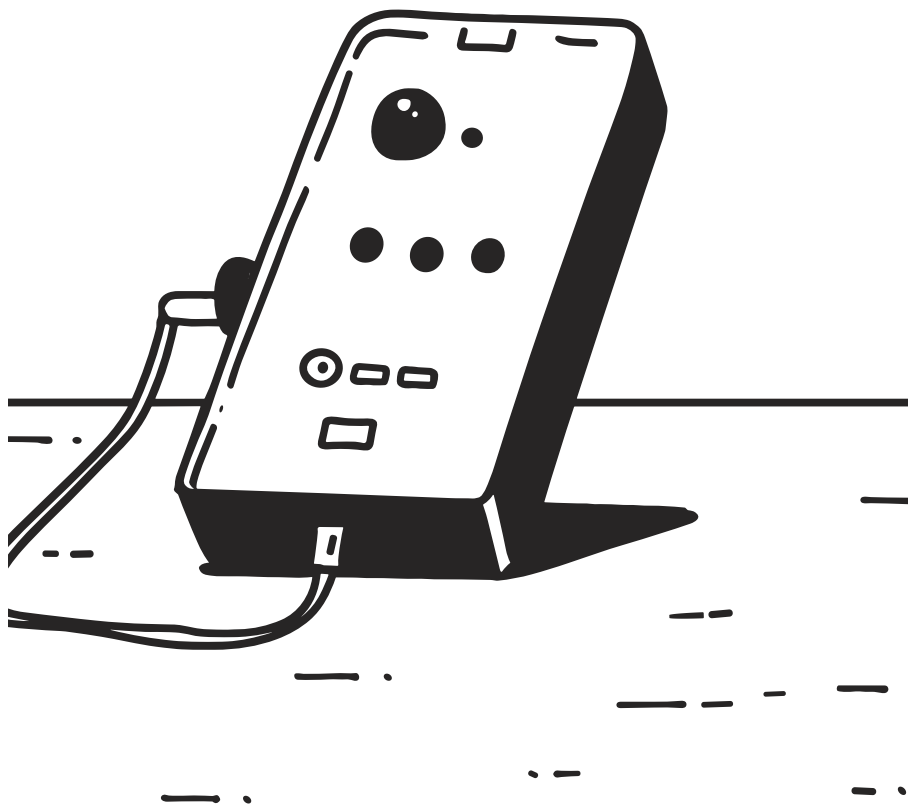
Definition:

Applications invoke services or procedures directly from remote systems.

Typically synchronous and request-response oriented.

Common Protocols:

REST, SOAP, gRPC



How RPC Integration Works

Client Request

Client application sends a request to a remote server

Server Processing

Server processes request and returns immediate response

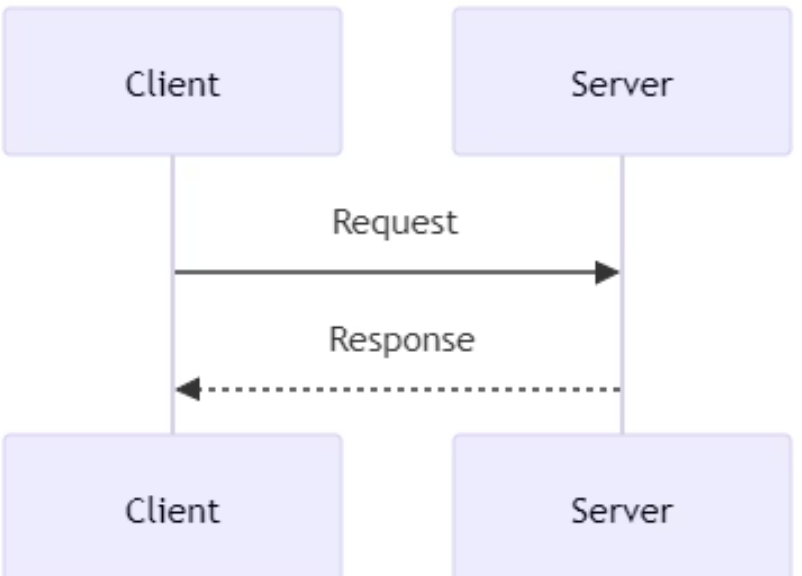
Communication Channel

Communication usually over HTTP/TCP

Pros: Simple request-response model, Easy to implement and debug

Cons: Tight coupling, Latency sensitivity, Reduced scalability under high load

RPC Integration – Diagram



3. File Transfer Integration

Definition

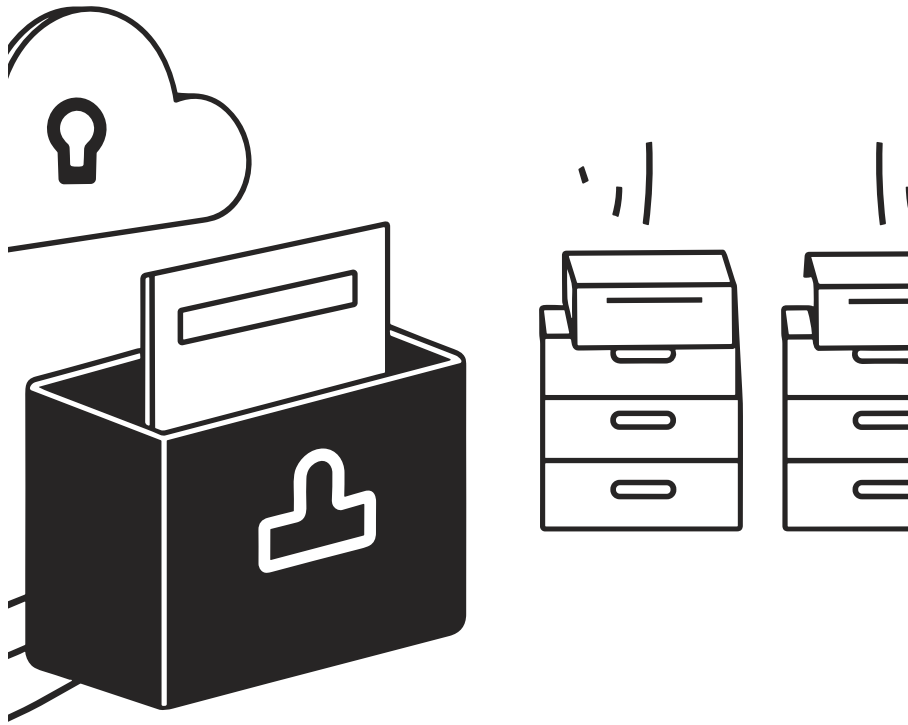
Asynchronous integration using file exchange (CSV, JSON, XML).

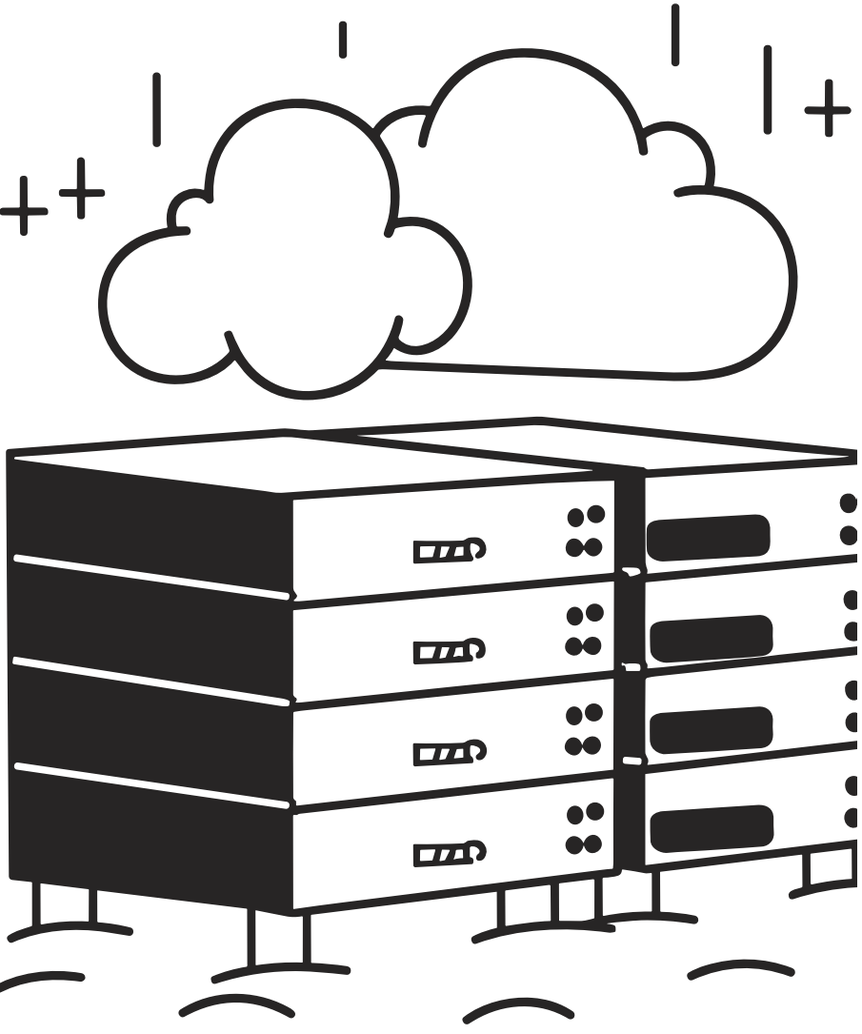
Usage

Frequently used for batch operations or scheduled synchronization.

