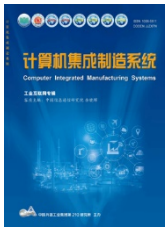

计算机集成制造系统
Computer Integrated Manufacturing Systems
ISSN 1006-5911,CN 11-5946/TP

《计算机集成制造系统》网络首发论文

题目：本体驱动的复杂产品设计模型集成
作者：董梦如，王国新，鲁金直，阎艳
DOI：10.13196/j.cims.2023.0299
收稿日期：2023-05-10
网络首发日期：2023-11-15
引用格式：董梦如，王国新，鲁金直，阎艳. 本体驱动的复杂产品设计模型集成[J/OL]. 计算机集成制造系统. https://doi.org/10.13196/j.cims.2023.0299

DOI: 10.13196/j.cims.2023.0299

本体驱动的复杂产品设计模型集成

董梦如[1]，王国新[1]，鲁金直[2+]，阎 艳[1]
（1. 北京理工大学 机械与车辆学院，北京 100081；2. 北京航空航天大学 航空科学及工程学院，北京 102206）

摘 要：针对复杂产品设计过程的需求模型、架构模型和追溯模型之间数据割裂、语义不一致、语义集成效率低等问题，提出一种基于本体的模型集成方法。首先结合模型体现形式和应用场景，分别建立需求、架构和追溯模型的本体框架。然后，采用基本形式化本体（Basic Formal Ontology, BFO）对该三个领域本体进行集成，形成一套用于面向基于模型的系统工程（Model-based Systems Engineering, MBSE）应用环境的通用的模型集成本体框架。基于该本体框架，构建本体转换引擎实现模型与本体的双向自动生成。最后，以智能电动汽车系统为例，验证了所提方法的有效性。

关键词：复杂产品设计；基本形式化本体；基于模型的系统工程；领域本体；语义集成
中图分类号：TP391　　**文献标识码**：A

Ontology-driven integration of complex product design models

DONG Mengru[1]，*WANG Guoxin*[1]，*LU Jinzhi*[2+]，*YAN Yan*[1]
(1. School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China;
2. School of Aeronautic Science and Engineering, Beihang University, Beijing 102206, China)

Abstract: Aiming at the problems of data separation, semantic inconsistency and low efficiency of semantic integration among requirement models, architecture models and traceability models in complex product design process, an ontology-based models integration method was proposed. Firstly, the ontology framework of requirement, architecture and traceability models was established by combining the model embodiment and application scenario. Then, the Basic Formal Ontology (BFO) was used to integrate the three domain ontologies, forming a general model integration ontology framework for model-based systems engineering (MBSE) application environment. Based on the ontology framework, the bidirectional automatic generation of models and ontology was realized. Finally, taking the intelligent electric vehicle system as a case, the effectiveness of the proposed method was verified.

Keywords: complex product design; basic formal ontology; model-based systems engineering; domain ontologies; semantic integration

0 引言

复杂产品设计涉及机械、电子、软件、控制等多个学科领域以及覆盖系统级、分系统级、设备级、单机级等多个层级，其设计过程涉及海量数据，传统以文档为中心的系统工程方法已经越来越难以满足大型复杂装备产品的设计。基于此，国际系统工程组织（INCOSE）提出了基于模型的系统工程（Model-based Systems

收稿日期：2023-05-10；修订日期：2023-09-22。Received 10 May 2023;accepted 22 Sep. 2023.
基金项目：部委科研资助项目（JCKY2022209A001）。Foundation item: Project supported by the Ministry's and National Commission's Scientific Research Foundation, China(No.JCKY2022209A001).

Engineering, MBSE）概念，通过建模方法的形式化应用，以使建模方法支持系统需求、设计、分析、验证和确认等活动，这些活动从概念性设计阶段开始，持续贯穿到设计开发以及后来的所有寿命周期阶段[1]。目前，MBSE 在航天、航空、汽车等领域具有广泛的研究和应用[2-12]。

从目前的研究成果来看，MBSE 的主要过程包括需求分析、功能分析、逻辑设计和物理架构设计[13-15]，需求是 MBSE 的基础，描述了系统应该具备的功能、性能、约束等方面的要求，指导着整个系统的设计和开发过程。功能、逻辑、物理架构模型的目标是将需求转化为可实现的设计，侧重于描述系统的结构、组件、模块、接口以及它们之间的交互，架构模型一定程度上反映了如何满足这些需求。基于此，本文将功能、逻辑、物理架构模型统一归类为架构，与需求进行区分，并将需求和架构之间的关联追溯关系归为一类，从而更好地确保设计的系统架构满足需求。

在需求分析环节，需求条目模型是一种广泛使用的需求模型。需求条目模型将自然语言需求转换为结构化的、可管理的条目，提供了清晰的需求层次关系，有助于更好地追溯和管理需求。目前存在多种工具支持需求条目模型的构建，如 IBM 的 DOORS、法国达索公司的 REQTIFY、MagicDaw 等。DOORS、REQTIFY 等不同需求工具之间通过传输相同格式的需求文件，如 csv，reqif 格式等，实现信息的交换和共享，即互操作[16]。在复杂系统设计过程中，涉及到多种需求管理工具，例如在智能电动汽车设计中，采用 Word、MagicDraw、MetaGraph 记录电动汽车的需求，然而，并不是所有需求工具都支持相同需求格式文件，导致需求知识重用率低、需求追溯困难；并且不同的需求工程师可能使用不同的术语，出现相同的术语表示于不同的概念、不同的术语表示相同的实体现象，导致语义不一致或者语义混淆，需求理解出现偏差问题。

在功能、逻辑、物理架构设计阶段，通常使用专门的建模语言来描述系统的结构、组件、接口、数据流等。然而，不同领域、层级之间往往采用不同的建模语言，例如智能电动汽车的设计，涉及到机械、电气、软件等多个领域，以及对象、系统、架构等多个层级，在设计中采用了 UML、SysML、AADL 等多种建模语言。建模语言是用于定义采用具体术语描述模型的标准语法和语义规则[17]，其中，语义（semantics）是对数据符号的解释，语法（syntax）是对于这些符号之间的组织规则和结构关系的定义。不同建模语言之间的语义和语法结构相差较大，对应的建模方法和工具也有所不同，例如，MagicDraw 工具支持 SysML 建模语言和 MagicGrid 方法，IBM Rhapsody 支持 SysML 建模语言和 Harmony SE 方法，OSATE 建模工具支持 AADL 建模、编译和分析等。在智能电动汽车的设计中，不同语言、方法、工具构建的架构模型之间存在语义不一致、模型异构的现象，导致智能电动汽车架构模型之间集成困难。

在需求、功能、逻辑、物理架构模型元素之间的追溯关系分析阶段，通常采用关系矩阵模型来表达追溯关系，例如，设计结构矩阵（Design Structure Matrix，DSM）、需求跟踪矩阵（Requirement traceability matrix，RTM）等。不同关系矩阵构建方式和应用场景有所不同，导致在实际应用中追溯模型之间也同样存在异构现象和集成困难问题。

此外，在复杂产品设计过程中，需求、架构、追溯关系三者密不可分，需求是架构设计的基础，也是判断架构是否正确的依据，追溯关系描述了需求和架构之间的关系。但由于需求分析和架构设计是两个相对独立的过程，并且需求、架构及追溯模型分别是三类不同类型的模型，存在数据割裂、语义不一致等现象，造成设计的架构无法满足需求、难以正确追溯需求和架构的关联关系等问题。

针对上述问题，目前常见的两种集成方式是数据集成和语义集成。数据集成是指将互相关联的分布式异构数据源集成到一起，使用户能够以透明的方式访问这些数据源[18]。语义集成是一种不仅仅考虑模式异构，而且考虑上下文和数值语义关系的集成方式[19]。数据集成是语义集成的基础，没有数据的整合和共享就无法进行语义的整合和共享。但是单纯的数据集成往往无法解决语义差异的问题，需要通过语义集成对数据的语义进行映射和整合，以便于数据在不同系统之间的共享和交流。因此，语义集成被越来越广泛地关注[20]。

本体是共享概念模型的明确的形式化规范说明[21]，是实现语义集成的重要方法。为了解决系统工程中语义异构的问题，本体被引入基于模型的系统工程领域，并被广泛应用。LIN J 等[22]针对工程设计领域提出了一个需求本体，该本体由零件、特征、需求和约束等元素组成，用于解决需求管理过程中需求沟通、可追溯性、完整性、一致性等问题。L.C. van Ruijven[23]基于 ISO 15926-11 和 ISO 15288-2008 标准，从系统工程的涉众需求定义过程、需求分析过程、验证过程、风险管理过程提出了系统工程本体，降低了流程、服务和数据级别上的语义互操作性障碍。Jean Duprez 等[24]基于 ISO/IEC/IEEE-42010 提出了一种构建系统架构框架本体的方法，以确保架构描述的语义一致性。王昊琪等[25]在公理化系统设计的基础上，手动构建了对应的本体模型，对系统设计的语义及其之间的关系进行了明确定义。综上可知，在 MBSE 中应用本体可以有效解决模型中存在的语义异构等问题，实现模型的语义集成。但仍然具有下列问题：

（1） 所提出的本体或单独针对需求工程领域，或针对特定领域或场景的架构模型构建，不同领域本体构建方法各不相同，导致构建的领域本体之间仍然存在本体异构的现象，重用性较差。

（2） 缺乏针对需求、架构、追溯模型的一体化本体，导致需求和架构设计间仍然存在信息割裂的问题。

（3） 本体主要通过手动构建，工作量大，当利益相关者需求发生变更时，难以对已经构建好的本体进行修改，本体重用性和扩展性较差，语义集成效率低。

因此，本文提出了一个基于顶层本体集成领域本体的模型集成方法，来实现复杂产品设计模型的集成，解决需求、架构、关联追溯各自领域的语义异构，以及三者之间数据割裂、语义异构、语义集成效率低的问题。该方法设计构建了需求、架构、关联追溯领域本体，并基于顶层本体—基本形式化本体（Basic Formal Ontology，BFO）对三者进行集成，形成一套面向 MBSE 应用环境通用的模型集成本体框架，在此基础上构建本体转换引擎实现模型与本体的双向打通。最后，通过智能电动汽车作为案例，证明了该方法的有效性。

1 复杂产品领域本体设计构建

领域本体是描述特定领域的一种专门本体，它给出了领域实体概念及其相互关系，领域活动以及该领域所具有的特征和规律的一种形式化描述[26]。在本体构建中，网络本体语言（Web Ontology Language，OWL）是W3C 基于 XML/RDF 等标准提出的本体描述语言，具有很强的语义表达和推理能力，并可以利用流行的本体描述工具 protégé来描述制造资源本体[27]。因此，本文使用 OWL 语言和 protégé工具进行领域本体的设计构建以及后续集成本体的构建。

1.1 需求领域本体设计构建

需求交换格式（Requirements Interchange Format，ReqIF）具有丰富的元模型，可以跨越工具和组织的界限，进行无损失的数据交换[28]。在需求管理领域，ReqIF 已经成为 OMG 的标准，用于在不同的需求管理工具之间交换需求。目前尚未出现任何一种标准或语言可以替代 ReqIF 标准的全部功能。因此，本文基于 ReqIF 标准设计需求条目模型本体。

ReqIF 标准涉及的术语如表 1 所示。

表 1 ReqIF 术语

术语	解释
ReqIF	ReqIF 是一种基于 XML 的需求格式，用作交换格式，XML 数据通常存储在扩展名为.reqif 的文件中。
Specifications	需求表单，通过需求表单将组织需求对象组织起来。
SpecObjects	需求条目，用于存储需求信息的数据结构。
SpecRelations	需求条目关联，描述两个需求条目之间的链接关系。
SpecRelationGroups	关联组，对不同的需求关联进行组织以及需求关联之间的关系进行描述。
SpecHierarchy	需求的层级，描述需求条目之间的层次关系。
SpecType	需求类型，包括属性定义和数据类型定义。
AttributeDefinition	属性定义。
DatatypeDefinition	数据类型定义。

