

ICS 35.080
CCS L 77



中华人民共和国国家标准

GB/T 43270—2023

复杂产品协同设计集成建模 语言 X 语言架构

Integrated modeling language X language architecture for complex
product collaborative design

2023-11-27 发布

2024-06-01 实施

国家市场监督管理总局
国家标准化管理委员会 发布

目 次

前言	III
1 范围	1
2 规范性引用文件	1
3 术语和定义	1
4 缩略语	4
5 X 语言体系结构	4
6 语法结构	7
6.1 类	7
6.1.1 概述	7
6.1.2 连续类	7
6.1.3 离散类	8
6.1.4 颗粒类	9
6.1.5 智能体类	10
6.1.6 记录类	10
6.1.7 函数类	11
6.1.8 连接器类	11
6.2 图形建模	12
6.2.1 概述	12
6.2.2 定义图	12
6.2.3 连接图	13
6.2.4 方程图	13
6.2.5 状态机图	14
6.2.6 活动图	15
6.2.7 需求图	16
6.2.8 用例图	17

前　　言

本文件按照 GB/T 1.1—2020《标准化工作导则 第 1 部分：标准化文件的结构和起草规则》的规定起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别专利的责任。

本文件由中国机械工业联合会提出。

本文件由全国自动化系统与集成标准化技术委员会(SAC/TC 159)归口。

本文件起草单位：北京航空航天大学、华如科技股份有限公司、北京机械工业自动化研究所有限公司、吉林大学、北京信息科技大学、北京仿真中心、清华大学、北京临近空间飞行器系统工程研究所、中国船舶工业系统工程研究院、哈尔滨工业大学、国家工业信息安全发展研究中心。

本文件主要起草人：张霖、古鹏飞、谢塑钰、杜已超、张雪松、陈敏杰、叶飞、尹作重、赵淳、赖李媛君、施国强、任磊、闫飞、林廷宇、张柯、李君、王霄汉、王昆玉、陈真、张和明、铁鸣、罗永亮、李伟、刘敬、窦克勤。

