

NDH 架構

文件編號: IDTF-V3.5-06-01 版本: 1.1 日期: 2025-10-16 作者: 林志錚 Michael Lin(Chih Cheng Lin)

1. 簡介

本文件旨在詳細闡述 IDTF V3.5 中 NDH (Neutral Data Hub) 的架構，特別是其與 MCP Control Plane 和 Agent Runtimes 層次的互動。NDH 作為工業數位分身框架的核心數據中樞，負責數據的採集、處理、儲存和分發，並為上層應用和 AI Agent 提供統一的數據服務。

2. NDH 核心架構

NDH 採用分層架構，確保數據流的清晰性、模組化和可擴展性。新的架構圖將 MCP Control Plane 和 Agent Runtimes 層次納入，以支援更智能的 Agent 協作和控制。

2.1 架構概述



2.2 各層職責

- **Apps / Dashboards:** 最上層的應用程式和儀表板，透過 WebSocket 或 HTTP 協議與 Service Layer 互動，提供使用者介面和數據視覺化。
- **Service Layer (服務層):** 提供 REST、GraphQL 或 WebSocket 介面，供上層應用程式存取 NDH 的數據和服務。負責認證、授權和 API 路由。
- **Processing Layer (處理層):** 負責數據的流式處理、批次處理和 AI 分析。包括數據清洗、轉換、聚合、異常檢測和預測模型運行等。

- **USD Integration Service (USD 整合服務):** 作為 IADL/FDL 語義數據與 USD 3D 場景之間的橋樑。負責解析 IADL/FDL 定義、動態生成或更新 USD 場景圖、實時數據同步、命令執行映射、USD 場景管理以及與 Omniverse Connect 的整合。
- **USD Scene / Omniverse (USD 場景 / Omniverse):** 透過 USD API 或 Omniverse Connect 接收 USD 整合服務的數據，提供 3D 視覺化、物理模擬和 AI 驅動的應用能力。
- **NDH (Data Hub):** NDH 的核心數據中樞，整合了 Kafka (訊息佇列)、TSDB (時序數據庫)、Postgres (關係型數據庫) 和 Redis (快取/訊息代理) 等組件，並提供統一的事件總線。
- **Data Acquisition (數據採集層):** 負責從各種 OT/IT 系統採集數據，支援 OPC UA、MQTT 等多種工業協議。採集到的數據會發送到 NDH Data Hub。
- **MCP Control Plane (MCP 控制平面):** 負責 AI Agent 的註冊、管理、排程和審計。提供 ACL (Access Control List) 管理 Agent 的權限，並透過 agents/# MQTT Topics 與 Agent Runtimes 進行通訊。
- **Agent Runtimes (Agent 運行時):** 執行 AI Agent 的實際運行環境，支援 Python、Go、Node.js 等多種語言。Agent Runtimes 透過 SDK 和策略與 MCP Control Plane 互動，並將決策結果回寫至 NDH。

3. NDH 資產層次結構的建構機制

NDH (Neutral Data Hub) 作為 IDTF 的核心數據中樞，其關鍵職能之一是根據 FDL (Factory Design Language) 和 IADL (Industrial Asset Description Language) 的定義，動態且智慧地建構和維護一個統一的資產層次結構。這個層次結構不僅反映了工廠的物理佈局和邏輯分區，也整合了資產的靜態屬性與實時運行數據，為上層應用、AI Agent 和 Omniverse USD Scene Graph 提供一致且豐富的數據視圖。

3.1. NDH 資產層次結構的核心原則

1. **ISA-95 對齊:** NDH 內部資產層次結構的設計嚴格遵循 ISA-95 設備階層模型，確保與工業標準的互操作性。
2. **FDL 驅動:** FDL 文件是定義工廠佈局和資產實例組織的「藍圖」，NDH 將其解析為可操作的內部結構。
3. **IADL 豐富:** IADL 提供資產的詳細「DNA」，包括其類型、屬性、數據點和 3D 模型引用，這些信息將被整合到層次結構中的每個資產節點。