基于 ReqIF 标准涉及的术语及术语之间的关系，设计构建需求领域本体框架。为了更好地区分和理解需求元素与需求类型的关联，取消需求类型 SpecType 单独作为类，在需求条目、需求条目关联、需求表单类下分别设置有无需求类型的子类。同时，为了更贴切需求条目模型的表达，将需求条目模型中的列抽象为 Column 类，设计基于 ReqIF 标准的需求模型领域的本体类框架，如图 1 所示。

在类框架的基础上，定义需求条目模型和本体之间的转换规则，明确需求条目模型的语义及其关系，实现语义集成和需求规范化表达。图 2 描述了基于 OWL 的 ReqIF 模型（需求条目模型）与本体之间的转换规则。

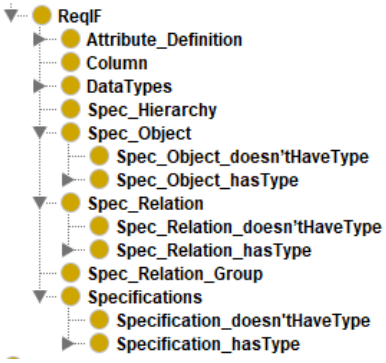

图 1　需求领域本体类框架

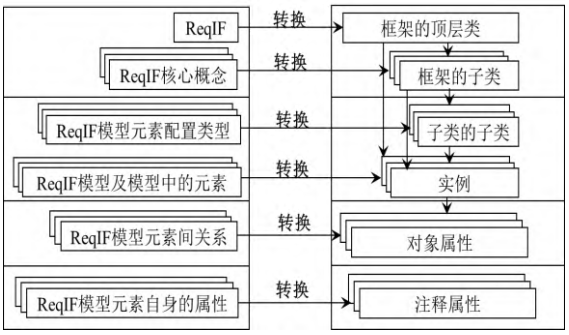

图 2　ReqIF 模型与本体之间的转换

转换规则描述如下：

（1）首先，生成图 1 所示的 ReqIF 模型的领域本体类框架。

（2）将 ReqIF 模型中属性定义及数据类型的具体类型生成领域本体类框架中对应类的子类。

（3）将 ReqIF 模型根据相关子类转化为实例(individuals)。

（4）将 ReqIF 术语之间的关系以及需求实例之间的关系转换为对象属性(Object properties)，如具有属性(has_Attribute)、具有类型(has_type)等。

（5）将 ReqIF 模型中各元素本身具有的属性转换为注释属性(Annotation properties)，如描述(Desc)、标识符(Identifier)、最后一次更改时间(Last Change)、长名称(Long Name)等。

（6）最后，基于对象属性定义实例之间的相互关系，基于注释属性定义实例本身具有的性质。通过使用实例、对象属性和注释属性，将 ReqIF 模型转换为用 OWL 语言描述的需求条目领域本体。

需求领域本体基于 ReqIF 标准为需求模型提供了一种统一的、标准化的知识表示，以及一致的语义描述，为不同需求工具构建的需求模型之间的信息交换和共享提供了载体，从而保证需求之间的一致性。

1.2　架构领域本体设计构建

基于元元模型的多架构建模方法可以对元模型进行重用和扩展，解决系统变化引起模型结构组成改变的问题，能够自主开发 MBSE 方法实施所需的架构[29]。通过对现有元建模方法 ARIS、Ecore、GME、GOPPRR、MS DSL Tools 和 MS Visio 具有的元元模型进行分析比较，发现 GOPPRR 是表达力最强的元元模型之一[30]。LU J 等在 GOPPRR 的基础上进行扩展提出 GOPPRR-E[31]元建模方法，进一步丰富元模型的表达能力，并提出统一建模语言 KARMA[32]（**K**ombination of **AR**chitecture **M**odel Specific**A**tion）支持该元建模方法。因此，本文采用 LU J 等提出的 GOPPRR-E 元建模方法和统一建模语言 KARMA 进行复杂产品架构模型构建及领域本体的设计。

GOPPRR-E 在 M0–M3 建模框架[31]的基础上，受 GOPPRR 元模型及其扩展的启发，构建 MBSE 模型语法和语义。GOPPRR-E 的概念关系如图 3 所示，各元素含义如下：

1）图（Graph）：对象、点、关系、角色、属性以及它们之间连接关系的集合。

2）对象（Object）：构造图的实体。

3）点（Point）对象中的连接端口。

4）关系（Relationship）：点和/或对象之间的连接。

5）角色（Role）：用于定义具有相关关系的绑定限制。一个关系与两个角色关联。

6）属性（Property）：元模型的一个特定属性，被其他五个元元模型（图、对象、关系、角色和点）调用。

7）扩展（Extension）：用于构造元模型的附加约束。

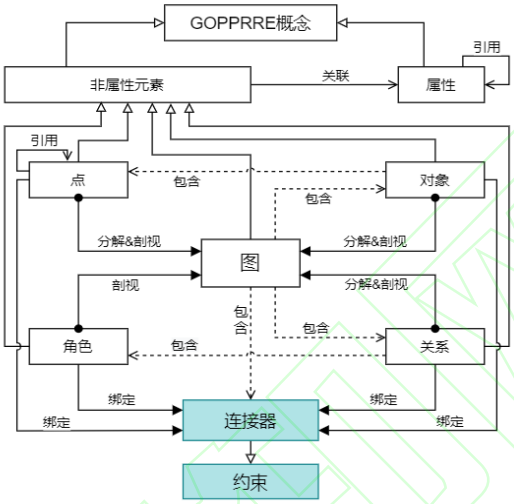

图 3　GOPPRR-E 的概念关系

基于 GOPPRR-E 的概念关系，设计构建架构领域本体框架。其中，“E”对应 Connector 类。考虑到架构模型的不同建模语言，定义 Language 类用于表示建模语言；定义 Project 类表示架构模型所属的项目；定义 Plugin 类表示架构模型可能具有的其它插件；形成基于 GOPPRR-E 的架构模型领域的本体类框架，如图 4 所示。

在类框架的基础上，定义架构模型和本体之间的转换规则，明确架构模型的语义及其之间的关系，实现不同建模语言之间的语义集成。图 5 描述了基于 OWL 的 GOPPRR-E 核心要素与本体之间的转换规则。

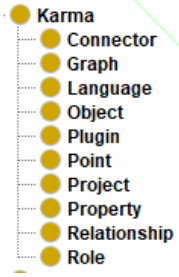

图 4　架构领域本体类框架

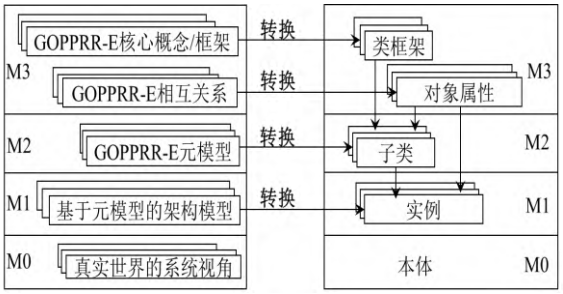

图 5　GOPPRR-E 核心要素与本体之间的转换规则

转换规则描述如下：

（1）首先，生成图 4 所示的架构模型的领域本体类框架。GOPPRR-E 之间的相互关系转换为对象属性。

（2）GOPPRR-E 元模型被转换为相关 GOPPRR-E 类的子类。

（3）GOPPRR-E 模型转换为对应子类的实例。

（4）基于对象属性，定义实例之间的相互关系。此外，通过数据属性定义每个属性的值，数据属性类型定义每个属性的数据类型。

（5）最后，使用实例、数据属性和对象属性，将代表真实世界视图的 MBSE 架构模型转换为用 OWL 语言描述的架构领域本体。

架构领域本体基于 GOPPRR-E 元建模方法为不同建模语言提供了一种统一的、标准化的知识表示，以及一致的语义描述，为不同建模工具、不同建模语言构建的架构模型之间的信息交换和共享提供了载体。

1.3 追溯领域本体设计构建

设计结构矩阵 DSM 是在复杂产品的多领域协同设计过程中解决耦合关系的有效途径，可以反映各种设计行为以及它们之间的相互关系[33]。相比于其它追溯关系模型，DSM 不限于特定领域，可用于各种类型的系统模型元素关联追溯分析。DSM 以紧凑的方式表达大量系统元素及其关系，支持对系统元素进行模块化和分组，并且 DSM 对角线上方和下方是相同的、对称的元素，上方和下方的两种元素间的相互关系结合起来构成了系统中元素之间的完整关系。因此，采用 DSM 来描述需求与架构之间的关联追溯关系，如图 6 所示。DSM 是一个具有相同数量的行与列的 n 阶方阵，首行和首列代表架构模型元素和需求模型元素，中间部分表示二者之间的追溯关联关系，对角线部分不可编辑。基于 DSM 构建追溯矩阵本体，对追溯矩阵的语义及其之间的关系进行明确定义，进而实现需求与架构关系的形式化表达，提高追溯关联关系的可继承性。

由于 DSM 中涉及需求条目模型元素、架构模型元素（采用 KARMA 语言构建，故将架构模型元素命名为 KARMA_Model_Element）、关联关系三类核心元素，故据此设计构建追溯领域本体类框架，如图 7 所示。

在类框架的基础上，定义 DSM 模型和本体之间的转换规则，明确需求和架构之间的关系，实现关系的形式化表达。图 8 描述了基于 OWL 的追溯矩阵 DSM 核心要素与本体之间的转换规则。

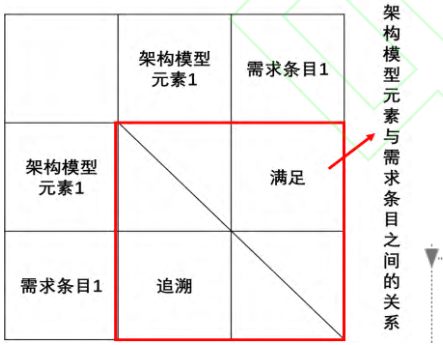

图 6 设计结构矩阵关联追溯表达

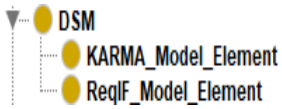

图 7 追溯矩阵本体类框架

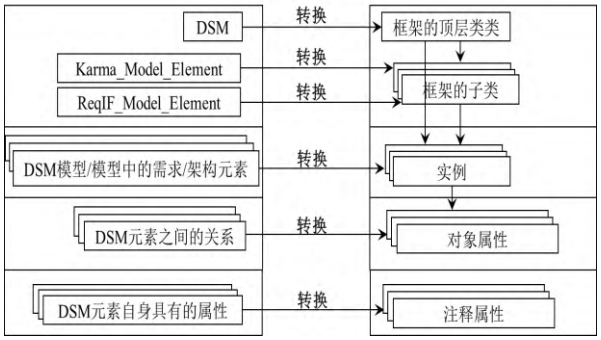

图 8 DSM 核心要素与本体之间的转换规则

转换规则描述如下：

（1）首先，生成图 7 所示的追溯矩阵 DSM 的领域本体类框架。

（2）将追溯矩阵 DSM 对应转换为 DSM 类的实例，DSM 矩阵中的元素对应生成两大子类的实例。

（3）将 DSM 矩阵、ReqIF 模型、KARMA 架构模型实例之间的相互关系转换为对象属性。

（4）将实例本身具有的属性转换为注释属性。

（5）通过使用实例、对象属性和注释属性，将 DSM 矩阵转换为用 OWL 语言描述的追溯领域本体。

通过追溯领域本体可以对具有追溯关系的需求、架构模型元素以及它们之间的追溯关联关系，进行形式化规范的描述，为不同工具构建的追溯模型的追溯信息的集成和共享提供了一套基于本体的规范。