复杂产品协同设计集成建模

语言 X 语言架构

1 范围

本文件规定了复杂产品协同设计集成建模语言 X 语言的体系结构、语法结构以及相关语法描述等方面的具体要求。

本文件适用于复杂产品协同设计领域的模型构建,适用于面向 MBSE 的全系统、全流程、多视角的一体化建模和仿真。

2 规范性引用文件

本文件没有规范性引用文件。

3 术语和定义

下列术语和定义适用于本文件。

3.1

基础类 basic class

X 语言的一种基类,支持泛化成特定的类以描述不同类型的模型。

3.2

受限类 restricted class

从基础类中继承某一部分功能的类的统称。

3.3

连续类 continuous class

X 语言中的一种受限类,用于描述复杂产品中基于方程定义的连续模型或连续行为。

3.4

离散类 discrete class

X 语言中的一种受限类,用于描述复杂产品中基于状态描述的离散模型或离散行为。

3.5

耦合类 couple class

X 语言中的一种受限类,用于描述复杂产品中多领域、多特征、多层次系统模型间的耦合关系。

3.6

智能体类 agent class

X 语言中的一种受限类,用于描述复杂产品中具有交互和学习行为的智能体模型。

3.7

记录类 record class

X 语言中的一种受限类,用来描述复杂产品中各模型中涉及的复杂数据结构。

3.8

函数类 function class

X 语言中的一种受限类,用于描述复杂产品各模型中涉及的功能行为模块。

3.9

连接器类 connector class

X 语言中的一种受限类,用于描述复杂产品中连续、离散、智能体等模型的内外数据接口及发送接受的数据类型。

3.10

定义图 definition diagram

用于显示不同类型模型元素和关系以说明系统结构信息的图。

3.11

方程图 equation diagram

用于描述连续类模型行为的图。

3.12

连接图 connection diagram

用于显示单个模块内部系统结构的图。

3.13

状态机图 state machine diagram

描述原子模型的一种动态行为图,主要关注的是系统的结构如何依据随时间发生的事件改变状态。

3.14

需求图 requirement diagram

描述利益相关者对设计制品的系统需求的图,主要关注需求之间以及需求和设计制品之间的关系。

3.15

用例图 use case diagram

描述系统提供的服务信息,以及需要服务的利益相关者的信息的一种黑盒视图。

3.16



活动图 action diagram

描述系统黑盒活动流、智能体模型以及函数相关的一种行为图。

注: 主要关注控制流程,以及输入通过一系列动作转换为输出的过程。

3.17

组成元素 part element

表明了一种“所属”关系,由组成部分元素所映射的模块组成。

注: “所属”关系包含物理层面的所属、逻辑的所属、以及组成部分属性一次只能属于一个复杂结构,但可以移除。

3.18

导入元素 import element

被导入模型在导入模型中进行实例化或者调用后的元素。

3.19

参数元素 parameter element

模型的一种结构特性,主要描述模型的实例化常数。

3.20

值元素 value element

模型的一种结构特性,主要描述模型的状态变量。

3.21

端口元素 port element

模型结构边缘不同交互点的一种属性,外部实体可通过交互点进行事件、能量、数据等交互。

注:端口的类型包括事件端口、物理端口和智能算法端口。

3.22

输出方程 output equation

描述模型输出行为的方程。

3.23

状态 state

离散模型处于各转化临界点时的形态。

3.24

外部事件 external event

模型外部输入的事件触发标识。

3.25

内部事件 internal event

模型内部事件触发标识。

3.26

外部事件行为 external event behavior

模型外部输入事件触发时所执行的行为。

3.27

外部事件方程 external event equation

描述原子模型接受外部事件行为的方程。

3.28

状态转移函数 internal event triggers transfer function

离散模型内部事件或外部事件触发进行状态转移的函数。



3.29

内部事件行为 internal event behavior

模型内部事件触发时所执行的行为。

3.30

输出 output

模型内部事件触发时执行输出的标识。

3.31

输出行为 output behavior

模型内部事件触发时的输出行为。

3.32

状态持续时间 state duration

原子模型在某一状态持续的时长。

3.33

状态持续函数 state persistence function

定义离散模型处于某一状态下的持续时间的函数。

3.34

信息发送函数 message sending function

智能体模型信息发送函数。

3.35

信息接受函数 information acceptance function

智能体模型信息接受函数。

3.36

信息 message

智能体模型之间交互的信息。

3.37

计划 plan

表示智能体模型的动作集合,用于实现各类行为。

4 缩略语

下列缩略语适用于本文件。

ACT:活动图(Activity Diagram)

BDD:块定义图(Block Definition Diagram)

DEVS:离散事件系统规范(Discrete Event System Specification)

IBD:内部模块图(Internal Block Diagram)

MBSE: 基于模型的系统工程(Model Based System Engineering)

PAR:参数图(Parameter Diagram)

REQ:需求图(Requirement Diagram)

STA:状态机(State Machine Diagram)

UC:用例图(Use Case Diagram)

5 X 语言体系结构

X 语言结合了 DEVS 规范和 Modelica 语义语法,对已有的 SysML 元素进行修订和扩展,实现系统的一体化建模,如图 1 所示。



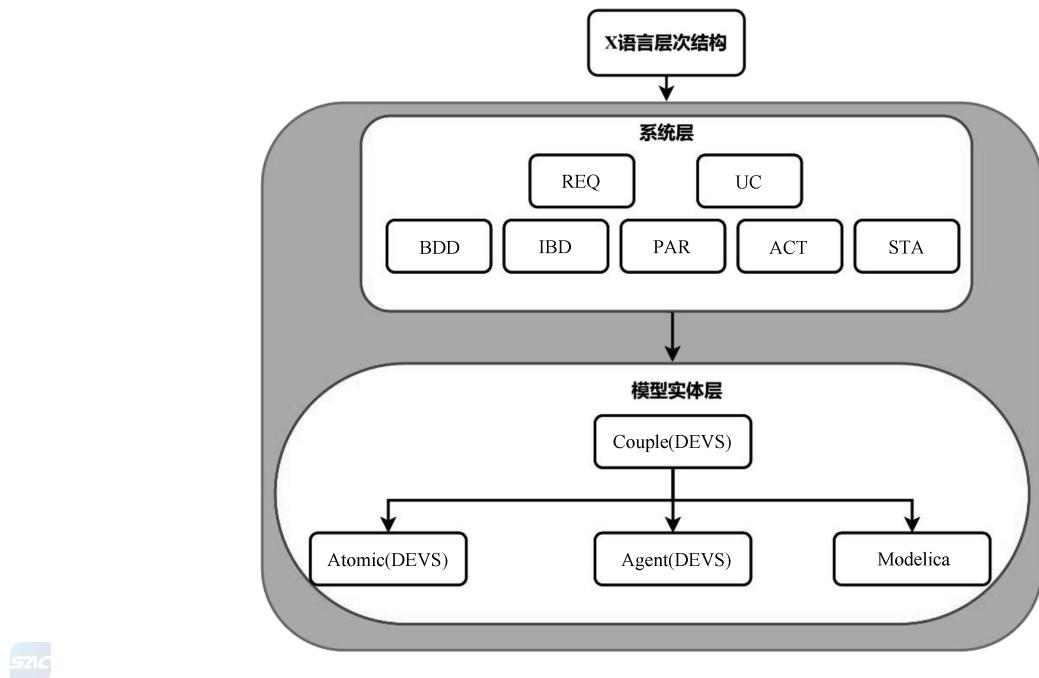


图 1 X 语言层次结构图

在图形建模层面,X语言参考SysML各类图的建模思想,清晰准确地表达系统的结构和行为;在模型实体层面,借助扩展的DEVS仿真框架(XDEVS),将离散模型、BDI智能体模型以及Modelica的连续模型统一于DEVS的耦合模型之下。基于此,X语言具有以下建模仿真能力:

- 支持图形和文本两种建模形式,并能实现图形与文本之间的相互转换,如图2所示;
- 支持系统级结构与物理行为的描述以及仿真验证能力;
- 支持对各类复杂智能体模型进行建模,包括智能体的学习过程、通信过程,以及多智能体的并行仿真过程;
- 支持连续、离散和连续/离散混合仿真。

X语言是一个支持MBSE的、面向对象的建模语言,能够提供对系统设计进行全流程验证的能力。基于图形或文本设计的模型,可以直接经由X语言提供的解释器解释为可仿真的XDEVS代码。底层的仿真器是基于XDEVS设计的多领域仿真器,能够提供连续、离散和智能体等多个领域的跨域建模。通过仿真得到的结果能够直接反馈给模型设计者,实现对系统设计的功能验证。

