

Part 2: Distributed Node-Based Logical Architecture

[Go to The FX Demo Root](#)



Financial Systems Archetype

Foreign Exchange (FX) Demo

Abstract

Part 2 defines the platform-independent logical architecture for the Financial Systems Archetype. It describes the logical nodes, runtime planes, information flows, responsibilities, and separation-of-concerns rules that apply before any specific technology stack is introduced into the architecture. Part 2 introduces the logical model that implementation profiles map onto selected platforms, products, protocols, and deployment environments. It serves as the primary bridge between the conceptual architecture in Part 1 and the technology-specific implementation profiles in later parts.

Contents

- [Foreword](#)
- [Introduction](#)
- [1. Scope](#)
- [2. Relationship to Part 0 and Part 1](#)
- [3. Traceability to Parent and Source Architectures](#)
- [4. Logical Architecture Principles](#)
- [5. Logical Architecture Overview](#)
- [6. Logical Node Model](#)
- [7. Logical Communication Model](#)
- [8. Logical Information Model](#)
- [9. Logical Runtime Plane Model](#)

- 10. Logical Interaction Patterns
- 11. Logical Governance Model
- 12. Logical Traceability Model
- 13. Relationship to Domain Logical Profiles
- 14. Relationship to Implementation Profiles / PSM
- 15. Logical Rules and Constraints
- 16. Logical Requirements

Contents

* Part 2: Distributed Node-Based Logical Architecture / PIM

- Foreword
- Introduction
- 1. Scope
- 2. Relationship to Part 0 and Part 1
- 3. Traceability to Parent and Source Architectures
 - 3.1 Traceability to SIP-RA
 - 3.2 Alignment with FDIS-RA
 - 3.3 Traceability to Part 1 Conceptual Architecture
 - 3.4 Coverage of the Original FX Demo Reference Architecture
 - 3.5 Traceability Position
- 6. 4. Logical Architecture Principles
 - 4.1 Distributed Node-Based Architecture
 - 4.2 Explicit Communication over Implicit Coupling
 - 4.3 Platform Independence
 - 4.4 Logical Separation of Runtime Planes
 - 4.5 Explicit Information Structures
 - 4.6 Traceability from Concept to Logical Model
 - 4.7 Evidence-Oriented Logical Design
 - 4.8 Policy-Governed Sharing Without Centralised Data Routing
- 7. 5. Logical Architecture Overview
 - 5.1 Overview
 - 5.2 Logical Architecture Context
 - 5.3 Distributed Logical Node Network
 - 5.4 Logical Communication Model
 - 5.5 Logical Information Model
 - 5.6 Logical Runtime Plane Model
 - 5.7 Logical Traceability Model
- 8. 6. Logical Node Model
 - 6.1 Overview
 - 6.2 Logical Node
 - 6.3 Logical Node Identity
 - 6.4 Logical Node Role
 - 6.5 Logical Node Responsibility
 - 6.6 Logical Node Boundary
 - 6.7 Logical Node Collaboration
 - 6.8 Logical Node Lifecycle State
- 9. 7. Logical Communication Model

- 7.1 Overview
- 7.2 Logical Communication Endpoint
- 7.3 Communication Role
- 7.4 Publisher Role
- 7.5 Subscriber Role
- 7.6 Requester Role
- 7.7 Responder Role
- 7.8 Command Producer Role
- 7.9 Command Consumer Role
- 7.10 Event Producer Role
- 7.11 Event Consumer Role
- 7.12 Communication Direction and Coupling
- 7.13 Communication Pattern Independence
- 10. 8. Logical Information Model
 - 8.1 Overview
 - 8.2 Logical Data Structure Definition
 - 8.3 Logical Data Structure Instance
 - 8.4 Logical Message
 - 8.5 Logical Event
 - 8.6 Logical Command
 - 8.7 Logical Acknowledgement
 - 8.8 Logical Assertion
 - 8.9 Logical Record
 - 8.10 Logical Information Lineage
- 11. 9. Logical Runtime Plane Model
 - 9.1 Overview
 - 9.2 Logical Control Plane
 - 9.3 Logical Data Plane
 - 9.4 Logical Health and Observability Plane
 - 9.5 Logical Policy and Release Plane
 - 9.6 Logical Audit and Provenance Plane
 - 9.7 Cross-Plane Logical Relationships
- 12. 10. Logical Interaction Patterns
 - 10.1 Overview
 - 10.2 Register Node
 - 10.3 Report Node Status
 - 10.4 Publish Domain Information
 - 10.5 Subscribe to Domain Information
 - 10.6 Issue Control Command
 - 10.7 Acknowledge Control Command
 - 10.8 Request Policy Decision
 - 10.9 Enforce Policy Decision and Produce Authorised Release
 - 10.10 Record Audit and Provenance
 - 10.11 Replay or Reconstruct Information
 - 10.12 Logical Interaction Pattern Summary
- 13. 11. Logical Governance Model
 - 11.1 Overview
 - 11.2 Logical Ownership of Definitions
 - 11.3 Logical Versioning
 - 11.4 Logical Compatibility
 - 11.5 Logical Change Control