2 基于顶层本体 BFO 的复杂产品领域本体集成

顶层本体是所有领域中共有的类别和关系的高度通用表示。基本形式化本体 BFO 与语言和认知工程领域本体（domain ontology for linguistic and cognitive engineering, DOLCE）、建议的顶层融合本体（suggested upper merged ontology, SUMO）并称为目前科学领域比较公认的三大顶层本体[34]。但相较于 DOLCE 和 SUMO，BFO 并不涉及具体的科学领域。BFO 是为支持集成、分析、整合科研数据而开发的顶层本体，设计更加简明，方便不同领域和粒度级别科学家复用，具有体量小而精的特点[35]。通过提供一个公共的顶层架构，来支持按照其理念构建的不同领域本体间的互操作。因此，在本文中选择 BFO 作为集成需求、架构、追溯领域本体的顶层本体。通过继承 BFO 合理的逻辑结构，形成一套新的规范的本体框架，以实现三者之间语义统一描述、数据的集成以及一体化表达。同时，领域本体可以复用顶层本体 BFO 的上层框架体系及标准，实现对顶层本体 BFO 的充分利用。

BFO 认为现实是由实体组成的，用“实体”来指代任何以任何方式存在的事物，并将实体分为持续体和发生体两大类。持续体是随着时间持续存在的实体，发生体是指计划发生或意外出现的实体。持续体和发生体下又可分为不同的层级结构，如图 9 所示。

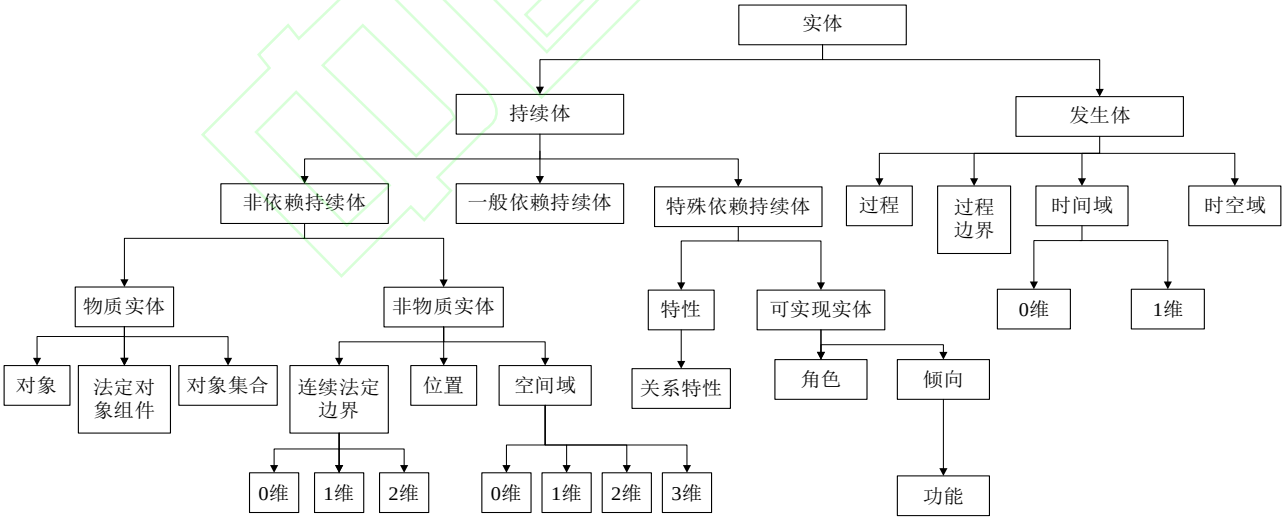

图 9 BFO 本体层级结构

为了更好地支持领域本体之间的集成和应用，采用工业本体组织 IOF（Industrial Ontologies Foundry，https://www.industrialontologies.org/）改进的 BFO 本体框架，为不同领域的本体提供一个共同的基础，使其可以共同利用 BFO 框架提供的通用概念和关系，避免重复建模和冗余内容。为了提高本体的通用性和普适性，IOF 组织改进的 BFO 本体框架在顶层本体 BFO 的基础上又扩展了一些本体，本体的类结构如图 10 所示。

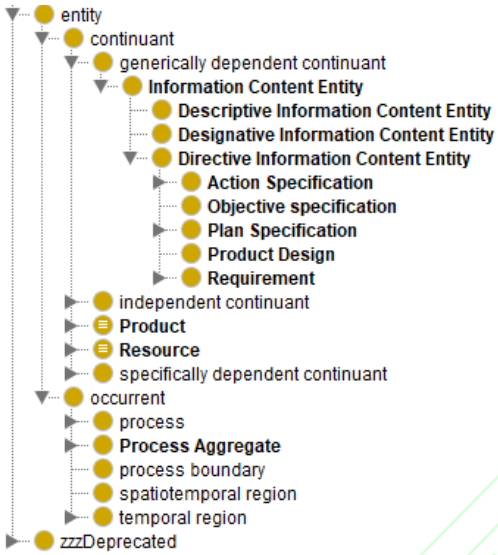

图 10　IOF 改进的 BFO 的类结构

BFO 中的持续体包括一般依赖持续体、非依赖持续体和特殊依赖持续体。一般依赖持续体指依赖于一个或多个非依赖持续体的持续体，这些持续体可以充当它的载体。IOF 在基本形式化本体 BFO 中的一般依赖持续体（generically dependent continuant）类下定义了信息内容实体（Information Content Entity）类，信息内容实体类下定义三种子类：描述性信息内容实体（Descriptive Information Content Entity）、指定性信息内容实体（Designative Information Content Entity）和指示性信息内容实体（Directive Information Content Entity），分别表示信息内容与信息内容之间的不同关系，即：描述、指定和指示。描述关系用于报告和图例等信息，例如需求报告对需求内容进行描述；指定关系用于指定名称和其他标识符等信息，例如指定一个模型的 ID；指示关系用于计划好的和工件规范等信息，用来传达指令、命令、规则、约束等等，用于控制某些操作或行为。例如将需求本体的名称规范为 Requirement。

考虑到本文设计的需求领域本体、架构领域本体以及追溯领域本体主要用于对需求条目模型、架构模型、追溯模型起到一个规定约束的作用，因此选择将该三个领域本体集成到 IOF 改进的 BFO 框架下的指示信息内容实体类下。同时，由于指示性信息内容实体类下包含需求（Requirement）子类，因此进一步将需求领域本体集成到需求类下。为了突出领域本体的集成结构，将图 10 的本体类结构中三种领域本体所属类及父类提取出来，集成的类结构如图 11 所示。

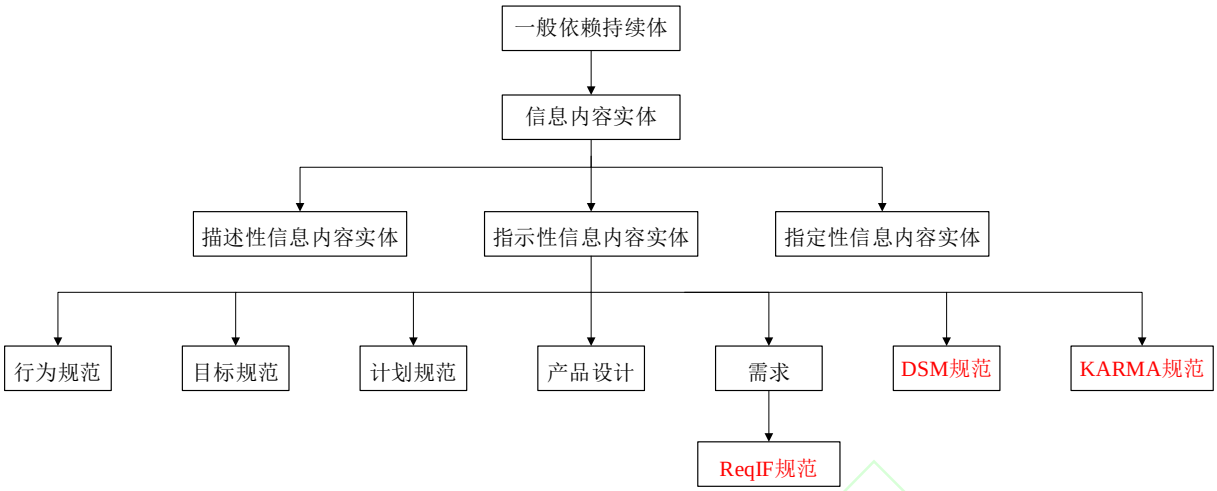

图 11　基于 IOF 改进 BFO 的三个领域本体的集成结构

基于图 10 的 BFO 框架及图 11 的领域本体集成结构，领域本体的具体集成方式如下：

（1）检查进行集成的本体是否存在语义不一致的情况。

（2）结合团队前期工作，由于架构模型是基于 GOPPRR-E 元建模方法通过多架构建模语言规范 KARMA 创建的，针对 GOPPRR-E 本体创建顶层类 KARMA，在 KARMA 下集成 GOPPRR-E 类。在 IOF 改进的 BFO 本体类结构中的指示性信息内容实体下，集成架构领域本体 KARMA。

（3）在指示性信息内容实体的子类需求（Requirement）下集成需求领域本体 ReqIF。

（4）在 IOF 改进的 BFO 本体类结构中的指示性信息内容实体下集成追溯领域本体 DSM。

基于 OWL 构建集成后的本体框架结构如图 12（c）所示，在图 10 的 IOF 改进的 BFO 框架基础上，采用图 11 的领域本体集成结构，对三类领域本体进行集成，形成一套基于 BFO 的通用的模型集成本体框架。

集成后的本体框架为需求、架构、关联追溯提供了一个共享的语义框架，通过本体框架中相关的概念和关系，帮助设计人员了解需求和架构的相关概念和关系，指导需求和架构设计的过程，有效解决需求、架构、关联追溯中的语义不一致、数据割裂的问题。此外，集成后的本体框架为不同领域的建模工具之间的互操作提供了一个通用的载体。通过提供一些标准接口和协议，不同的建模工具可以访问本体库，实现与其他建模工具的互操作。例如，MagicDraw 等其他建模工具通过访问本体库中的需求本体，与 MetaGraph2.0 工具中的需求插件共同使用这些本体进行需求分析和建模。