4. **動態與實時:** 層次結構不僅包含靜態定義，還能與實時數據源（透過 Asset Servant）綁定，反映資產的當前狀態和行為。
5. **可擴展性:** 支援動態添加、修改和移除資產或其關係，以適應工廠生命週期的變化。

3.2. 從 FDL 到 NDH 邏輯層次結構的建構流程

NDH 內部資產層次結構的建構是一個多階段的過程，涉及 FDL 解析、IADL 引用解析、實例化、層次化和數據綁定。

1. FDL 解析與驗證 (FDL Parsing & Validation):

- NDH 的 FDL 解析器首先讀取 FDL YAML 文件。它會根據預定義的 FDL JSON Schema 對文件內容進行嚴格的語法和結構驗證，確保其符合規範。
- 解析器將 FDL 文件轉換為 NDH 內部可操作的數據模型，其中包含 `factory_design`、`buildings`、`layout` (包含 `areas` 和 `instances`) 等核心元素。

2. IADL 引用解析與整合 (IADL Reference Resolution & Integration):

- 對於 FDL 中 `layout.instances` 裡定義的每個資產實例，NDH 會提取其 `type` 字段 (例如 `DieSorter_v1.0`)，這代表了一個 IADL AssetType 的引用。
- NDH 會查詢其內部 IADL 註冊中心，載入對應的 IADL 定義。這些定義包含了資產的詳細藍圖，如 `data_tags` (數據點)、`properties` (靜態屬性)、`geometry` (3D 模型路徑) 和 `components` (子組件)。
- IADL 的信息將與 FDL 中資產實例的特定配置 (如 `instance_params`、`origin`、`orientation`) 進行整合，形成一個完整的資產實例描述。

3. 資產實例化與唯一識別 (Asset Instantiation & Unique Identification):

- NDH 根據 FDL `layout.instances` 中的 `count`、`naming_prefix` 和 `naming_pattern` 規則，為每個資產實例生成唯一的識別符 (例如 `DS_2F_A_001`, `DS_2F_A_002` 等)。這些識別符將作為 NDH 內部資產節點的唯一鍵。
- 對於每個實例，NDH 會創建一個內部資產節點對象，其中包含其唯一的 ID、類型、FDL 中定義的空間位置 (`origin`, `orientation`) 和 IADL 整合後的屬性。

4. 層次結構構建 (Hierarchy Construction):

- NDH 根據 FDL 中定義的物理和邏輯關係，將這些資產節點組織成一個階層樹狀結構。這個結構將直接反映 ISA-95 的設備階層：
 - **Enterprise**: FDL 的 `factory_design` 頂層。
 - **Site**: FDL 的 `buildings` 映射為 Site 節點。
 - **Area**: FDL 的 `floors` 和 `layout.area` 映射為 Area 或 Process Cell 節點。
 - **Unit/Equipment Module**: FDL 中的資產實例 (例如 `DS_2F_A_001`) 映射為 Unit 或 Equipment Module 節點。
 - **Control Module**: IADL 中定義的 `components` (例如 `Sensor_Pressure_01`) 映射為 Control Module 節點，作為其父資產的子節點。
- 這個層次結構將儲存在 NDH 的核心數據庫中，並可透過 API 進行查詢和遍歷。

5. Asset Servant 綁定與數據集成 (Asset Servant Binding & Data Integration):

- 對於每個資產實例節點，NDH 會根據其 IADL 定義中的 `data_tags` 部分，自動初始化一個或多個 **Asset Servant** 實例。
- 每個 Asset Servant 負責將資產的抽象 `tag_id` (例如 `discharge_pressure`) 映射到後端時序數據庫 (如 TDengine, PI System) 中的實際數據點。
- NDH 的資產節點將持有對這些 Asset Servant 的引用，從而提供統一的數據訪問接口。當應用層請求資產的實時數據時，NDH 會透過對應的 Asset Servant 進行路由和查詢，隱藏底層數據源的複雜性。