Example

Daily batch synchronization between sales and inventory systems.





How File Transfer Integration Works

Export

Producing application exports files regularly

Storage

Files stored on shared storage (FTP, Cloud storage)

Import

Consuming application imports and processes files asynchronously

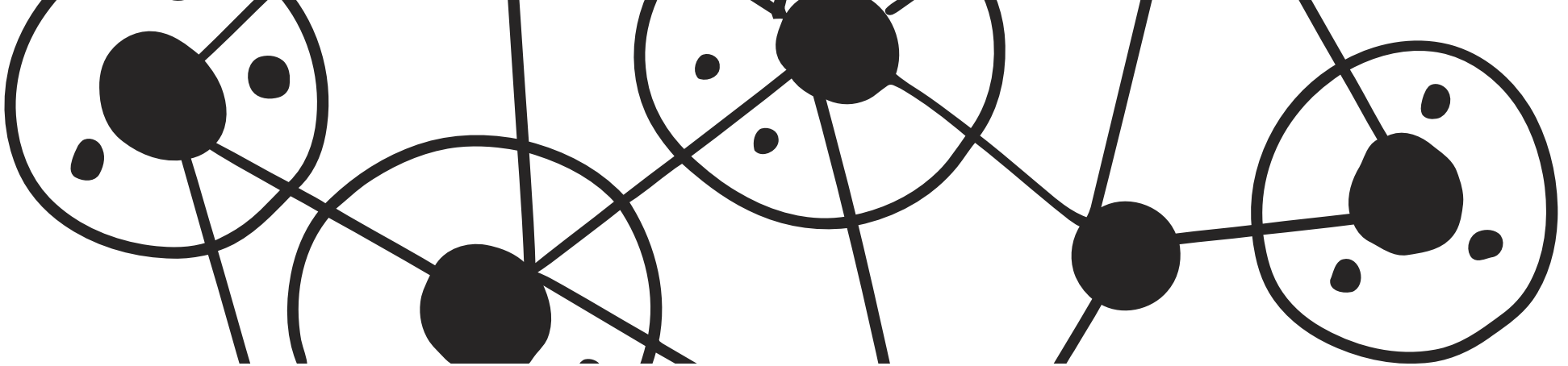
Repeat

Process repeats according to schedule

Pros: Loose coupling, Handles large volumes of data easily, Simple implementation

Cons: Delay between data production and consumption, Complex error management, Data freshness concerns

File Transfer Integration – Diagram



4. Integration via Message Brokers (Review)

Definition

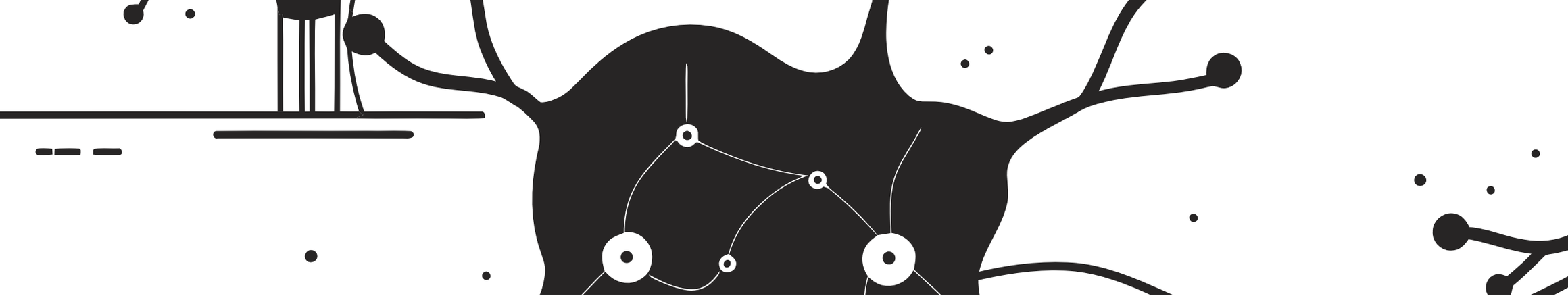
Applications communicate asynchronously through intermediary message brokers

Function

Broker manages message queuing and distribution

Scenario

High-volume transaction systems such as order processing



How Message Broker Integration Works

1

Producer

Producer publishes messages to broker.