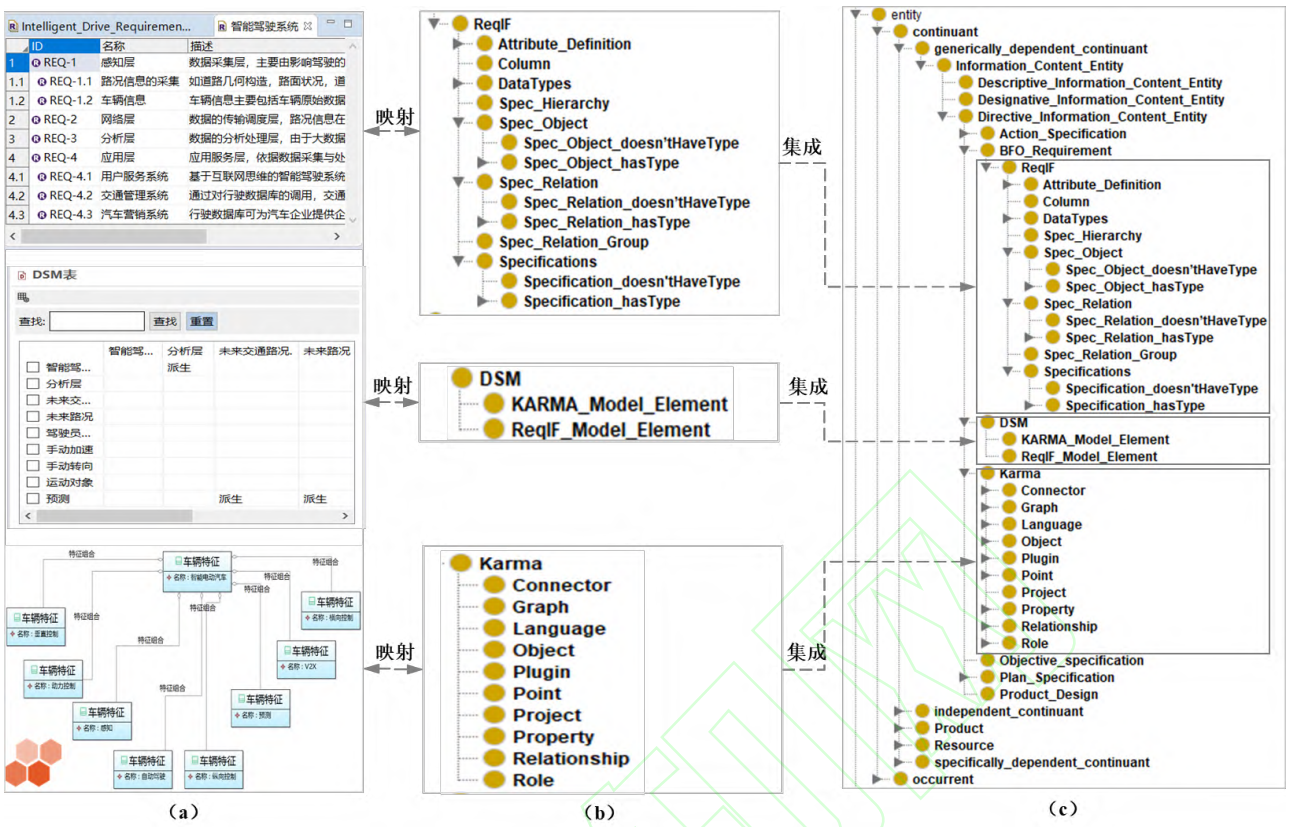

图 12　集成的本体框架

3　复杂产品模型与本体的自动生成

基于集成后的本体框架，以及需求条目模型、架构模型和追溯矩阵与本体之间的转换规则，在 MetaGraph2.0 工具中构建本体转换引擎，调用 Dom4j API 和 Jena API 实现模型与本体的自动生成，进一步实现复杂产品模型信息与本体数据的集成，解决手动构建本体工作量大、难以对已经构建好的本体进行修改等问题，提高语义集成效率。

3.1　复杂产品模型自动生成本体

模型自动生成本体的过程需要读取模型中的信息，并基于集成的本体框架生成对应的本体。采用 Dom4j API 对模型的信息进行读取，并通过 Jena API 生成对应的本体。相关过程如图 13 所示。

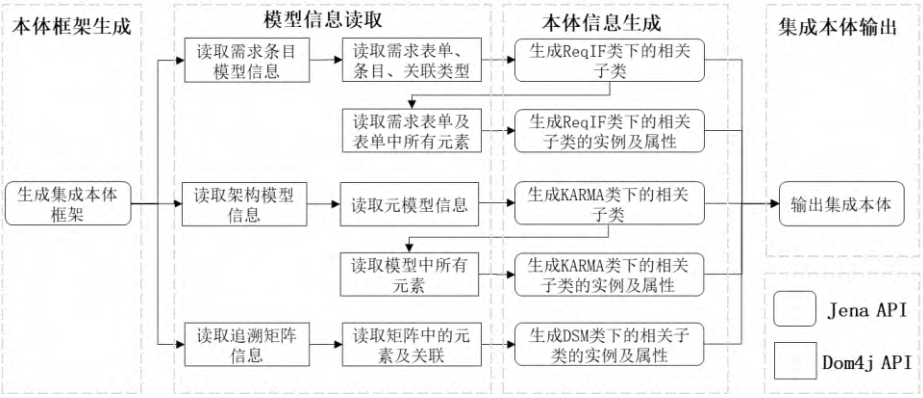

图 13　模型自动生成本体的过程

由图 13 可知，模型自动生成本体主要包括两大步骤，模型信息的读取，对应本体框架下本体的生成。两

部分伪码如下：

```
//模型自动生成本体1：读取模型信息
//读取模型文件
File A = new File(url);
SAXReader reader = new SAXReader();
Document B = reader.read(A);
//读取根节点
Element root = document.getRootElement();
//遍历根的子元素：模型元素
for (Iterator<Element> it = root.elementIterator(); it.hasNext();) {
Element element = it.next();
//读取模型元素内容
String classId = element.attribute("CLASSIDENTIFIER").getValue();
String id = element.attribute("IDENTIFIER").getValue();
}
```

（a）读取模型信息

```
//模型自动生成本体2：生成本体
//生成注释属性
AnnotationProperty annotationProId = ontModel.createAnnotationProperty("ID");
//生成对象属性（关系）
ObjectProperty objectProOwn = ontModel.getObjectProperty("own");
//生成类
OntClass owlRefClass = ontModel.createClass("className");
//生成类的属性
owlRefClass.addProperty(annotationProId, classId);
//生成类之间的关系
SomeValuesFromRestriction sometype = ontModel.createSomeValuesFromRestriction(null,
objectProOwn, owlRefClass);
owlRefClass.addEquivalentClass(sometype);
//生成实例
Individual owlIndividual1=ontModel.createIndividual(“IndividualName”, owlRefClass);
// 生成实例的属性
owlIndividual1.addProperty(annotationProId, id);
//生成实例之间的关系
owlIndividual1.addProperty(objectProOwn, owlIndividual2);
```

（b）生成本体

图 14　模型自动生成本体相关伪码截图

3.2　复杂产品本体自动生成模型

本体自动生成模型的过程需要读取 BFO 本体中需求、架构、追溯本体中的信息，并基于需求条目模型、架构模型和追溯矩阵的底层结构生成对应的模型。采用 Jena API 对本体的信息进行读取，并通过 Dom4j API 生成对应的模型。相关过程如图 15 所示。

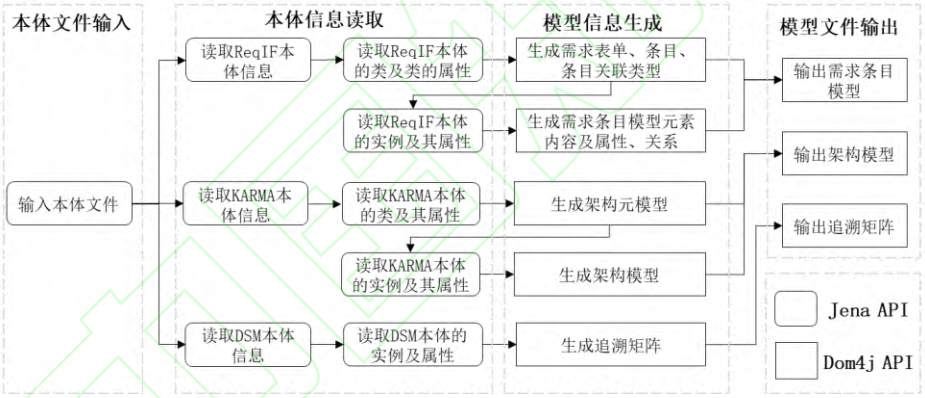

图 15　本体自动生成模型的过程

由图 15 可知，本体自动生成模型主要包括两大步骤，本体数据读取，对应模型生成。两部分伪码如下：

```
//本体自动生成模型1：读取本体数据
//创建本体Model
OntModel ontModel =
ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM);
//读取本体Model路径
ontModel.read(path);
//读取类
OntClass graph= ontModel.getOntClass("Graph");
//读取类具有的属性
String Identifier = graph.getPropertyValue(annotationProIdentifier)
.toString();
//读取类的实例
for (Iterator it = graph.listInstances(); it.hasNext();) {
//类具有的实例
Individual owlGraphModel= (Individual) it.next();
//实例具有的所有属性
for (Iterator ig = owlGraphModel.listProperties(); ig.hasNext();) {
Statement owlGraphModelPro = (Statement) ig.next();
//判断实例是否与其他实例具有 “复制”关系
if (owlGraphModelPro.getPredicate().getLocalName().equals("复制")) {
//读取与实例具有 “复制”关系的实例
Individual owlGraph = ontModel.getIndividual(owlGraphModelPro.
getResource().getURI());
//读取实例具有的属性
String ID = owlGraphIndividual.getPredicate().getLocalName();
}}}
```

（a）读取本体数据

```
//本体自动生成模型2：生成模型
//创建模型文档
Document document = DocumentHelper.createDocument();
//添加根节点
Element root = document.addElement("root");
//在根节点下添加元素
Element Specification = root.addElement("SPECIFICATION");
//给元素添加属性
root.addAttribute("DataType", "T_String32k");
//给元素添加文本内容
root.addText("ID");
//生成模型
XMLWriter writer = new XMLWriter(new FileOutputStream(new
File(fileName)), format);
writer.write(document);
writer.close();
```

（b）生成模型

图 16　本体自动生成模型相关伪码截图

4 智能电动汽车案例

采用智能电动汽车自动驾驶系统自动制动的案例来验证所提方法的有效性。案例基于文献[36]构建，以两个智能电动汽车为研究对象，以双车正常行驶过程中遇到紧急情况下的避障场景为典型场景，进行智能电动汽车的设计。图 17 描述了基于上述场景，使用 MetaGraph2.0 工具构建智能电动汽车需求、架构、追溯关系模型，以及生成一体化本体的完整工作流程。其中，通过本体转换引擎，需求、架构、追溯关系模型被转换为一体化本体框架中对应的本体类、实例及属性，用于实现需求、架构、追溯关系的语义集成，解决语义不一致问题；同时将本体中的修改对应自动生成模型，实现本体和模型的无缝集成，提高语义集成的效率。

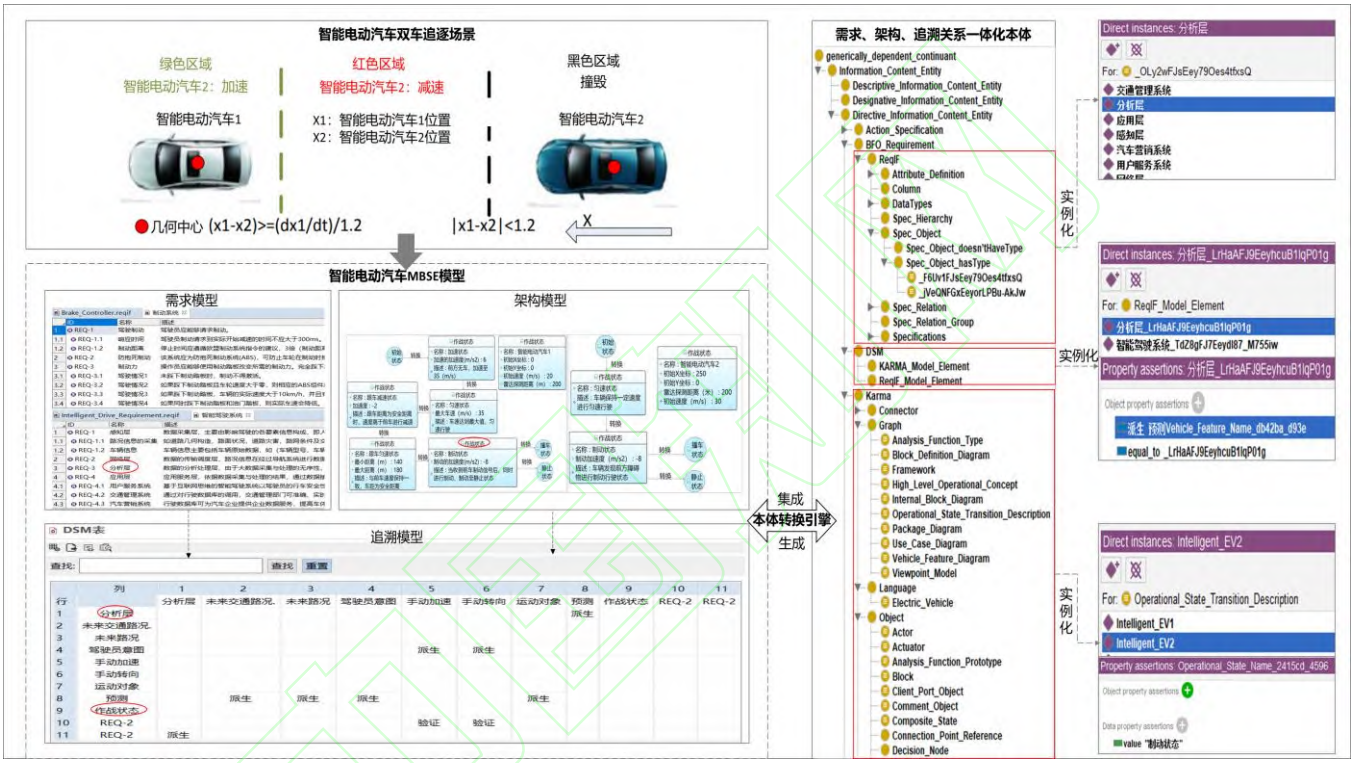

图 17　智能电动汽车需求、架构、追溯关系模型语义集成

4.1 智能电动汽车模型构建和本体生成

在设计过程中，采用需求条目模型描述涉及到的需求，采用 SysML、UML、UPDM[37]（Unified Profile for DoDAF and MoDAF）、BPMN[38]（Business Process Modeling Notation）和 EAST-ADL[39]（Architecture Description Language (ADL) for automotive embedded systems）等多种建模语言描述架构模型，采用设计结构矩阵 DSM 描述需求和架构之间的关系。基于统一建模语言 KARMA 及 MetaGraph2.0 工具，构建智能电动汽车的需求、架构、追溯关系模型，部分模型如图 18 到图 20 所示。其中，架构模型基于使命-运行-功能-逻辑-物理（Mission、Operation、Function、Logic、Physic，MOFLP）方法构建，涉及的建模语言对应图元模型及模型如表 2 所示，通过 KARMA 语言和 MetaGraph2.0 工具可以实现多种建模语言架构模型的统一构建。