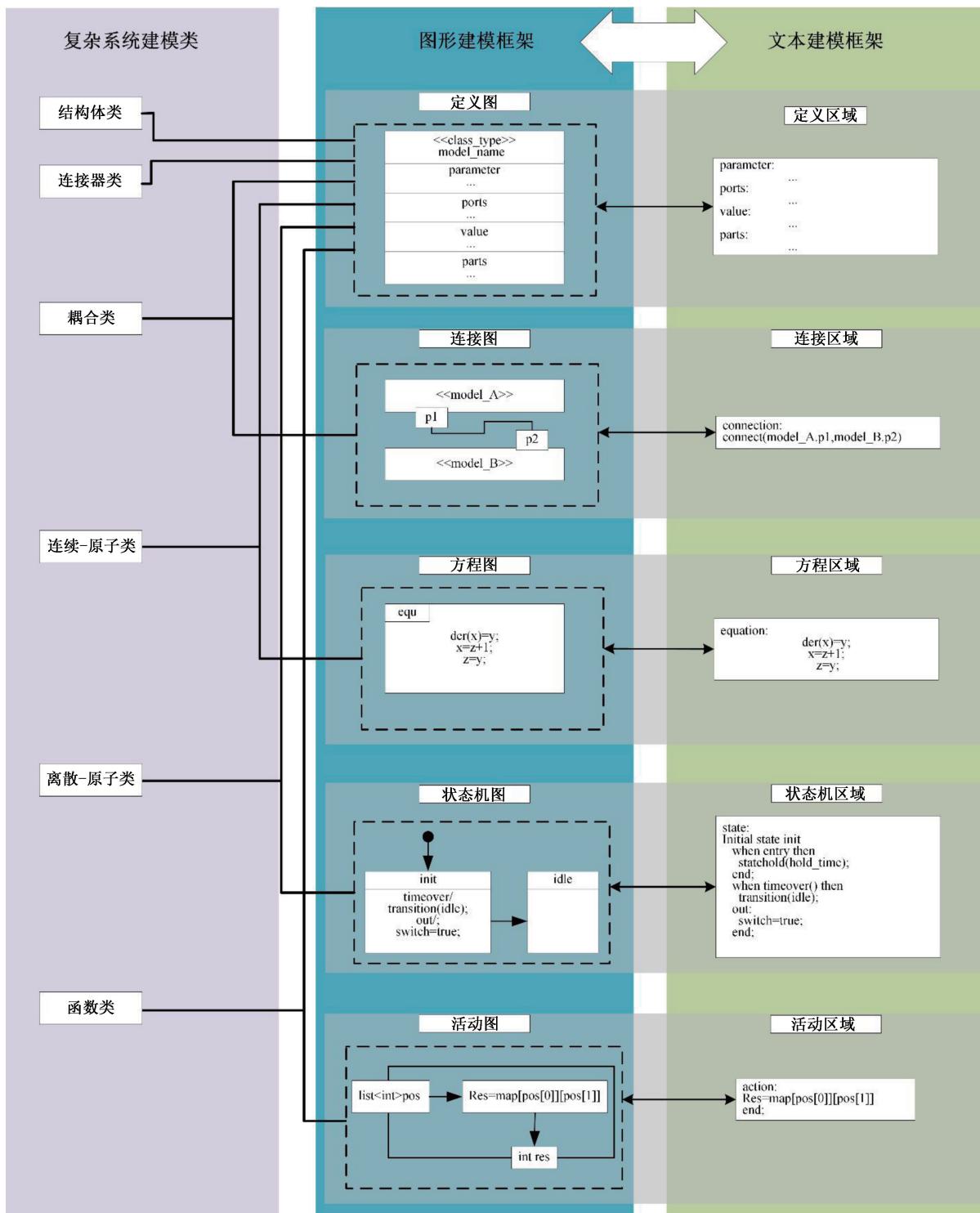


图 2 X 语言图文建模对应关系

6 语法结构

6.1 类

6.1.1 概述

类是 X 语言建模规范的基本结构元素,是构成模型的基本单元。类的架构图如图 3 所示,类分为基础类和受限类。

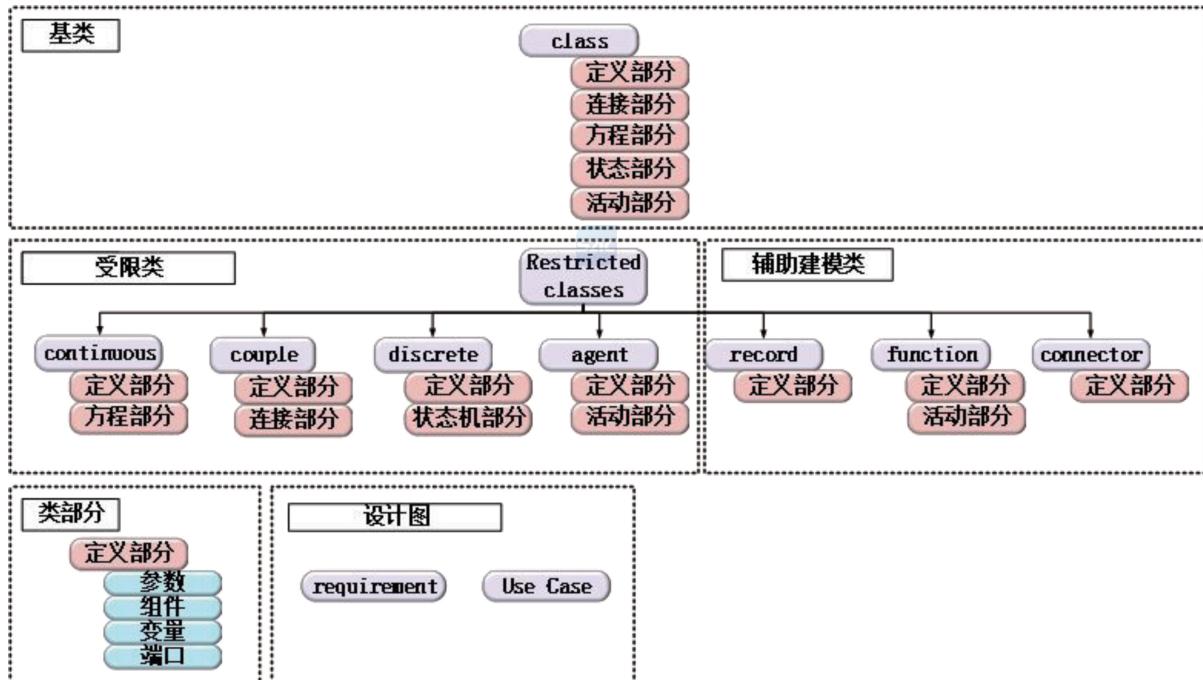


图 3 类的架构图

基础类由 definition 部分、connection 部分、equation 部分、state 部分和 active 五个部分构成;受限类具有特殊用途,在语法规规范上有一定约束,比如 discrete 类只能包括 definition 部分和 state 部分,只用于描述基础原子模型(离散)、couple 类只能包括 definition 部分和 connection 部分,只用于描述耦合模型。基础类由关键字 class 修饰,受限类由特定的关键字修饰,该建模规范中,一共有七大受限类,分别是 continuous、discrete、couple、agent、record、function 和 connector。从建模层面看,七大受限类是复杂产品建模的核心对象。

6.1.2 连续类

连续类(continuous)是用来描述复杂产品中基于方程定义的连续模型(连续行为)。一般地,continuous 类常包含头部部分、定义部分、方程部分。头部部分包括导入外部模型(结构关键字 import)以及继承外部模型(结构关键字 extends)两部分内容。定义部分用来初始化参数和变量的值,以及相关组件和输入输出端口;方程部分通过方程组的形式描述连续模型的行为。其描述的图文建模架构如图 4 所示。

