04_IADL_V3_Specification.md - IADL V3.1 語法規範

1. 引言

IDTF (Industrial Digital Twin Framework) V3.1 引入了 IADL (Industrial Automation Description Language) V3.1，這是一種專為工業自動化領域設計的描述語言。IADL V3.1 旨在提供一個標準化、可擴展且易於理解的語法，用於描述工業數位孿生系統中的各種實體、行為和關係。本章將詳細闡述 IADL V3.1 的語法規範、核心概念及其在 IDTF 生態系統中的應用。

2. IADL V3.1 核心理念與設計原則

IADL V3.1 的設計秉承了 IDTF V2 的核心理念，旨在構建一個開放、中立且普惠的工業數位孿生生態系統。其主要設計原則包括：

- **開源中立，無供應商鎖定**：IADL V3.1 採用開放標準和開源實現，確保用戶在選擇工具和平台時擁有最大的自由度，避免被特定供應商綁定。
- **低成本，中小企業可負擔**：透過簡化的語法、高效的工具鏈和開源的生態系統，降低數位孿生技術的導入成本，使中小企業也能受益於先進的工業自動化解決方案。
- **全生命週期覆蓋**：IADL V3.1 支援從設計、建造到運維的整個工業資產生命週期，提供統一的描述方式，促進各階段數據的無縫流動和協同。
- **基於 IADL + NDH + Omniverse 三大核心組件**：IADL V3.1 作為描述語言，與 NDH (Neutral Data Hub) 負責數據整合與交換，以及 Omniverse 提供強大的即時協同與渲染能力，共同構成了 IDTF 的三大核心支柱。這些組件共同作用，使得 IDTF 能夠提供一個全面的數位孿生解決方案，而無需依賴單一的外部平台。

3. IADL V3.1 語法結構

IADL V3.1 採用基於文本的聲明式語法，易於人類閱讀和機器解析。其基本結構由模組（Module）、實體（Entity）、屬性（Property）、行為（Behavior）和關係（Relation）等關鍵元素組成。這種結構化的描述方式，使得複雜的工業系統能夠被清晰、一致地表達。

3.1 模組 (Module)

模組是 IADL V3.1 中組織和管理複雜系統的基本單元。每個模組可以包含多個實體定義、行為定義和關係定義，並可以引用其他模組，實現模組化設計和重用。

```
module MyFactory {
    // 定義工廠中的實體、行為和關係
    entity ProductionLine { /* ... */ }
    entity Robot { /* ... */ }
    relation Connects(ProductionLine, Robot);
}
```

3.2 實體 (Entity)

實體代表工業系統中的物理或邏輯對象，例如機器人、感測器、生產線或工廠區域。每個實體都有一組屬性來描述其狀態，並可以定義行為，反映其功能和交互能力。

```
entity Robot {
    property string  id;                  // 機器人唯一識別符
    property Location currentLocation;  // 機器人當前位置
    property bool    isOperating;         // 機器人是否正在運行
    behavior MoveTo(Location targetLocation); // 移動到指定位置的行為
}

entity Sensor {
    property string  id;                  // 感測器唯一識別符
    property double  value;               // 感測器讀取值
    property string  unit;                // 讀取值的單位
}
```

3.3 屬性 (Property)

屬性用於描述實體的特徵或狀態。IADL V3.1 支援多種基本數據類型（如 `string`, `int`, `double`, `bool`）以及複雜數據類型（如 `Location`, `Vector3` 等自定義類型），確保數據描述的豐富性和精確性。

```
struct Location {
    double x;
    double y;
    double z;
}

struct Vector3 {
    double x;
    double y;
    double z;
}
```

3.4 行為 (Behavior)

行為定義了實體可以執行的操作或響應的事件。每個行為可以有輸入參數和輸出結果，這使得 IADL 能夠描述動態的系統行為和交互邏輯。

```
behavior Robot.MoveTo(Location targetLocation) returns bool; // 機器人移動行為，返回是否成功
behavior ProductionLine.StartProduction(int batchSize);     // 生產線啟動生產行為
```

3.5 關係 (Relation)

關係用於描述實體之間的連接或交互。例如，「包含」、「連接到」或「監控」。這些關係對於理解系統的拓撲結構和功能依賴至關重要。

```
relation Contains(Factory, ProductionLine); // 工廠包含生產線
relation Monitors(Sensor, Robot);           // 感測器監控機器人
```