NDH 內部邏輯層次結構示例 (概念性，強調 ISA-95 對應):

```
''' Enterprise (FDL factory_design) └── Site (FDL Building: MainBuilding) └── Area
(FDL Floor: 2F) └── Process Cell / Area (FDL Layout Area: Production_Zone_A) └──
Equipment Module (FDL AssetInstance: DS_2F_A_001) |   └── Control Module (IADL
ComponentType: Sensor_Pressure_01) |   |   └── Asset Servant (映射
Sensor_Pressure_01 的數據點) |   └── Control Module (IADL ComponentType:
Actuator_Valve_01) |   └── Asset Servant (映射 Actuator_Valve_01 的數據點)
└── Asset Servant (映射 DS_2F_A_001 的數據點) └── Equipment Module (FDL
AssetInstance: DS_2F_A_002) |   └── Asset Servant (映射 DS_2F_A_002 的數據點)
└── Equipment Module (FDL AssetInstance: DB_2F_A_001) └── Asset Servant (映
射 DB_2F_A_001 的數據點)'''
```

3.3. 動態更新與生命週期管理

NDH 的資產層次結構是動態的，能夠響應工廠環境的變化：

- **FDL/IADL 更新:** 當 FDL 或 IADL 定義文件發生變化時，NDH 可以重新解析並更新其內部層次結構，自動調整資產的佈局、屬性或數據映射。
- **實時數據更新:** Asset Servant 持續監聽來自數據採集層的實時數據流，並更新其內部緩存或直接從時序數據庫中獲取最新值，確保層次結構中的數據始終是最新的。
- **資產生命週期事件:** NDH 可以整合資產的生命週期事件 (例如安裝、維護、退役)，並更新層次結構中資產的狀態，這與 ISA-95 的狀態轉換模型相符。

4. 資產層次結構到 Omniverse USD Scene Graph 的映射機制

將 NDH 中建構的資產層次結構映射到 NVIDIA Omniverse USD (Universal Scene Description) Scene Graph 是實現 3D 視覺化、物理模擬和 AI Agent 交互的關鍵步驟。USD 提供了一個強大且可擴展的框架來描述 3D 場景，其層次結構與 IDTF 的資產模型具有良好的對應關係，能夠有效地橋接數位分身數據與虛擬世界。

4.1. USD Scene Graph 基礎與 IDTF 相關性

USD Scene Graph 是一個有向無環圖 (DAG)，由 Prims (基本元素) 組成。Prims 可以是模型、材質、燈光、攝影機或任何其他場景元素。Prims 可以嵌套，形成一個階層結構，這與 ISA-95 和 FDL 所定義的資產階層概念高度契合。USD 的核心優勢在於其強大的組合性 (Composition) 和分層覆蓋 (Layering) 機制，允許不同團隊在同一場景上協同工作，並將數據與視覺表示分離，這對於複雜的工業數位分身應用至關重要。

4.2. NDH USD Integration Service 的映射策略

NDH 中的 **USD Integration Service** 將作為核心組件，負責將 NDH 內部建構的資產層次結構及其相關數據，轉換並發布到 Omniverse USD Scene Graph。這個服務將執行以下詳細的映射步驟：

1. 根節點映射 (Root Node Mapping):

- FDL 的 `factory_design` (對應 ISA-95 的 Enterprise 層級) 將映射為 USD Scene Graph 的根節點，通常是一個 `xform Prim`，例如

`/World/Factory_Harvatek`。這個根節點將作為整個工廠數位分身的容器。

2. 物理結構映射 (Physical Structure Mapping):

- FDL 中定義的 `buildings` (對應 ISA-95 的 Site 層級) 和 `floors` 將映射為 USD 中的 `Xform Prims`。這些 Prims 用於組織物理空間，並作為其內部資產的父節點。例如，`/World/Factory_Harvatek/MainBuilding/Floor_2F`。
- 這些 `Xform Prims` 將包含其自身的空間變換信息 (位置、旋轉、縮放)，反映建築結構在 3D 空間中的佈局。