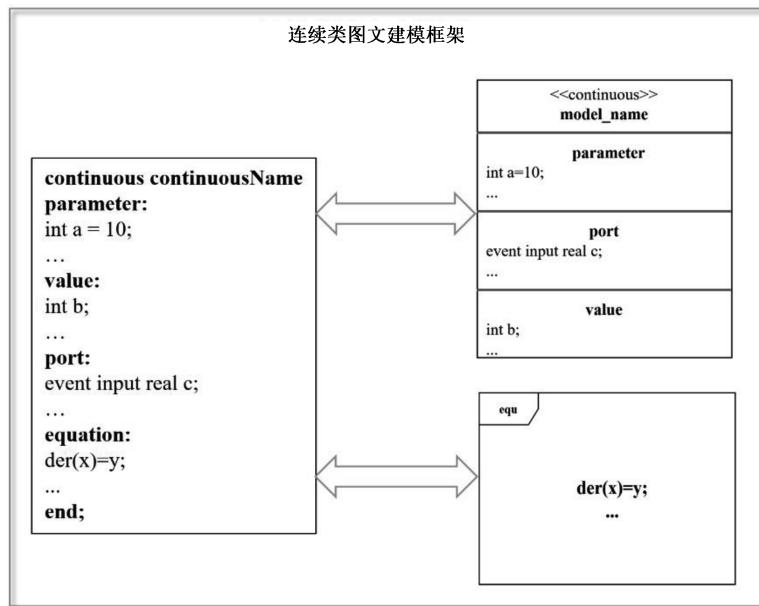


图 4 连续类图文描述架构

6.1.3 离散类

离散类(discrete)是用来描述复杂产品中基于状态描述的离散模型(离散行为)。一般地,discrete类常包含定义部分、状态机部分。定义部分用来初始化参数和变量的值,以及相关组件和输入输出端口;状态机部分用来定义离散模型的状态以及状态之间的转移逻辑。其描述的图文建模架构如图5所示。

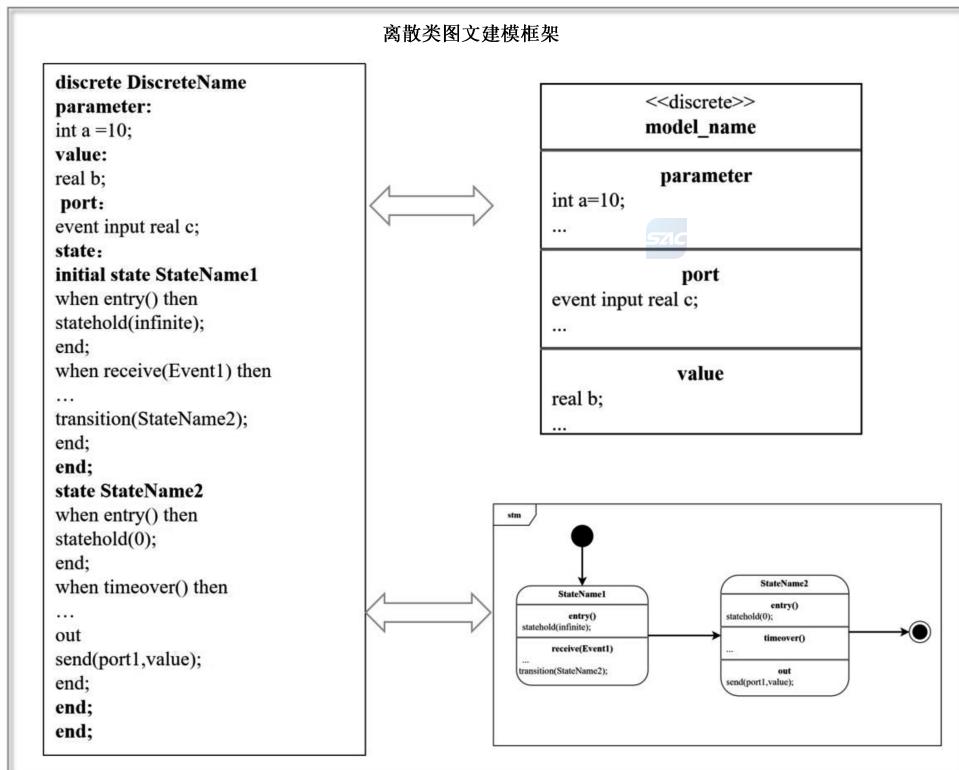


图 5 离散类描述架构

由于状态部分中关于状态的定义是 discrete 类行为的核心,在此单独进行介绍,其描述架构如图 6 所示。

```

state state_name
when entry() then
    //设置进入状态时的准备工作
    statehold(time);
    //在进入状态时状态持续时长
end;
when receive event_1 then
    //针对该状态下收到事件event_1的行为, 可定义状态
    //转移
when timeover() then
    //针对该状态下持续时间结束时(内部事件)的行为
    out
    //定义内部事件对应的输出行为
end;
end;