4. IDTF 原生能力與 IADL V3.1 的整合

IADL V3.1 不僅僅是一種描述語言，它更是 IDTF 實現其強大原生能力的基石。以下是 IADL V3.1 如何支撐 IDTF 關鍵技術設計的闡述，所有這些能力均為 IDTF 的原生功能，不依賴於特定的外部平台：

4.1 MEP 設計與 IADL

IDTF 的 MEP (Mechanical, Electrical, Plumbing) 設計能力透過 IADL V3.1 對於物理實體及其連接關係的精確描述來實現。IADL 允許定義 MEP 相關的實體（如管道、電纜、HVAC 設備）及其屬性（如流量、電壓、溫度）和連接點。這使得 IDTF 能夠基於 IADL 模型進行 MEP 系統的自動化設計、模擬和驗證，確保設計的準確性和效率。

4.2 衝突檢測 (Conflict Detection)

透過 IADL V3.1 對於空間位置、尺寸和連接關係的詳細描述，IDTF 能夠在設計階段自動執行衝突檢測。IADL 模型可以精確表示各個實體的三維幾何信息，IDTF 引擎則能基於這些信息識別潛在的物理碰撞或邏輯衝突，顯著提高設計質量並減少返工，從而節省成本和時間。

4.3 ISO 圖面生成 (ISO Drawing Generation)

IADL V3.1 提供了足夠的結構化信息，使得 IDTF 能夠自動生成符合 ISO 標準的工程圖面。從 IADL 模型中提取的實體幾何、尺寸、材料和連接信息，可以直接用於生成平面圖、立面圖、剖面圖以及詳細的部件清單，極大地提高了圖面生成的效率和準確性，減少了人工繪圖的錯誤。

4.4 流程分析 (Flow Analysis)

IADL V3.1 能夠描述工業流程中的實體行為、數據流和事件序列。IDTF 利用這些 IADL 定義來執行複雜的流程分析，例如生產線的吞吐量分析、物流路徑優化或能源消耗模擬。這有助於識別瓶頸、優化操作策略並提升整體系統效率，為決策提供數據支持。

4.5 即時協同 (Real-time Collaboration)

IDTF 透過 IADL V3.1 和 NDH (Neutral Data Hub) 的結合，實現了多用戶、多應用程式之間的即時協同。IADL 模型作為共享的數位孿生數據基礎，任何對模型的修改都能透過 NDH 即時同步給所有協同參與者，並在 IDTF 的可視化模組中即時呈現，確保所有團隊成員始終在最新的數據上工作，提高協同效率。

5. 程式碼範例

以下是一個簡化的 IADL V3.1 程式碼範例，展示如何描述一個包含機器人和感測器的生產單元：

```
module ProductionUnit {

    entity RobotArm {
        property string id = "RA-001";
        property Location position = {x: 10.0, y: 5.0, z: 2.0};
        property double jointAngle1;
        property double jointAngle2;
        property bool isBusy = false;

        behavior MoveToPosition(Location targetPosition) returns bool;
        behavior PickUpItem(string itemId) returns bool;
        behavior PlaceItem(string itemId, Location dropPosition) returns bool;
    }

    entity ProximitySensor {
        property string id = "PS-001";
        property Location position = {x: 12.0, y: 5.0, z: 1.0};
        property double detectionRange = 0.5; // meters
        property bool objectDetected = false;

        behavior Calibrate();
    }

    relation Monitors(ProximitySensor.PS-001, RobotArm.RA-001);
    relation LocatedAt(RobotArm.RA-001, {x: 10.0, y: 5.0, z: 0.0});
}
```

6. 結論

IADL V3.1 作為 IDTF 的核心描述語言，為工業數位孿生系統提供了一個強大、靈活且開放的語法規範。它不僅繼承了 IDTF V2 的開放中立理念，更透過精確的實體、行為和關係描述，全面支撐了 IDTF 在 MEP 設計、衝突檢測、ISO 圖面生成、流程分析和即時協同等方面的原生能力。IADL V3.1 將成為推動工業數位化轉型，實現智慧製造的關鍵驅動力。