

Introduction

[Go to Top](#)

[Financial systems](#) often operate across institutions, platforms, jurisdictions, organizations, and technical environments. They exchange structured information, coordinate operational behavior, apply rules, produce analytical results, control release, record [provenance](#), and preserve [evidence](#). A Logical Architecture must organize those responsibilities before any implementation profile selects technologies, products, protocols, programming languages, deployment mechanisms, or runtime infrastructure.

[Part 1: Conceptual Architecture](#) defines the conceptual foundation for the Financial Systems [Archetype](#). This part builds outward from that foundation by defining a distributed, node-based Logical Architecture as a [Platform Independent Model](#). It treats the system as a network of identifiable logical [Nodes](#) that communicate via explicit logical [Communication Endpoints](#) and exchange logical Data Structure Instances conforming to logical Data Structure Definitions.

This part makes distribution a logical architectural choice. The Logical Architecture supports independently operating Logical Nodes, cross-boundary communication, and participation in [Runtime Planes](#), including the [Control Plane](#), [Data Plane](#), [Health and Observability Plane](#), [Policy and Release Plane](#), and [Audit and Provenance Plane](#). This logical distribution supports modularity, separation of concerns, traceability, evidence, governance, and independent evolution.

This part remains technology-neutral. Implementation profiles select mechanisms such as [DDS](#), [REST](#), [RPC](#), Protocol Buffers, message queues, event streams, file exchange, shared services, libraries, threads, tasks, processes, containers, virtual machines, or other mechanisms to realize the logical architecture. Those choices support a selected implementation context. They do not define the Logical Architecture / PIM.

This part provides the reusable logical foundation for domain profiles, including the FX Demo Logical Profile. Domain profiles specialize this Logical Architecture for a specific financial [Domain](#). Implementation profiles then map the Logical Architecture to selected technologies, and deployment, testability, and evidence plans define how those implementations run, produce evidence, and support review.

© 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/00-introduction/start?rev=1784403224>

Last update: **2026/07/18 12:33**