```

图 6 状态定义描述架构

6.1.4 桥接类

桥接类(couple)是用来描述复杂产品中多领域、多特征、多层次的系统模型。一般地,couple 类常包含头部分、定义部分、连接部分。头部分包括导入外部模型(结构关键字 import)以及继承外部模型(结构关键字 extends)两部分内容;定义部分用来描述模型中定义好的 continuous 类、discrete 类、agent 类并将其实例化为系统模型的组件及其输入输出端口声明;连接部分用来描述各组件之间的连接关系,使用各子模块的 event 端口进行连接。其描述的图文建模架构如图 7 所示。

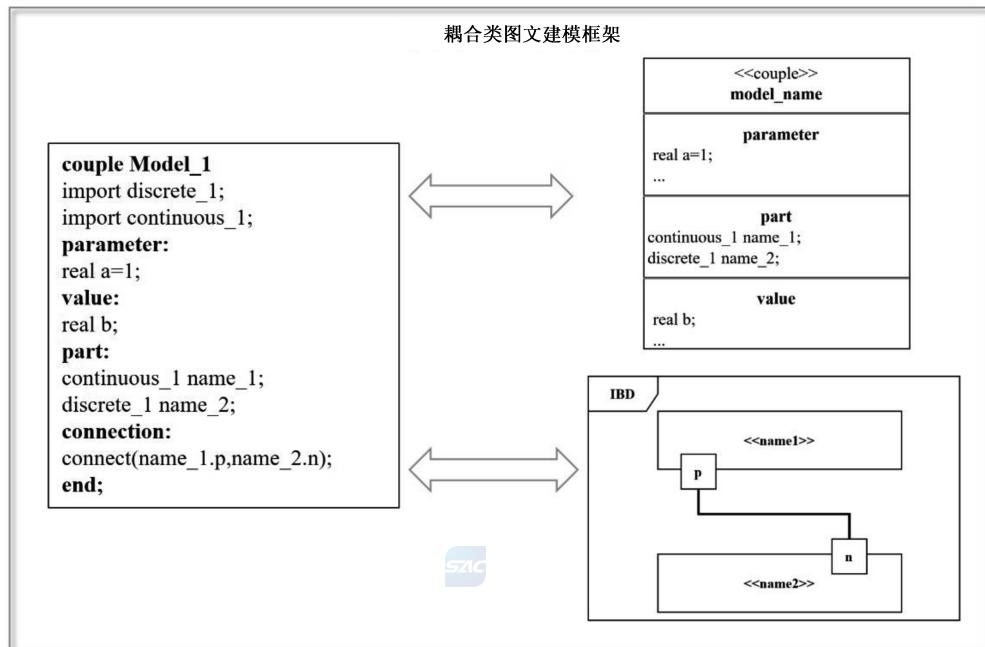


图 7 桥接类描述架构

6.1.5 智能体类

智能体类(agent)是用来描述复杂产品中具有交互和学习行为的智能体模型。一般地,agent 类常包含定义部分、活动部分。定义部分用来初始化参数和变量的值,以及函数和计划的声明也在该部分完成;活动部分用来控制计划的执行,以及设置智能体仿真的开始和终止条件。

在整个 agent 类中,存在一个特殊的结构,即计划(plan),用于表征由一组动作组成的智能体的行为序列。一个计划往往由多个函数组成,是智能体控制其行为的最小单位。计划可以看作是由多个函数组成的一个函数集,但是计划与函数又存在以下差异:

- 计划没有输入值与返回值,参数的传递需要利用全局参数实现;
 - 计划内部可以调用智能体的消息发送与接受动作,而函数却不能做到这一点;
 - 计划只能由整个智能体语言架构中的执行部分调用,而函数则可以被任意其他函数调用。
- 其描述的图文建模架构如图 8 所示。