表 2　涉及建模语言对应图元模型及模型

建模语言	图元模型	模型
SysML	内部模块图、模块定义图、包图、驾驶控制逻辑分析模型、驾驶控制物理接口模型、	

	用例图	驾驶控制分解结构模型、设计运行分析模型、驾驶运行分析模型、测试运行分析模型
BPMN	流程图	MOFLP 视点图
UML	包图	MOFLP 框架图
UPDM	作战状态转换图元模型、顶层作战概念图元模型	智能电动汽车 1、2 模型、作战状态转换模型、运行概念模型；使命任务概念模型
EAST-ADL	车辆特征图元模型、分析功能类型图元模型	智能电动汽车特征模型、制动控制特征模型、纵向控制特征模型、横向控制特征模型、动力控制特征模型、感知特征模型和预测特征模型；智能电动汽车功能分析模型、生成全局路径功能分析模型、制动扭矩分配和电子稳定控制功能分析模型、驾驶命令功能分析模型、横向控制功能分析模型和动力总成控制功能分析模型

Brake_Controller.reqif — 制动系统

	ID	名称	描述	Link
1	REQ-1	驾驶制动	驾驶员应能够请求制动。	2▷▷0
1.1	REQ-1.1	响应时间	驾驶员制动请求到实际开始减速的时间不应大于300ms。	0▷▷1
1.2	REQ-1.2	制动距离	停止时间应遵循欧盟制动系统指令的建议，3倍（制动距离）对于结冰是可以接受的。	0▷▷1
2	REQ-2	防抱死制动	该系统应为防抱死制动系统(ABS)，可防止车轮在制动时抱死。	
3	REQ-3	制动力	操作员应能够使用制动踏板改变所需的制动力。完全踩下踏板意味着最大的制动力。	4▷▷0
3.1	REQ-3.1	驾驶情况1	未踩下制动踏板时，制动不得激活。	0▷▷1
3.2	REQ-3.2	驾驶情况2	如果踩下制动踏板且车轮速度大于零，则相应的ABS组件施加在车轮上的制动扭矩值最终应大于0。	0▷▷1
3.3	REQ-3.3	驾驶情况3	如果踩下制动踏板，车辆的实际速度大于10km/h，并且有一个车轮打滑，则该车轮对应的制动扭矩为零。	0▷▷1
3.4	REQ-3.4	驾驶情况4	如果同时踩下制动踏板和油门踏板，则实际车速会降低。	0▷▷1

Intelligent_Drive_Requirement.reqif — 智能驾驶系统

	ID	名称	描述	Link
1	REQ-1	感知层	数据采集层，主要由影响驾驶的各要素信息构成，即人、车、路的信息采集及三者信息的相互联系与交叉影响	2▷▷0
1.1	REQ-1.1	路况信息的采集	如道路几何构造，路面状况，道路灾害，路网条件及交通状况等，一般可通过GPS或北斗系统等高精度导航系统	0▷▷1
1.2	REQ-1.2	车辆信息	车辆信息主要包括车辆原始数据，如（车辆型号，车辆理论参数等）以及车辆行驶动态数据，如（行车速度，行	0▷▷1
2	REQ-2	网络层	数据的传输调度层，路况信息在经过导航系统进行数据采集后通过报文通信的方式进行数据传输，车辆信息有C	
3	REQ-3	分析层	数据的分析处理层，由于大数据采集与处理的无序性，在已定义的函数模型下，对影响驾驶的数据进行计算处理	
4	REQ-4	应用层	应用服务层，依据数据采集与处理的结果，通过数据接口的方式可进行跨应用，跨系统之间的信息共享与信息协	3▷▷0
4.1	REQ-4.1	用户服务系统	基于互联网思维的智能驾驶系统以驾驶员的行车安全性、舒适度等为约束，通过互联网的云处理与计算平台，得	0▷▷1
4.2	REQ-4.2	交通管理系统	通过对行驶数据库的调用，交通管理部门可准确、实时地掌握的行驶状况，更好地组织、规划、协调、指挥运输	0▷▷1
4.3	REQ-4.3	汽车营销系统	行驶数据库可为汽车企业提供企业数据服务，提高车体质量，促进企业方向性的发展。	0▷▷1

图 18　智能电动汽车需求条目模型（部分）

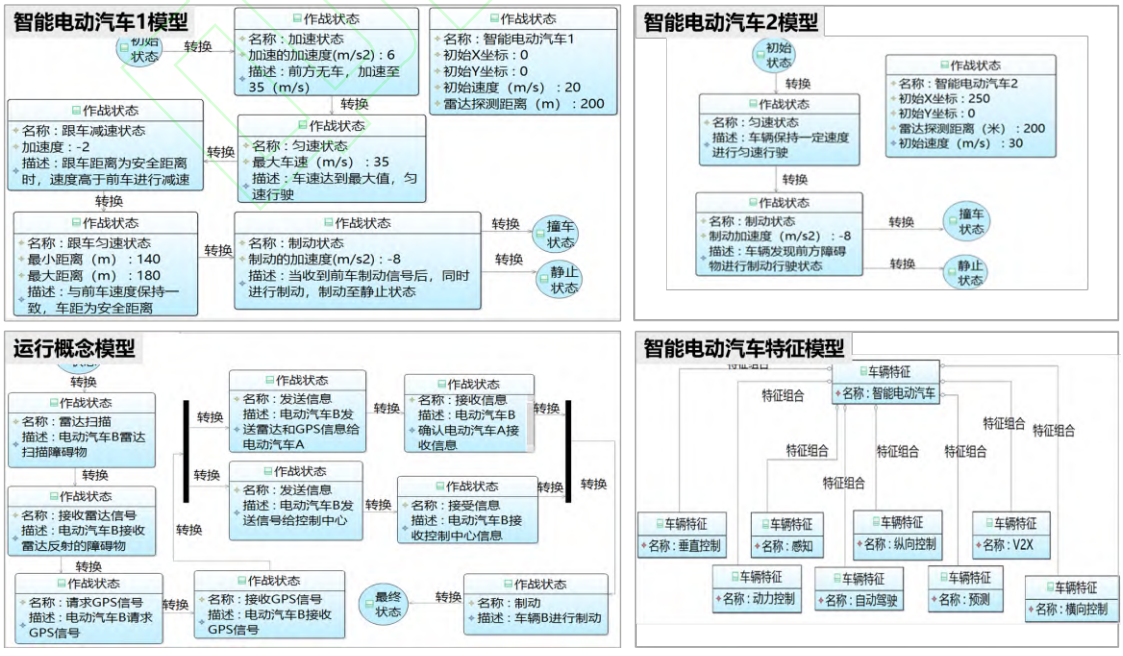

图 19　智能电动汽车架构模型（部分）


DSM表

查找: 查找 重置

行	列	1	2	3	4	5	6	7	8	9	10	11
		分析层	未来交通路况.	未来路况	驾驶员意图	手动加速	手动转向	运动对象	预测	作战状态	REQ-2	REQ-2
1	分析层								派生			
2	未来交通路况.											
3	未来路况											
4	驾驶员意图					派生	派生					
5	手动加速											
6	手动转向											
7	运动对象											
8	预测		派生	派生	派生			派生				
9	作战状态											
10	REQ-2					验证	验证					
11	REQ-2	派生										

图 20　智能电动汽车追溯矩阵（部分）（追溯关系为行到列）

通过调用在 MetaGraph2.0 工具中构建的本体转换引擎，将智能电动汽车的需求、架构、追溯关系模型自动生成 BFO 下的需求、架构、追溯关联一体化本体，如图 21 和图 22 所示。其中，模型对应的本体元素及数量如表 3 所示，智能电动汽车中的需求条目模型转换成本体中 ReqIF 类及其子类下的 123 个实例，架构模型转换成 GOPPRR-E 类下的子类及实例，DSM 追溯矩阵转换成 DSM 类及其子类下的 10 个实例，模型元素之间的关系及模型本身具有的属性对应转换成本体中的对象属性及注释属性等，实现了智能电动汽车需求、架构、追溯模型与一体化本体之间的转换；将转换生成的本体再导入到 MetaGraph2.0 工具中，生成完全相同的智能电动汽车模型，保证了生成的本体包含智能电动汽车模型的所有信息。

表 3　智能电动汽车模型对应的本体元素及数量

MBSE 模型		数量	本体	数量
需求条目模型		2	ReqIF 类及其子类下的实例	123
架构模型	图元模型	10	Graph 类下的子类	10
	对象元模型	46	Object 类下的子类	46
	属性元模型	29	Property 类下的子类	29
			Property 类的子类下的子类	109
	点元模型	17	Point 类下的子类	17
			Point 类的子类下的子类	29
	角色元模型	42	Role 类下的子类	42
	关系元模型	21	Relationship 类下的子类	21
	扩展	10	Connector 类下的子类	10
	模型	30	Graph 类下各子类的实例	30
			Object 类下各子类的实例	434
			Property 类下各子类的实例	344
			Point 类下各子类的实例	497
			Role 类下各子类的实例	1208
			Relationship 类下各子类的实例	605
			Connector 类下各子类的实例	1210
追溯矩阵 DSM		1	DSM 类及其子类下的实例	10