3. 邏輯區域映射 (Logical Area Mapping):

- FDL 的 `layout.area` (對應 ISA-95 的 Area 或 Process Cell 層級) 也將映射為 USD 中的 `Xform Prims`，作為邏輯分區。例如
`/World/Factory_Harvatek/MainBuilding/Floor_2F/Production_Zone_A`。
- 這些邏輯區域 Prim 可以包含額外的元數據 (Metadata)，如 `zone_type` (`CleanRoom`, `ProductionZone` 等) 和環境參數 (溫度、濕度等)，這些信息可以來自 FDL 的定義。

4. 資產實例映射 (Asset Instance Mapping):

- FDL 中定義的每個資產實例 (例如 `DS_2F_A_001`，對應 ISA-95 的 Equipment Module) 將映射為一個 USD `Model Prim` 或 `Xform Prim`，具體取決於其複雜度和交互需求。
- **IADL 3D 模型引用:** 每個資產實例的 USD Prim 將透過 USD 的 Reference 或 Payload 機制，引用其 IADL 定義中指定的 `geometry.model_file` (例如 `.usd` 或 `.gltf` 文件)。這使得資產的視覺外觀可以獨立於其邏輯定義進行管理和更新。
- **空間變換:** FDL 中定義的 `origin` (位置) 和 `orientation` (方向) 參數將用於設置 USD Prim 的 `xformOp` 屬性，精確定位資產在 3D 場景中的位置和方向。這確保了數位分身與物理世界的一致性。

5. 組件映射 (Component Mapping):

- IADL 中定義的 `ComponentType` (對應 ISA-95 的 Control Module，如感測器、執行器) 可以映射為其父資產 USD Prim 的子 Prim。這允許在 3D 場景中獨立表示和交互資產的關鍵組件。

- 或者，如果組件沒有獨立的 3D 表示，其屬性可以直接附加到父資產 Prim 上，作為自定義屬性。

6. 屬性與數據綁定 (Attribute & Data Binding):

- **靜態屬性:** NDH 中來自 IADL 的靜態資產屬性 (例如製造商、型號、額定功率) 將作為 USD Prim 的自定義屬性 (Custom Attributes) 附加到對應的 USD 節點上。這些屬性可以是 `string`、`float`、`int` 等 USD 支援的數據類型。
- **動態數據:** 透過 NDH 的 Asset Servant 機制，資產的實時運行數據 (例如壓力、溫度、狀態、OEE) 將被提取並作為動態更新的自定義屬性綁定到 USD Prim 上。USD Integration Service 將監聽 Asset Servant 提供的數據更新，並實時更新 USD Scene Graph 中的對應屬性。這使得 Omniverse 中的應用可以直接訪問這些數據，實現數據驅動的視覺化、儀表板和實時監控。
 - 例如，`DS_2F_A_001` 的 USD Prim 可能會有 `custom:discharge_pressure` 和 `custom:operational_status` 等屬性，其值由 Asset Servant 提供並實時更新。

7. 關係映射 (Relationship Mapping):

- FDL 中定義的 `relationships` (例如 `material_flow`, `data_connection`, `power_supply`) 可以映射為 USD 中的關係 (Relationships) 或自定義屬性。USD 的關係機制允許在不同 Prims 之間建立強類型連接，這對於表示資產之間的物理或邏輯互聯至關重要，例如物料流路徑、數據通訊鏈路或能源供應網絡。
- 這些關係對於在 Omniverse 中進行模擬、路徑規劃或影響分析非常有用。