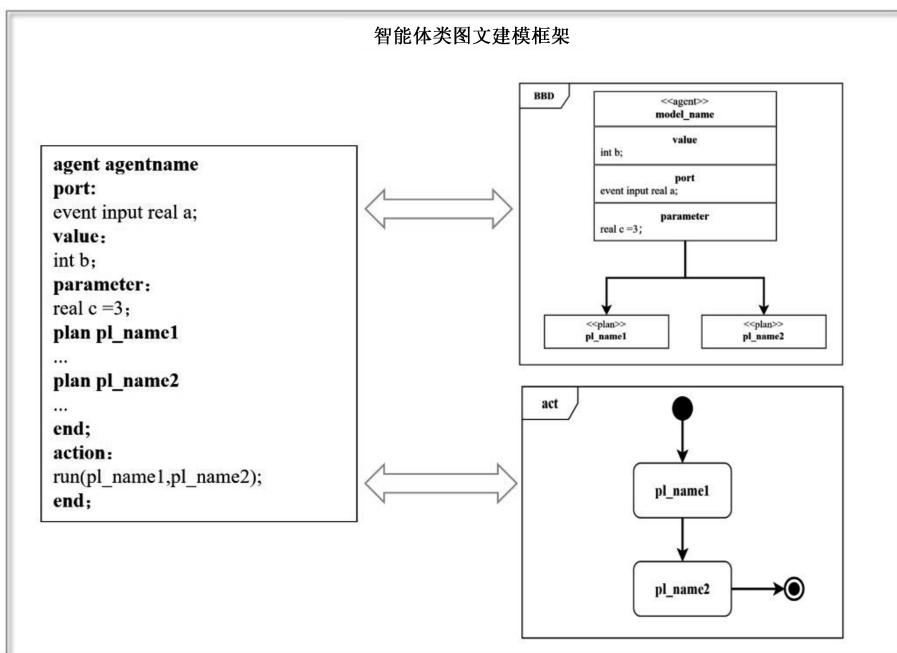


图 8 智能体类描述架构

6.1.6 记录类

记录类(record)是用来描述复杂产品中各模型中涉及的复杂数据结构。一般地,record 类是由 definition 部分组成。definition 部分用来定义各种数据类型。其描述架构如图 9 所示。

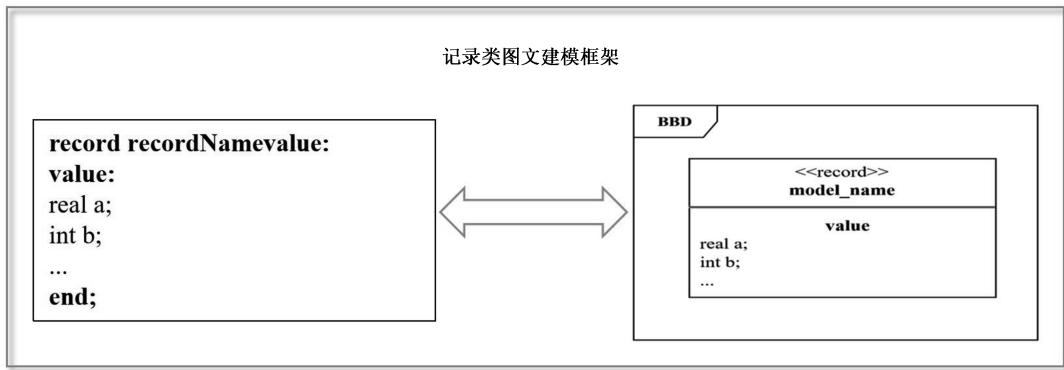


图 9 记录类描述架构

6.1.7 函数类

函数类(function)是用来描述复杂产品中各模型中涉及的功能行为模块。一般地,function类是由definition部分、action部分两个部分组成。definition部分用来定义输入输出的参数,action部分用来定义function类特定功能。其描述架构如图 10 所示。

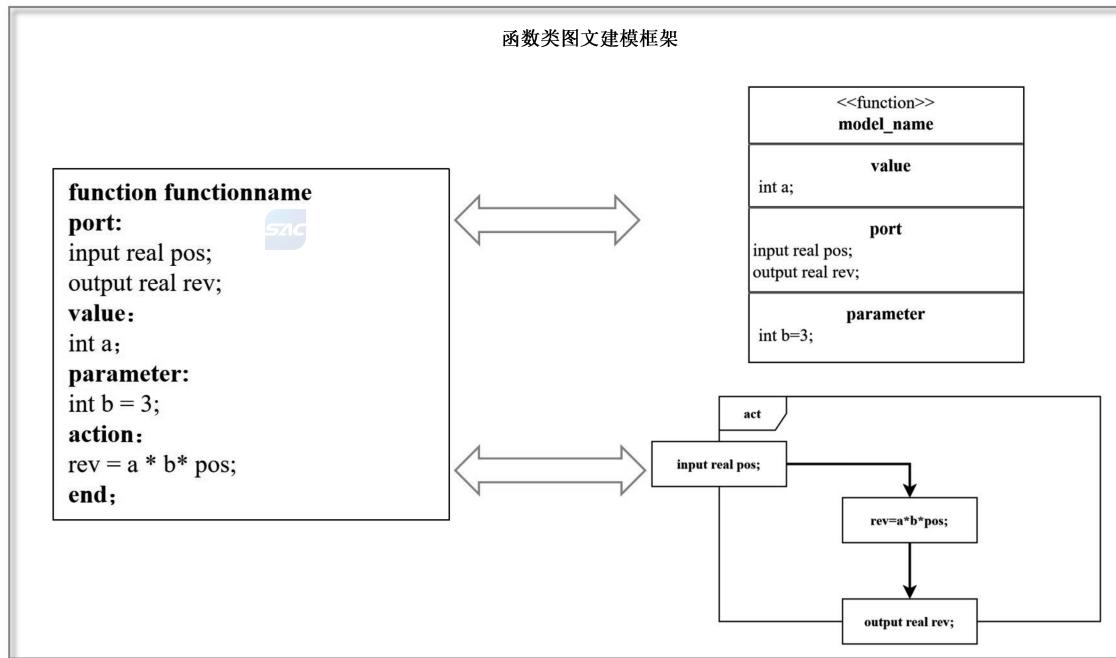


图 10 函数类描述架构

6.1.8 连接器类

连接器类(connector)是用来描述复杂产品中各模型中涉及非因果连接器端口。一般地,connector类是由definition部分组成。definition部分用来定义端口数据类型。其描述架构如图 11 所示。

