计算机科学 1999Vol. 26№. 5

RIG 推荐标准 BIDM 与 UDM 简介

An Introduction to the RIG Proposed Standard BIDM and UDM

李克勤 邹 炜 文进炜

（北京大学计算机科学技术系 北京 100871）

Abstract The research on software reuse is a time-honored focus of academic and industrial communities. Software reuse is deemed as one of the practical approaches to alleviate "Software Crisis". Currently there are few commonly practical methods for assets sharing among so many government and industry software reuse libraries. The result is inefficiency and unnecessary redundancy. The Reuse Library Interoperability Group (RIG), founded in 1991, is engaged in developing interoperability solutions. Under such conditions, Technical Committee 2 (TC2) of the RIG developed the Uniform Data Model (UDM), which defined asset information set which reuse libraries should be able to exchange to achieve interoperability. The goal of this effort is to enable reuse libraries to interoperate seamlessly based on UDM.

Keywords RIG, UDM, Reuse Library, Interoperability, Data Model

现存的复用库使用各自不同的数据模型、分类模式和术语.这种多样性对于支持不同的领域、客户和技术研究是十分有用的,但同时也妨碍了库之间软件资产的共享。在某一个库中工作的复用者可能对其它库中的软件资产一无所知，即使复用者在其它库中发现了感兴趣的软件资产,他们仍然需要帐户等手段以进入其它复用库。不能共享其它库中的可复用软件资产,使得复用者可用的软件资产的集合缩小了,并可能导致重新开发软件资产的重复劳动。

可互操作性,将允许库使自己的公共可访问软件资产被其它库所了解,并使得复用者不必对其它库进行直接的访问.没有可互操作性,复用者浏览不熟悉的库时,或从不熟悉的库获得软件资产时,将需要了解库如何为软件资产建模,分类模式如何工作,以及术语如何定义等诸多问题，而在软件复用中理解问题被认为是一个关键的问题.没有可互操作性,复用者在理解其它库中的资产时将会花费更多的时间,因为他必须理解库中关于资产的表示.通过可互操作性,其它库的数据模型将映射为熟悉的库的模型,这样就减少了复用者的理解负担。

可互操作性,即复用库间软件资产的共享,可能是在短期内提高效率、增加复用库的价值和影响,并从长期避免不兼容协议的爆炸性增长的关键。基于这样的认识,由政府、企业界和学校的代表组成的RIG 复用库可互操作性组织,致力于开发资产库间可互操作性的解决方案。本文将对 RIG 开发的关于可互操作性的标准进行简要的介绍。

一、RIG 及其数据模型

BIDM 与 UDM 是由 RIG 开发的标准。RIG 成立于 1991 年,是一个志愿的基于一致意见的组织,由来自政府、学校和私人企业的成员组成。RIG 的工作是起草复用库可互操作性的术语、交互协议、软件资产交换格式等方面的标准。RIG 的建议标准可以提交给 IEEE、ANSI 或其它标准化组织.

RIG 的技术委员会研究在一个库的机制中表示的资产如何被具有不同机制的其它库使用，这些小组研究为达到基本的表示能力所需的数据类型,描述库数据模型的标准手段,以及传输这些数据的最好方法，这些信息支持互相协作的复用库间的软件资产的交换,使得某一个库的用户可以使用本库的浏览机制与另一个远程库交互,并浏览和提取其中的资产,就像这个远程库是本库的扩展一样。

RIG 当前研究的技术策略基于 ALOAF 的三层数据模型,其中包括元模型层、模型层、数据层。元模

型提供一组基本的组成成分和规则，用于数据模型的产生和修改。数据模型描述在库系统中维护的软件资产的数据的结构。数据层是由数据模型组织并与数据模型一致的真正的数据。RIG 正在开发元模型和数据模型层的标准。RIG 的标准数据模型就是"统一数据模型"(UDM)。

RIG 认为，现有库系统的数据模型各不相同，并且为了保持各个库中的软件资产与领域相关的特性，它们之间的差异将会继续存在，各个库的数据模型和描述模型的方法的差异使得库之间的数据交换较为困难——即降低了可互操作性。UDM 提供了一个可能的解决方案。它提供了一个标准的数据模型，各个库可以将它作为自己的数据模型的中间表示，来和其它库交换使用各自数据模型的数据。

作为定义 UDM 的一个步骤，RIG 的第二技术委员会(TC2)开发了"基本可互操作性数据模型"(BIDM)，它是 UDM 的一个子集。BIDM 定义了为了实现互操作，复用库交换软件资产时所需的信息的最小集。TC2 认为这个信息的最小集应使复用库的用户可以快速、准确地判断出其它复用库中的哪些软件资产可能适合自己的需要。