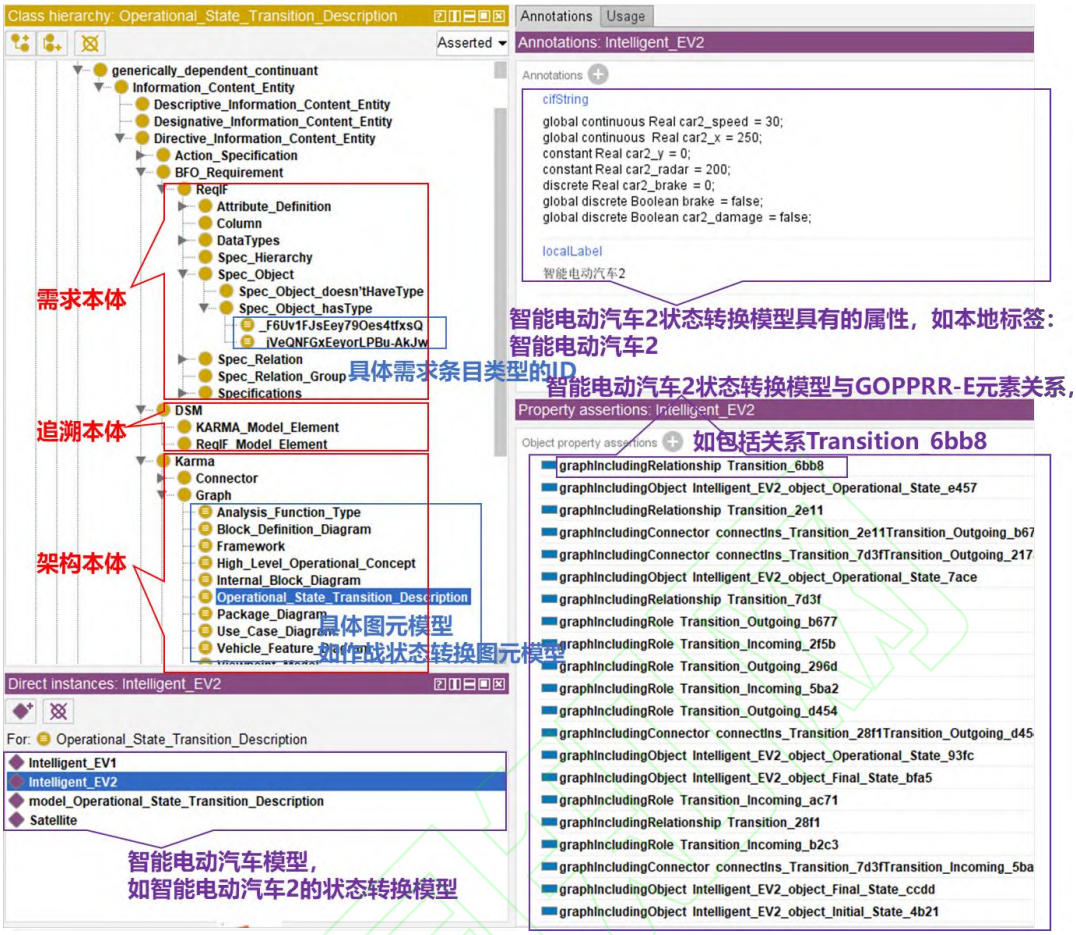

图 21　智能电动汽车需求、架构、关联追溯一体化本体

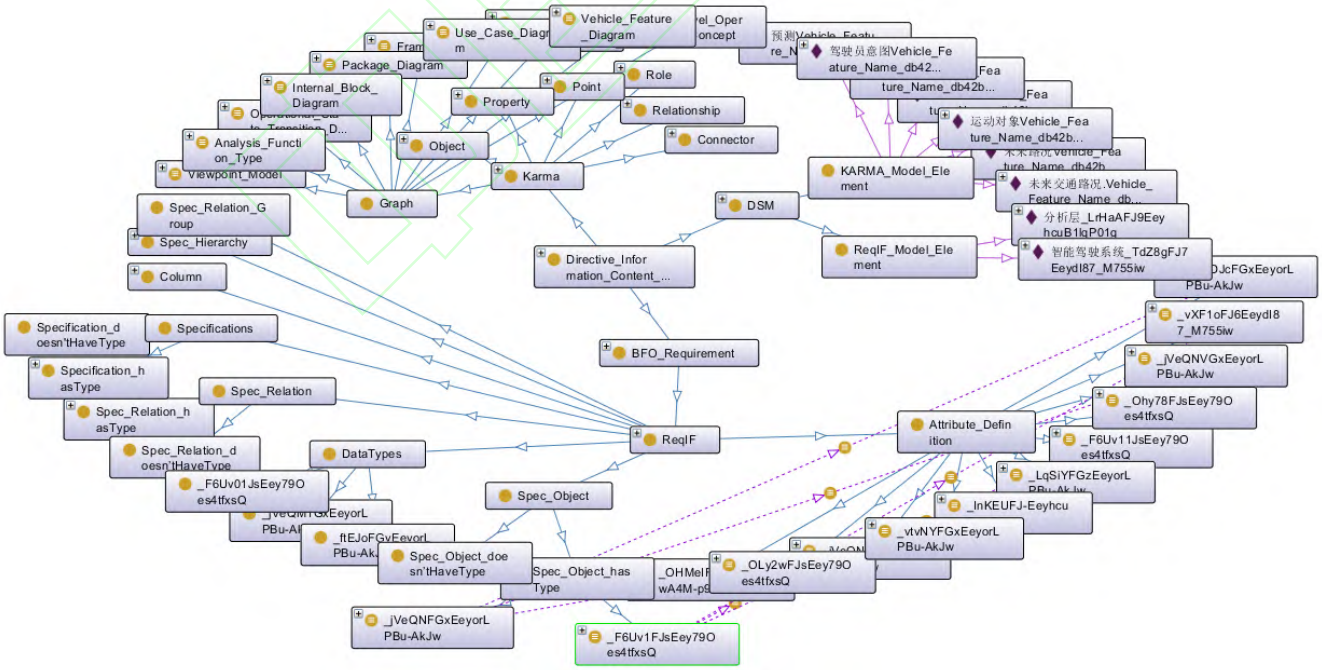

图 22　智能电动汽车需求、架构、关联追溯一体化本体可视化（部分）

4.2　定性分析

在智能电动汽车设计中，主要存在语义不一致和数据割裂两类问题，通过一体化本体可以有效解决上述问

题，具体内容如下：。

（1）语义不一致问题

由图 18 构建的智能电动汽车的需求条目模型中可以发现，制动系统和智能驾驶系统的需求条目模型中具有相同 ID 属性值的需求条目，例如 REQ-2。REQ-2 在制动系统需求条目模型和智能驾驶系统需求条目模型中具有不同的意义，制动系统中 ID 属性为 REQ-2 的需求条目是防抱死制动的需求，智能驾驶系统中 ID 属性为 REQ-2 的需求条目是网络层的需求。在采用 DSM 矩阵进行需求条目和架构模型追溯过程中，如果采用 ID 属性代表需求条目，则会出现语义不一致的现象，难以确定 REQ-2 属于哪一个需求条目，如图 20 所示，导致智能电动汽车需求追溯模糊，增加开发人员理解和协调模型一致性的时间，给设计造成不便。

通过需求、架构、追溯关系一体化本体可以有效解决上述存在的问题。一体化本体具有丰富的语义信息，通过实例、对象属性、注释属性等可以对概念进行充分地解释和区分。在生成一体化本体的过程中，基于本体生成规则，追溯矩阵 DSM 中的两个相同名称的元素“REQ-2”分别对应生成 DSM 类的子类 ReqIF_Model_Element 下的实例。“REQ-2”生成实例的注释属性“Name=REQ-2”，其底层结构 ID 生成注释属性 Identifier 的值。通过对象属性“派生”表达了该需求条目与其它元素的关联关系，如“第一个 REQ-2 和运动对象的关系是派生”，同时通过对象属性“equal_to”，将 DSM 矩阵中的“REQ-2”追溯到需求条目模型中的对应需求条目，并可获得该需求条目所有信息。由图 23 可知，第一个“REQ-2” 表示的是防抱死制动相关的需求；由图 24 可知，第二个“REQ-2”表示的是应用层相关的需求，解决了追溯矩阵 DSM 中歧义、语义不一致、混淆的问题，实现了模型的语义一致描述，保证了智能电动汽车需求和架构之间关联追溯关系的正确性。

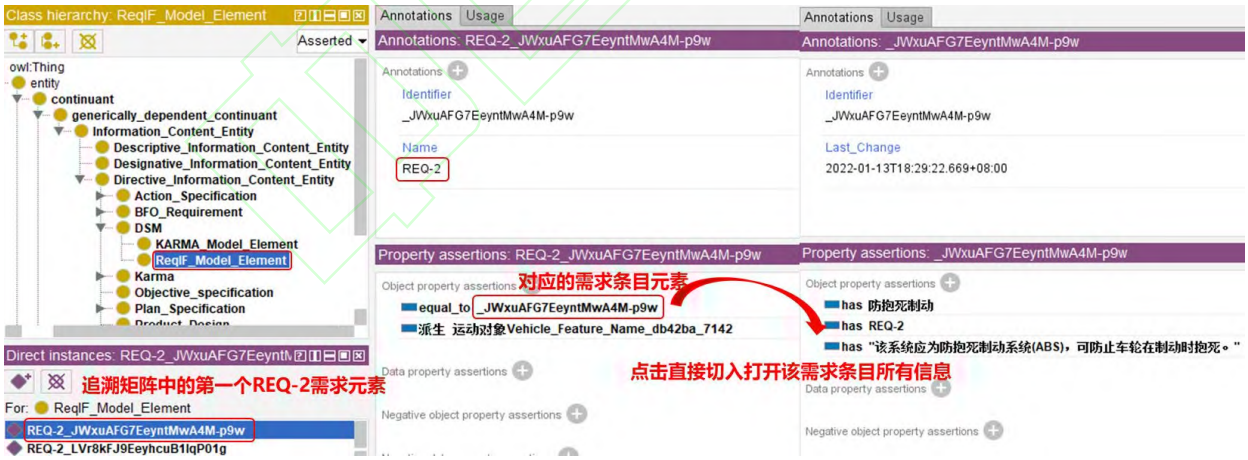

图 23　追溯矩阵第一个“REQ-2”元素对应的本体

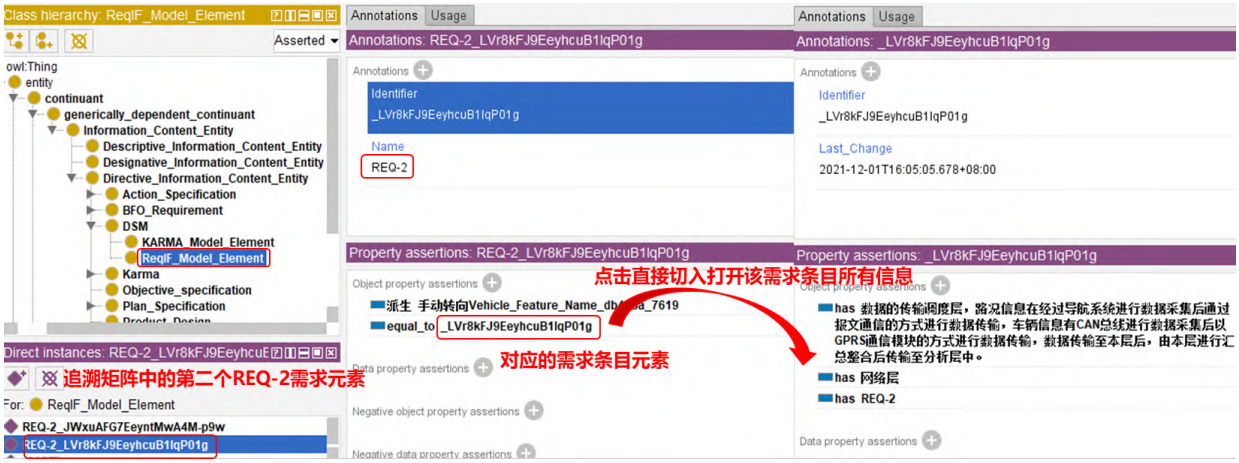

图 24　追溯矩阵第二个“REQ-2”元素对应的本体

（2）需求、架构数据割裂问题

此外，在需求分析阶段产生的如图 18 所示的智能电动汽车需求条目模型中，制动系统的制动距离根据欧盟制动系统相关标准确定；在架构设计阶段产生的如图 19 所示的智能电动汽车架构模型中，制动距离通过最大车速和制动的加速度计算得到，而这里计算得到的制动距离需要满足需求中的制动距离，因此，需求中的制动距离的数据与架构模型中的最大车速和制动加速度的数据密切相关，但二者分别存储在 ReqIF 格式文件和 KARMA 格式文件中，需求和架构之间存在数据割裂的现象，当需求中的制动距离发生变更时，架构模型难以捕捉到需求的变更，导致设计的智能电动汽车架构模型不满足制动系统的需求。