Broker

Broker queues and distributes messages to subscribers.

Consumer

Consumers asynchronously receive and process messages.

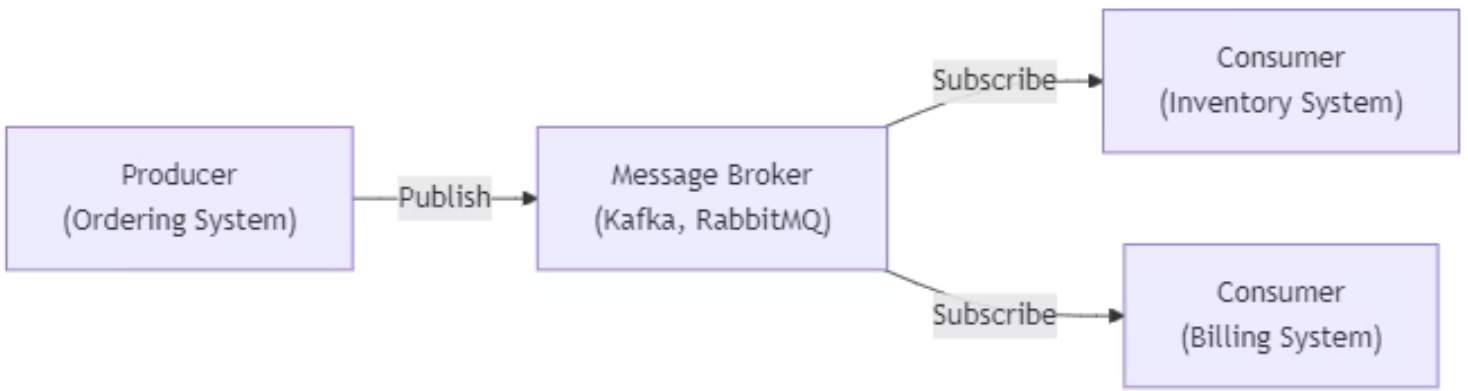
Pros

- Scalability and loose coupling
- Reliability through guaranteed delivery
- Fault-tolerance

Cons

- Complexity of managing broker infrastructure
- Potential latency in processing
- Debugging challenges

Message Broker Integration – Diagram



Choosing an Integration Style

Integration styles should match your specific enterprise scenario.

Real-time vs. batch

Consider your timing requirements

Coupling

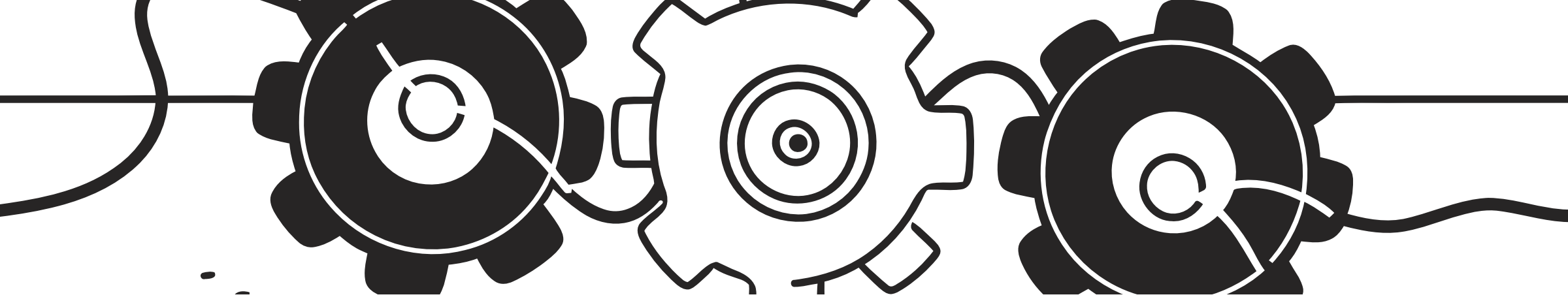
Tight vs. loose coupling

Complexity

Complexity vs. simplicity

Performance

Performance and scalability requirements



Integration Styles Comparison

	Shared DB	RPC	File Transfer	Message Broker
Coupling	Tight	Tight	Loose	Loose
Communication	Synchronous	Synchronous	Asynchronous	Asynchronous
Latency	Low	Low/Medium	High	Medium
Complexity	Low	Medium	Low/Medium	Medium/High
Scalability	Low	Medium	Medium	High