Actor-Based Programming

Fundamental Concepts
- Actors as fundamental units
- Message Passing
- Concurrency
- Encapsulation
- Mailboxes for message queues
- Creating and destroying actors
- Error handling and supervision
- Location transparency
- State management
- Fair scheduling

Advanced Concepts
- Distributed actor systems
- Fault tolerance
- State persistence and snapshots
- Actor lifecycle management
- Advanced messaging patterns
- Dynamic system reconfiguration
- Security in actor systems
- Custom schedulers and dispatchers
- Testing and debugging
- Reactive streams and Back-Pressure

Future Trends and Evolving Aspects
- Application in IoT and Edge Computing
- Use in Large-Scale Distributed Systems

Use Cases
- Concurrent processes in Erlang
- Distributed systems with Akka Framework

Interconnections and Influences
- Relationship with concurrent and distributed programming
- Influence on system design and scalability
- Used in Real-Time and responsive systems

Technical Proficiency and Best Practices
- Design patterns for actors
- Performance tuning in Actor-Based systems
- Distributed deployment
- Secure communication
- Monitoring and telemetry
- Error handling and recovery
- Versioning and compatibility
- Integration with other systems
- Scalability strategies
- Community and standards