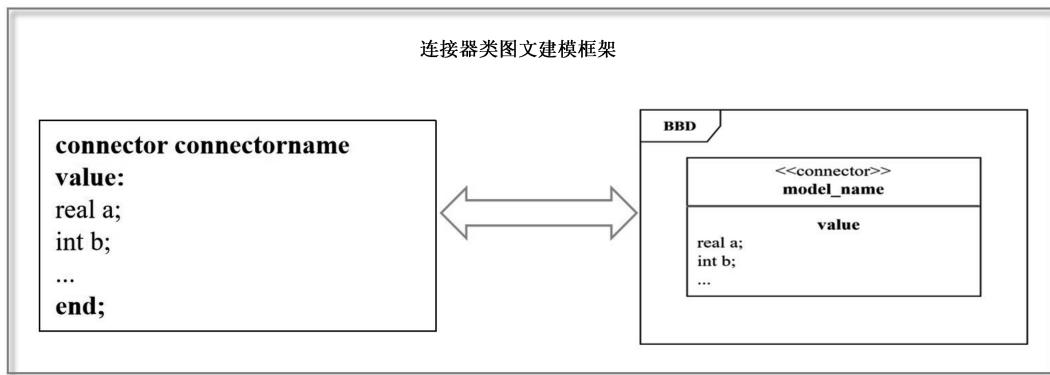


图 11 连接器类描述架构

6.2 图形建模

6.2.1 概述

X 语言的图形建模主要基于系统建模语言图形建模进行设计,参考 SysML 的需求图(REQ)、用例图(UC)、块定义图(BDD)、内部模块图(IDB)、参数图(PAR)、状态机图(STM)、活动图(ACT)七种图,并对其元模型进行了相对应的删减、修改与扩展。其中 BDD、IBD、PAR、STM、ACT 五种图与 X 语言中的 definition 部分、connection 部分、equation 部分、state 部分和 active 五个部分对应。图形建模包括定义图、连接图、方程图、状态机图、活动图、需求图以及用例图七种模型图。

6.2.2 定义图

6.2.2.1 结构特性

定义图的结构特性包含组成元素、导入元素、值元素、参数元素和端口元素 5 种类型。这些类型在图中的描述规则如表 1 所示。

表 1 结构特性命名规则

序号	结构特性类型	命名规则
1	组成元素	<pre> part partPropertyName </pre>
2	导入元素	<pre> import importPropertyName </pre>
3	值元素	<pre> value Type valuePropertyName </pre>
4	参数元素	<pre> parameter Type parameterPropertyName = constant </pre>

6.2.2.2 关联

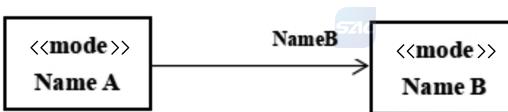
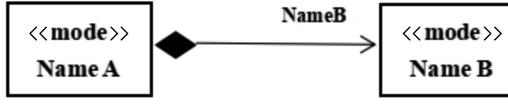
定义图的关联分为引用关联和组合关联。

引用关联表示双方存在一种连接,双方可以相互访问。模块间的引用关联通过实线标识,若实线无箭头,则代表双向访问,若有箭头,则表示单向的访问。用 import 表示,括号内包含所连接的两个模块的名称(包含角色名称和多重性)以及引用关联的名称。此处使用点标识法来将角色名称绑定到模块名称上,表示所属关系。

组合关联表达的是一种构成关系,组合段的模块实例由组成部分端的实例组合而成。

定义图的关联命名规则如表 2 所示。

表 2 关联的命名规则

序号	结构特性类型	命名规则
1	引用关联	
2	组合关联	

6.2.3 连接图

连接图中两个属性之间的连接器表示两个结构的正确组装且可操作的系统中,拥有某种方式可以相互访问,传递事件、能量或数据。连接器表示如图 12 所示。



图 12 连接器的表示形式

6.2.4 方程图

方程图描述 continuous 类模型中变量的行为约束,通过多组不同的等式方程组刻画 continuous 模型的行为。

如果一个 continuous 类的行为需由多个不同类型的方程去描述,方程图会提供不同的区域来描述不同的部分,具体如图 13 所示。

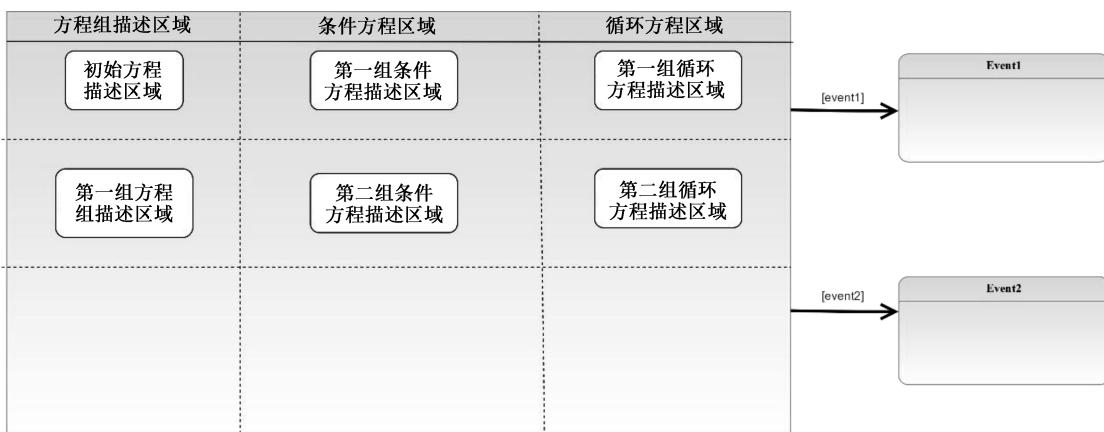


图 13 方程图框架

6.2.5 状态机图

6.2.5.1 状态定义

state 定义一个状态,后面接该模块的名称 State_name1,最后以 end 结束该模块的定义。其表现形式如图 14 所示。

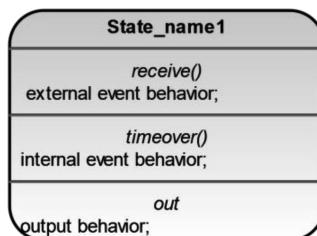


图 14 状态的定义形式

其中,recieve(event_name1) 表示在 State_name1 下收到 event_name1 事件,然后执行外部事件行为。timeover()表示状态持续时间结束,执行内部事件及内部事件行为,out 表示内部事件结束时相关输出行为。