USD Scene Graph 結構示例 (對應上述 NDH 邏輯層次結構，強調數據綁定):

```
''' /World /Factory_Harvatek (FDL factory_design, ISA-95 Enterprise) /MainBuilding
(FDL building_id: MainBuilding, ISA-95 Site) /Floor_2F (FDL floor_id: 2F)
/Production_Zone_A (FDL area: Production_Zone_A, ISA-95 Area/Process Cell)
/DS_2F_A_001 (FDL AssetInstance, ISA-95 Equipment Module) asset_id =
"DS_2F_A_001" asset_type = "DieSorter_v1.0" references =
@./models/die_sorter.usd@ xformOp:translate = (10.0, 20.0, 0.0) xformOp:rotateZ = 0
custom:manufacturer = "VendorX" (來自 IADL 靜態屬性) custom:model = "DS-Pro" (來自
IADL 靜態屬性) custom:discharge_pressure = (動態數據) custom:operational_status =
(動態數據) custom:oee = (動態數據) rel material_output = (FDL relationships)
/Sensor_Pressure_01 (IADL ComponentType, ISA-95 Control Module) component_id =
"Sensor_Pressure_01" custom:value = custom:unit = "bar" /DS_2F_A_002
/DB_2F_A_001'''
```

4.3. 實時數據同步與交互

NDH 的 USD Integration Service 將利用 Omniverse Connect SDK 或 USD API，實現 NDH 內部資產數據與 Omniverse USD Scene Graph 之間的實時雙向同步。這包括：

- **數據推送到 USD:** 當 NDH 中的資產狀態或數據 (透過 Asset Servant) 發生變化時，USD Integration Service 會將這些更新推送到 USD Scene Graph 中的對應自定義屬性。Omniverse 中的應用 (如 Composer, Isaac Sim) 可以訂閱這些屬性變化，並實時更新 3D 視覺化或模擬狀態。
- **USD 事件回傳 NDH:** Omniverse 中的用戶交互或模擬結果 (例如，在虛擬環境中操作一個閥門，或模擬資產故障) 可以透過 USD 事件機制回傳到 NDH。USD Integration Service 將捕獲這些事件，並將其轉換為 NDH 可理解的指令或數據更新，進而影響物理世界或觸發 AI Agent 的響應。
- **高效數據流:** 透過優化數據傳輸協議和增量更新機制，確保大規模工業場景下的實時數據同步性能。

5. MCP Control Plane 與 Agent Runtimes 的互動

MCP Control Plane 和 Agent Runtimes 是 IDTF V3.5 中引入的關鍵組件，旨在實現多個 AI Agent 之間的協作和智能控制。

5.1 通訊機制

- **agents/# (MQTT Topics):** MCP Control Plane 和 Agent Runtimes 之間的主要通訊通道。Agent Runtimes 透過發布和訂閱 agents/# 命名空間下的 MQTT Topics 來交換事件、狀態和指令。
- **ndh/# (MQTT Topics):** Agent Runtimes 透過 ndh/# 命名空間下的 MQTT Topics 監聽來自 Data Acquisition 層的遙測數據、事件和控制指令，並將其決策結果回寫至 NDH 的 /actions 和 /logs topic。

5.2 數據流

1. **數據採集:** Data Acquisition 層從 OT/IT 系統採集數據，並將其發送到 NDH Data Hub 和 ndh/# MQTT Topics。
2. **Agent 監聽:** Agent Runtimes 監聽 ndh/# MQTT Topics，獲取實時數據和事件。

3. **Agent 決策**: Agent Runtimes 根據其內建的 AI 模型和策略進行決策，並可能與 MCP Control Plane 互動以獲取排程或權限資訊。
4. **Agent 協作**: 多個 Agent Runtimes 透過 `agents/#` MQTT Topics 進行協作，交換彼此的狀態和中間結果。
5. **決策回寫**: Agent Runtimes 將最終決策結果發布到 NDH 的 `/actions` topic，將運行日誌發布到 `/logs` topic。
6. **NDH 處理**: NDH Data Hub 接收 Agent 的決策和日誌，並進行相應的處理，例如更新資產狀態、觸發控制指令或儲存歷史數據。

6. 結論

MCP Control Plane 和 Agent Runtimes 的引入，使得 IDTF V3.5 能夠支援更複雜、更智能的工業應用。透過標準化的通訊協議和清晰的職責劃分，IDTF 為構建高效、可擴展的 AI Agent 生態系統奠定了堅實的基礎，進一步推動智慧工廠的發展。¹¹¹