针对需求和架构数据割裂的问题，一体化本体实现了智能电动汽车需求条目模型、架构模型、追溯矩阵的语义集成，通过在本体中定义智能电动汽车需求中的制动距离与架构模型中的最大车速和制动加速度的追溯关系，实现制动距离和最大车速、制动加速度的直接关联，当制动距离需求、架构设计中的最大车速和制动加速度发生变更时，可以直接追溯到变更，如图 25 所示，消除了需求和架构数据割裂的现象，一定程度上解决了因没有捕获到需求在设计过程中发生的变更，导致的架构模型不满足制动系统需求的问题。此外，可以定义需求中的制动距离与架构中的最大车速和制动加速度之间的规则，在设计过程中基于本体推理技术快速判断架构设计是否满足需求。

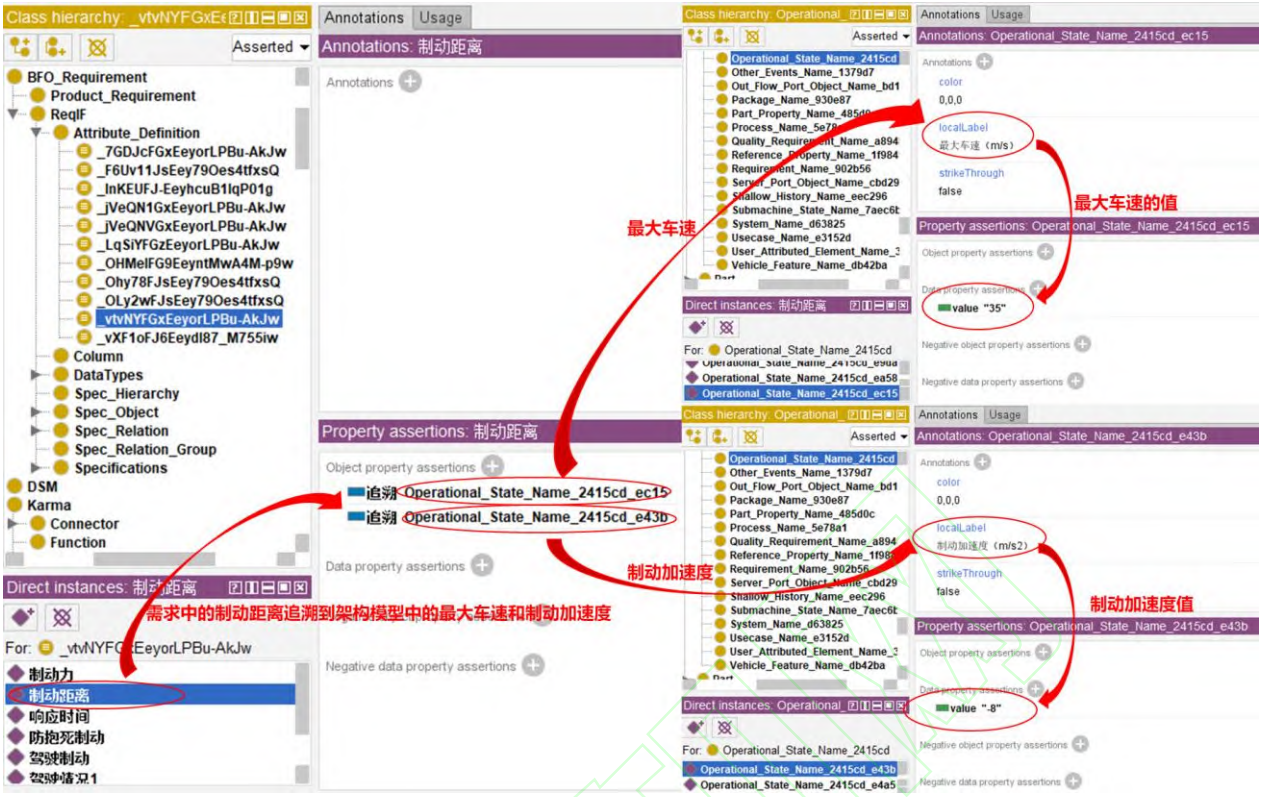

图 25　制动距离与最大车速和制动加速度的关联关系

4.3　定量分析

在智能电动汽车设计中，对所提方法的语义集成能力和语义集成效率进行定量分析，具体内容如下：。

（1）语义集成能力

采用美国佐治亚大学 LSDIS 实验室提出的 OntoQA 方法[40][41]，通过一系列度量指标从不同维度上对一体化本体语义集成能力进行量化分析。这些度量指标分为两大类：模式指标和实例指标。模式指标可以度量本体模式的丰富性、宽度、深度和继承性，包括关系丰富性、属性丰富性、继承丰富性三个子指标。实例指标可以反映本体设计的有效性和由本体表示的知识量，分为两类：知识库指标，将知识库作为一个整体来描述；类指标，描述每个类在知识库中的使用方式。类指标对语义集成能力的影响较小，因此在量化分析中不考虑类指标。

采用 OntoQA 方法对一体化本体语义集成能力进行定量分析的指标内容和指标结果如表 4 所示，体现了一体化本体具有良好的语义集成能力，为消除语义不一致、数据割裂问题奠定基础。

表 4　一体化本体语义集成能力定量分析

指标	指标内涵	评价公式	计算数值	评价结果
模式指标				
关系丰富性	反映关系的多样性。包含除类-子类关系之外的许多关系的本体比只有类-子类关系的本体更丰富。	RR=|P|/(|SC|+|P|) RR：关系丰富性 |P|：继承关系数 SC：P 继承关系外的关系数	RR=644/(644+486)=0.57	除继承关系外的关系数大于继承关系数，说明一体化本体具有丰富的关系，充分支持智能电动汽车的关系表达。

属性丰富性	反映本体包含的领域知识。通常情况下，定义的属性越多本体可以传达的知识越多。	AR=| att |/|C| AR：属性丰富性 |C|：类的数量 |att|：类具有的属性的数量	AR=651/488=1.334	由于 BFO 本体类框架中的类属性尚未添加，属性集中在集成的领域本体中，说明一体化本体具有丰富的属性，充分支持智能电动汽车的属性表达。
继承丰富性	描述本体层次结构中不同层次的信息分布，可以表达知识是如何组织到本体中不同的类和子类。低继承丰富性反映了本体的垂直性质，即详细的知识类型；高继承丰富性反映了本体的水平性质，即广泛的一般的知识。	$IR_S=\sum\|H^c(C_1,C_i)\|/\|C\|$ IR_S：继承的丰富性 |C|：类的数量 $\|H^c(C_1,C_i)\|$代表类 Ci 具有的子类数量	IR_S=644/488=1.320	$IR_S>1$，一体化本体具有丰富的层级结构，充分支持智能电动汽车中详细知识类型的表达。
知识库指标				
类丰富性	描述实例如何跨类分布。	CR=|C'|/|C| CR：类丰富度 |C'|：有实例的类的数量 |C|：本体中类的总数	CR=112/488=0.230	由于在智能电动汽车元模型设计中，有些元模型并未被使用，导致 CR 值偏低，一体化本体结构整体分布良好。
平均分布	如果本体开发者不确定与类的数量相比是否提取了足够多的实例，可以据此进行分析。	P=|I|/|C| P：知识库中类的平均分布 I：实例的数量 C：类的数量	P=4492/488=9.205	一体化本体充分提取了智能电动汽车中的需求、架构、追溯模型元素。
聚合度	如果实例和它们之间的关系被认为是一个图，其中节点表示实例，边表示它们之间的关系，聚合度可以用以反映实例的连接紧密情况。	Coh=|SCC| Coh：聚合度 |SCC|：分离连接的组件（separate connected components，SCC）的数量	Coh=1	一体化本体具有良好的集成能力，一定程度上消除了“信息孤岛”问题。

（2）语义集成效率

为了评估所提方法的语义集成效率情况，选取 4 名熟悉 MetaGraph2.0 建模与本体构建的研究者，参与关于智能电动汽车系统设计迭代过程中模型变更对应的本体变更的实验。通过对智能电动汽车需求、架构和追溯关系模型以及基于 BFO 的需求、架构、追溯关系一体化本体的详细介绍和演示，向每个参与者清楚地解释了

当前设计、变更需求和任务的问题。然后在没有作者干预的情况下，4 名研究者完成智能电动汽车系统设计模型变更中对应的本体变更。实验设置如下：

1）选择工具：智能电动汽车的需求、架构、追溯关系模型采用 MetaGraph2.0 工具构建，需求、架构、追溯关系一体化本体采用 Protégé 工具构建。

2）指定变更需求和任务：提供了智能电动汽车系统的初始以及第一次变更后的需求模型、架构模型和追溯关系矩阵。第一次变更需求为：1）智能电动汽车需求模型变更，将智能驾驶需求模型中的“感知层”修改为“数据采集层”，“网络层”修改为“数据传输调度层”，“应用层”修改为“应用服务层”。2）智能电动汽车架构模型变更，将智能电动汽车 1 模型中的加速的加速度由 6m/s^2 修改为 8m/s^2，最大车速由 35m/s 修改为 45m/s，减速的加速度由-2m/s^2 修改为-5m/s^2，制动的加速度由-8m/s^2 修改为-10m/s^2，跟车匀速状态的最小距离由 140m 修改为 160m，最大距离由 180m 修改为 200m，雷达探测距离由 200m 改为 150m，初始速度由 20m/s 改为 15m/s，智能电动汽车 2 模型中的初始速度由 30m/s 改为 35m/s，制动的加速度由-8m/s^2 修改为-10m/s^2。3）智能电动汽车追溯关系模型变更，为智能电动汽车追溯矩阵中的“手动加速”和“手动转向”之间添加满足关系，删除“预测”和“未来交通路况”、“未来路况”之间的派生关系。第一次变更任务为：实现上述变更对应的一体化本体变更。

3）选择评价标准：以变更完成时间为评价标准。

4）将参与者分组：参与者被分成四组，A 组、B 组、C 组和 D 组。A、B、C、D 四组的研究者分别使用 Protégé 本体工具手动修改模型变更对应的一体化本体，以及通过本体转换引擎自动生成模型变更对应的一体化本体。每组实验进行 3 次，以排除偶然因素。

5）实验条件：使用 Protégé 本体工具手动修改本体的时间不包括 Protégé 工具打开和加载的时间；使用本体转换引擎自动生成变更本体的时间不包括工具 MetaGraph2.0 工具打开和加载的时间；四位研究者对 MetaGraph2.0 工具建模及 Protégé 工具构建本体的熟悉度基本一致，四位研究者使用电脑软硬件配置一致。

四组研究者完成本体变更所花费的时间被用来评估所提方法的语义集成效率，如图 26 所示。其中，A、B、C、D 四组成员手动修改一体化本体的时间，均远长于通过本体转换引擎自动生成变更对应的一体化本体所需时间。并且手动修改的本体难以保证修改后本体的正确性。实验结果证实，在复杂系统设计迭代过程中，所提方法可以在更短的时间内以更高的效率实现系统模型的语义集成。