6.2.5.2 状态转换

SAC 状态之间的转换是通过箭头表示的,外部事件会触发转换。另外,在某一状态保持时间由箭头上方 after time 表示。其表现形式如图 15 所示。



图 15 状态转换的表现形式

6.2.6 活动图

活动图的表现形式如表 3 所示。

表 3 活动图的表现形式

序号	名称	描述	表现形式
1	对象节点	一种能够存在于活动之中的节点,会对对象令牌(代表事件、能量或者数据的实例)通过活动的流建模	Type: objectNodeName
2	栓	一种特殊类型的对象节点,一般会附加到动作上,表示动作的输入或输出	
3	活动参数	一种特殊类型的对象节点,一般会附加到活动图外框上,从总体上表示活动的一种输入或者输出	
4	对象流	一种传输对象令牌的边。使用对象流,可以表示事件、能量或数据的实例通过活动	
5	控制流	一种传递控制令牌的边。控制令牌的到达可以启动等待它的动作	
6	初始节点	标记活动的起点,它标记了活动中的一个位置,控制令牌的流会从此处开始	
7	活动最终节点	标记控制令牌流结束的控制节点。当控制令牌到达活动最终节点的时候,整个活动都会结束,以此标记所有控制流的结束	
8	流最终节点	标记控制令牌流结束的控制节点。当控制令牌到达流最终节点的时候,那个令牌就会被销毁,以此标记单独一个控制流的结束	
9	决定节点	标记在活动中可替换序列的开始。决定节点应拥有单一的输入边,一般拥有两个或多个输出边。每个输出边会带有布尔表达式的标签	
10	合并节点	标记活动中可选序列的结尾。合并节点拥有两条或多条输入边,而只拥有一个输出边。当一个令牌通过任意一条输入边到达合并节点,令牌马上就会提供给输出边	

6.2.7 需求图

需求图的表现形式如表 4 所示。

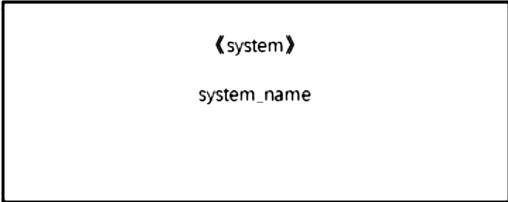
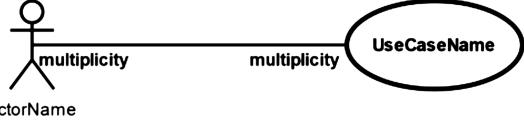
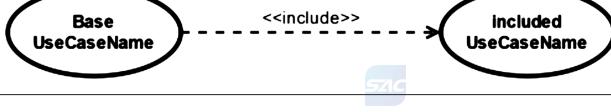
表 4 需求图的表现形式

序号	名称	描述	表现形式
1	利益相关者	对系统有具体的需求要求并与其利益相关的人员	<pre><<stakeholder>> stakehold_name</pre>
2	需求关系	系统需求间以及需求和设计制品之间的关系标识	<pre><<requirement>> Requirement Name id id="text string" Type type= General/Functional/Nonfunctional/ Level level=StakeholderReq/SystemReq/ComponentReq Text Text="text string" tracedTo 《elementKind》 Element Name A tracedFrom 《elementKind》 Element Name B derived 《requirement》 Requirement Name A derivedFrom 《requirement》 Requirement Name B refineBy 《elementKind》 Element Name C satisfiedBy 《couple/continuous...》 couple/continuous_name verifiedBy 《testCase》 Test Case Name</pre>
3	包含关系	表示某一需求包含其他需求的标识	<pre><<requirement>> Requirement NameA + <<requirement>> Requirement Name B <<requirement>> Requirement Name C</pre>
4	继承关系	表示客户端需求继承了提供方需求的标识	<pre><<requirement>> Requirement Name A --><<deriveReq>>--> <<requirement>> Requirement Name B</pre>
5	改善关系	表示客户端元素比提供方元素更加详细的标识	<pre><<elementKind>> Element Name --><<refine>>--> <<requirement>> Requirement Name</pre>
6	满足关系	表示客户端元素的实例会满足提供方的需求标识, 客户端元素一般是类	<pre><<block>> Block Name --><<satisfy>>--> <<requirement>> Requirement Name</pre>
7	验证关系	表示客户端元素验证了提供方元素的需求标识, 客户端元素一般是测试案例	<pre><<testCase>> Test Case Name --><<verify>>--> <<requirement>> Requirement Name</pre>

6.2.8 用例图

用例图的表现形式如表 5 所示。

表 5 用例图的表现形式

序号	名称	描述	表现形式
1	系统边界	表示拥有并执行用例的系统	
2	用例	特定的一系列动作,包括变种系列动作和错误系列动作,系统、子系统或者类可以通过与外部对象交互来执行,以提供值的服务。一般用动词短语描述	
3	执行者	表示执行系统用例的人或者系统	
4	关联(执行者与用例)	表示执行者与用例之间的交互	
5	内含用例	表示两种用例之间的一种包含关系,内含用例行为是源端用例所需要的组成部分	
6	扩展用例	表示两种用例之间的一种关联关系,目标端的用例被触发时,源端的扩展用例可能被选择性地执行	