- 11.6 Logical Admission Rules
- 11.7 Logical Policy Constraints
- 14. 12. Logical Traceability Model
 - 12.1 Overview
 - 12.2 Part 1 Concept to Part 2 Logical Element
 - 12.3 Logical Node Traceability
 - 12.4 Logical Endpoint Traceability
 - 12.5 Logical Data Structure Traceability
 - 12.6 Logical Runtime Plane Traceability
 - 12.7 Logical Interaction Traceability
 - 12.8 Logical Evidence Traceability
 - 12.9 Logical Traceability Summary
- 15. 13. Relationship to Domain Logical Profiles
 - 13.1 Overview
 - 13.2 How Domain Profiles Specialise the Logical Architecture
 - 13.3 Domain Profile Additions
 - 13.4 What Domain Profiles Preserve
 - 13.5 Relationship to the FX Demo Logical Profile
 - 13.6 Boundary Between Part 2 and Domain Logical Profiles
- 16. 14. Relationship to Implementation Profiles / PSM
 - 14.1 Overview
 - 14.2 Logical Elements Prepared for Technology Mapping
 - 14.3 Communication Technology Independence
 - 14.4 Data Representation Independence
 - 14.5 Execution and Deployment Independence
 - 14.6 Implementation Mapping Boundaries
 - 14.7 Relationship to Phase 0 Implementation Profile
 - 14.8 Boundary Between Part 2 and Implementation Profiles
- 17. 15. Logical Rules and Constraints
 - 15.1 Overview
 - 15.2 Logical Nodes Remain Distinct from Runtime Deployments
 - 15.3 Logical Endpoints Remain Distinct from Communication Technologies
 - 15.4 Logical Data Structures Remain Distinct from Serialisation Formats
 - 15.5 Logical Runtime Planes Remain Distinct from Implementation Groupings
 - 15.6 Logical Interactions Remain Traceable
 - 15.7 Logical Architecture Does Not Prescribe Deployment Topology
 - 15.8 Logical Architecture Does Not Prescribe Communication Pattern Implementation
 - 15.9 Logical Architecture Does Not Prescribe Governance Tooling
 - 15.10 Domain Profiles Preserve the Logical Architecture
 - 15.11 Implementation Profiles Preserve the Logical Architecture
 - 15.12 Logical Constraints Summary
- 18. 16. Logical Requirements
 - 16.1 Overview
 - 16.2 Distributed Node-Based Architecture Requirements
 - 16.3 Logical Node Requirements
 - 16.4 Logical Communication Requirements
 - 16.5 Logical Information Requirements
 - 16.6 Logical Runtime Plane Requirements
 - 16.7 Logical Interaction Requirements
 - 16.8 Logical Governance Requirements
 - 16.9 Logical Traceability Requirements

- [16.10 Profile and Implementation Boundary Requirements](#)

Notes for Editors

This document should be maintained as a set of linked wiki pages rather than a single monolithic page. Each major section should have its own stable URI so it can be referenced from other wiki pages, exported documents, presentations, AI-assisted tools, and semantic models.

Do not rename pages once they have been cited externally unless a redirect or move plan is in place.

© 2026 Dido Solutions, Inc. and Jackrabbit Consulting, Inc.

From:

<https://wiki.didosolutions.com/> - **Dido Solutions, Inc Wiki**

Permanent link:

<https://wiki.didosolutions.com/fxdemo/02-part/start?rev=1783108709>

Last update: **2026/07/03 12:58**