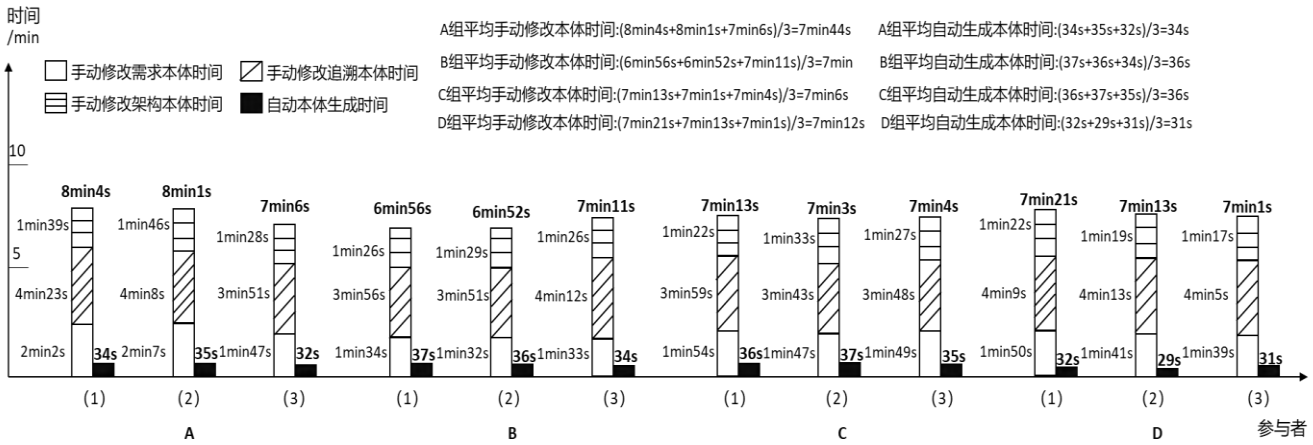

图 26　对应的本体变更时间

5　结束语

本文提出一种基于顶层本体 BFO 的需求、架构和追溯关系的一体化描述和集成的方法。该方法在构建需求领域本体、架构领域本体和追溯领域本体的基础上，通过顶层本体 BFO 对该三类领域本体进行集成，形成一套集成的通用本体框架。同时，构建本体转换引擎实现模型和集成本体的自动转换。实现了需求条目模型、架构模型、追溯矩阵的语义集成和一致描述，并为不同领域的建模工具之间的互操作提供一个通用的规范。通过智能电动汽车案例，验证了所提方法框架具有统一描述需求、架构及追溯关系的能力，确保需求和架构之间的语义一致性、关联追溯关系的正确性，减少开发人员理解和协调模型不一致性的时间，简化模型之间的交互过程，一定程度上解决设计的架构不满足需求的问题。该方法的应用，提高了智能电动汽车架构设计的效率和质量。

未来工作中，在通过本体解决模型中存在的语义不一致问题的同时，研究将语义不一致问题反馈到模型，同步解决模型中本身存在的语义不一致问题。并且，基于集成的一体化本体，进一步构建推理查询规则，将需求、架构、关联追溯领域的知识表示为一组逻辑规则，使得计算机能够更好地理解和管理知识，实现设计信息补全和纠错，确保需求、架构、关联追溯知识的准确性、一致性和重用性。

参考文献：

[1] International Council on Systems Engineering (INCOSE). Systems engineering vision 2020[R]. Seattle, WA: International Council on Systems Engineering, 2007: 5-32.

[2] ELAKRAMINE F, JARADAT R, HOSSAIN N U I, et al. Applying systems modeling language in an aviation maintenance system[J]. IEEE Transactions on Engineering Management, 2021, 69(6): 4006-4018.

[3] SPANGELO S C, KASLOW D, DELP C, et al. Applying model based systems engineering (MBSE) to a standard CubeSat[C]//2012 IEEE aerospace conference. IEEE, 2012: 1-20.

[4] HERZIG S J I. SysML and beyond: applications of MBSE at the NASA Jet Propulsion Laboratory[J]. 2018.

[5] FOSSE E. MBSE for Mars2020[J]. 2021.

[6] REED D, SIMPSON K. Orion flight test architecture benefits of MBSE approach[J]. 2012.

[7] BAYER T. Progress check: how well is the planned Europa's Mission's MBSE application addressing systems engineering challenges?[J]. 2017.

[8] DONAHUE K, KADESCH A, KHAN O, et al. MBSE in Development: SMAP Pilot Project[J]. 2012.

[9] WANG Haoqi, LI Hao, WEN Xiaoyu, et al. Reuse framework method of ontology-driven system design information[J].Computer Integrated Manufacturing Systems, 2021, 27(6): 1662 (in Chinese).[王昊琪, 李浩, 文笑雨, 等. 本体驱动的复杂产品系统设计信息重用框架[J]. 计算机集成制造系统, 2021, 27(6): 1662.]

[10] ZHANG Yujin, HUANG Bo, LIAO Wenhe. MBSE unified modeling and design method of commercial aeroengine for operation scenario[J]. Computer Integrated Manufacturing Systems, 2021, 27(11): 3093 (in Chinese).[张玉金, 黄博, 廖文和. 面向场景的航空发动机基于模型的系统工程设计[J]. 计算机集成制造系统, 2021, 27(11): 3093.]

[11] CAPASSO C, HAMMADI M, Patalano S, et al. A multi-domain modelling and verification procedure within MBSE approach to design propulsion systems for road electric vehicles[J]. Mechanics & Industry, 2017, 18(1): 107.

[12] DRAXLER D, NEUREITER C, LASTRO G, et al. A domain specific systems engineering framework for modelling electric vehicle architectures[C]//2019 IEEE Transportation Electrification Conference and Expo, Asia-Pacific (ITEC Asia-Pacific). IEEE, 2019: 1-6.

[13] BINDSCHADLER D L, SMITH R R,VALEIO C P,et al. A Structured, Model-Based System Engineering Methodology for Operations System Design[J]. Space Operations: Contributions from the Global Community, 2017: 271-290.

[14] ALLEN J L. An overview of model-based development verification/validation processes and technologies in the aerospace industry[C]//AIAA Modeling and Simulation Technologies Conference. 2016: 1922.

[15] WANG L, IZYGON M, OKON S, et al. Effort to accelerate MBSE adoption and usage at JSC[M]//AIAA SPACE 2016. 2016: 5542.

[16] GERACI A.IEEE standard computer dictionary: Compilation of IEEE standard computer glossaries[M].IEEE Press, 1991.

[17] WANG Wenyue, HOU Junjie, MAO Yanxuan, et al. Research on MBSE Architecture for Complex Product Development and Trends[J]. Control and Decision, 2022: 3073-3082 (in Chinese) . [王文跃, 侯俊杰, 毛寅轩, 等. 面向复杂产品研制的 MBSE 体系架构及其发展趋势研究[J]. 控制与决策, 2022: 3073-3082.]

[18] WANG Wei. Research on Library Data Integration Based on Ontology[J]. Information Research,2012(02):93-96 (in Chinese) . [王蔚.基于本体的图书馆数据集成研究[J].情报探索,2012(02):93-96.]

[19] YU Qianfan. Chinese Terms in Computer Science and Technology (Third Edition) officially published[J]. China Terminology,2019,21(2):1 (in Chinese).[余前帆.《计算机科学技术名词》(第三版)正式公布[J]. 中国科技术语, 2019, 21(2):1.]

[20] CALHAU R F, DE ALMEIDA FALBO R. An ontology-based approach for semantic integration[C]//2010 14th IEEE International Enterprise Distributed Object Computing Conference. IEEE, 2010: 111-120.

[21] STUDER R, BENJAMINS V R, FENSEL D. Knowledge engineering: Principles and methods[J]. Data & knowledge engineering, 1998, 25(1-2): 161-197.

[22] LIN J, FOX M S, BILGIC T. A requirement ontology for engineering design[J]. Concurrent Engineering, 1996, 4(3): 279-291.

[23] RUIJVEN V, L. C. Ontology for Systems Engineering as a base for MBSE[J]. INCOSE International Symposium, 2015, 25(1):250-265.

[24] DUPREZ J, ERNADOTE D.Towards a semantic approach of MBSE frameworks specification[J]. INCOSE International Symposium, 2020, 30(1):1405-1419.

[25] WANG Haoqi, LI Hao, WEN Xiaoyu. Ontology-based Axiomatic System Design Semantic Modeling and Reasoning Rules[J]. Journal of Mechanical Engineering,2021,57(05):205-221 (in Chinese) . [王昊琪,李浩,文笑雨.基于本体的公理化系统设计语义建模与推理规则[J].机械工程学报,2021,57(05):205-221.]

[26] CHEN W. The Research and Application of Requirement Elicitation Technique in Domain-specific based on Ontology[D]. North China Electric Power University(HeBei),2008 (in Chinese). [陈伟. 基于本体的特定领域需求获取技术研究及应用[D].华北电力大学（河北）,2008.]

[27] LUO Junli. Research on Modeling Method of Manufacturing Resources Based on Ontology[J].Software Guide,2016,15(08):4-6 (in Chinese) . [罗俊丽.基于本体的制造资源建模方法研究[J].软件导刊,2016,15(08):4-6.]

[28] NOYER A, IYENGHAR P, PULVERMUELLER E, et al. Traceability and interfacing between requirements engineering and UML domains using the standardized ReqIF format[C]//2015 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD). IEEE, 2015: 1-6.

[29] SUN Shengnan, LU Jinzhi, CHEN Jinwei,et al. Implementation and application of multi-architecture modeling method for aircraft entertainment system[J]. Science & Technology Review, 2020, 38(21):10 (in Chinese) . [孙胜楠、鲁金直、陈金伟等. 多架构建模方法在飞机娱乐系统的实现及应用[J]. 科技导报, 2020, 38(21):10.]

[30] Kern H, Hummel A, Kühne S. Towards a comparative analysis of meta-metamodels[C]//Proceedings of the compilation of the co-located workshops on DSM'11, TMC'11, AGERE! 2011, AOOPES'11, NEAT'11, & VMIL'11. 2011: 7-12.

[31] LU J, MA J, ZHENG X , et al. Design Ontology Supporting Model- based Systems-engineering Formalisms [J]. 2020.

[32] LU J, WANG G, MA J, et al. General Modeling Language to Support Model‐based Systems Engineering Formalisms (Part 1)[C]//INCOSE International Symposium. 2020, 30(1): 323-338.

[33] XU Luning, ZHANG Heming, ZHANG Yongkang. Multidisciplinary Cooperative Design With DSM(Design Structure Matrix) [J].China Mechanical Engineering,2005,(12):1035-1038. [徐路宁,张和明,张永康.基于设计结构矩阵的多领域协同设计[J].中国机械工程,2005,(12):1035-1038.]

[34] ZHU Ling, DONG Yan, YANG Feng. Research Progress of Basic Formal Ontology[J].Chinese Journal of Experimental Traditional Medical Formulae,2018,24(02):208-212 (in Chinese) . [朱玲,董燕,杨峰.基本形式化本体的研究进展[J].中国实验方剂学杂志,2018,24(02):208-212.]

[35] ZHU Yan, ZHENG Jie, LI Xiaoying, et al. Introduction to Basic Formal Ontology and Its Chinese Version[J]. Journal of Medical Informatics,2021,42(01):24-28+60(in Chinese) . [朱彦,郑捷,李晓瑛等.基本形式化本体及其中文版介绍[J].医学信息学杂志,2021,42(01):24-28+60.]

[36] Lu J, Chen D, Törngren M, et al. A model-driven and tool-integration framework for whole vehicle co-simulation environments[C]//8th European Congress on Embedded Real Time Software and Systems (ERTS 2016). 2016.

[37] HAUSE M. The Unified Profile for DoDAF/MODAF (UPDM) enabling systems of systems on many levels[C]//2010 IEEE international systems conference. IEEE, 2010: 426-431.

[38] CHINOSI M, TROMBETTA A. BPMN: An introduction to the standard[J]. Computer Standards & Interfaces, 2012, 34(1): 124-134.

[39] SIRGABSOU Y, BARON C, PAHUN L, et al. Model-driven engineering to ensure automotive embedded software safety. Methodological proposal and case study[J]. Computers in Industry, 2022, 138: 103636.

[40] Tartir S, Arpinar I B, Moore M, et al. OntoQA: Metric-based ontology quality analysis[J]. 2005.

[41] Tartir S, Arpinar I B. Ontology evaluation and ranking using OntoQA[C]//International conference on semantic computing (ICSC 2007). IEEE, 2007: 185-192.

作者简介：

董梦如（1999-），女，安徽六安人，硕士研究生，研究方向：基于模型的系统工程，E-mail：3120210371@bit.edu.cn；

王国新（1977-），男，黑龙江佳木斯人，教授，博士，研究方向：系统工程、体系工程、数字孪生、可重构设计等，E-mail：wangguoxin@bit.edu.cn；

+鲁金直（1988-），男，辽宁人，洛桑联邦理工学院博士后研究员，博士，研究方向：系统工程和基于模型的系统工程，认知孪生和数字孪生，面向服务的工具链技术等，E-mail：jinzhi.lu@epfl.ch；

阎 艳（1967-），女，重庆人，教授，博士，研究方向：知识工程、系统工程、数字孪生等，E-mail：yanyan@bit.edu.cn。