UDM 可能不是定义可互操作数据模型的过程的最终结果。库数据模型的结构和使用这些结构的数据模型的表示和交换可能需要 UDM 的一个超集。这个超集将可能提供比 BIDM 和 UDM 更加强大的软件资产交换能力。

二、术语定义

在 UDM 和 BIDM 中使用了一些重要的术语，它们的定义如下：

1. **软件资产(Asset)**：(1)保存在复用库中的、复用者可能感兴趣的项，例如设计文档、规约、源代码、文档、测试方案等，或对复用者有潜在价值的任何其它信息单元。(2)在 UDM 和 BIDM 中为复用库中的软件资产建模的一个类。

2. **属性(Attribute)**：提供类的性质的、预定义的特征。类的属性可以被它的子类继承。

3. **类(Class)**：一组相似的对象，它们具有相同的结构，但可能具有不同的属性值和(或)关系值。

4. **类层次(Class Hierarchy)**：类的一个排列次序，其中子类是其父类的特殊化，类从其父类继承属性和关系，并且可以定义自己的附加的属性和关系。

5. **库数据模型(Library Data Model)**：作为复用库中结构化数据的基础的组织原则和概念，以及表示该结构的手段。

6 **对象(Object)**：现实世界中实体的表示。对象是类的实例，对于类中定义的属性和关系具有具体值。

7 **关系(Relationship)**：两个类之间的联系。

三、UDM 类层次

UDM 和 BIDM 定义了一个基于 ADOAF 的方法学和概念的元模型，这个元模型使用了下列实体：类、类层次、类属性、类间的双向关系和关系属性。

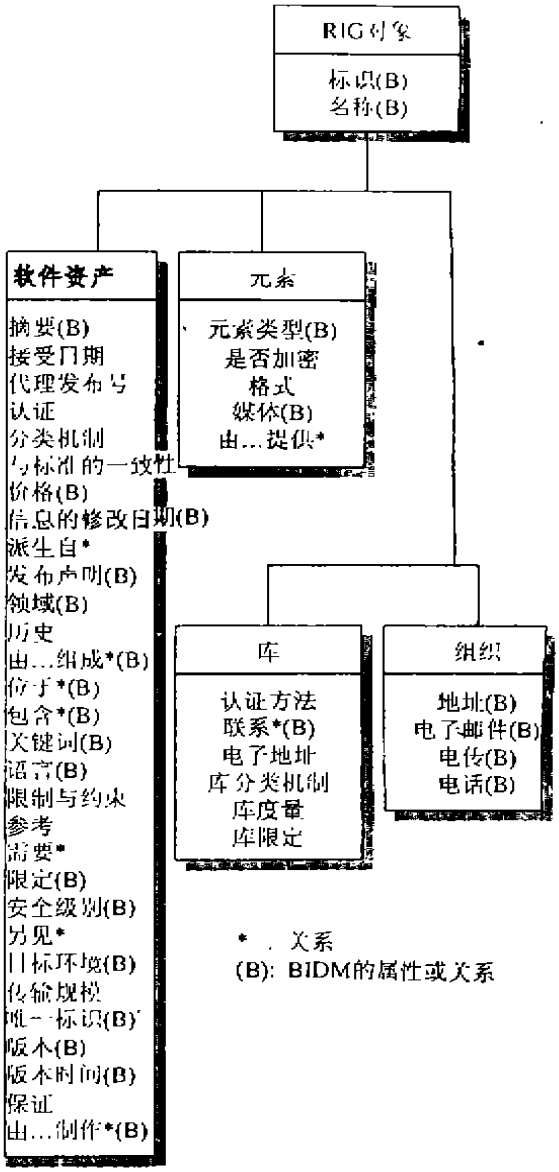

图 1 UDM 类层次

图1是UDM的一个总览，显示了UDM中的类、属性和关系，其中标有(B)的属性和关系是BIDM定义的。

四、UDM元模型

UDM元模型使用下列实体：类、类层次、类属性、类间的双向关系、关系属性。

类用来为行为者和事物建模。类使用表1中的模板来定义。类模板中的部分数据允许为空。

表1 UDM类模块

名称	
父类	
子类	
描述	
使用说明	
继承属性	
类的局部属性	
继承关系	
局部关系	

类具有属性。属性使用表2中的模板来定义。属性模板中的部分数据允许为空。

表2 UDM属性模板

属性名称	
描述	
类	
单值/多值	
数据类型	
上界	
值域	
长度	
是否可选	

类之间可能存在关系。所有关系都是双向的，在两个方向上都要给出基数。关系的属性显示在表3的模板中。其中"逆向名称"一项允许为空。

表3 UDM关系模板

名称	
描述	
源类	
目标类	
逆向名称	
最小目标基数	
最大目标基数	
最小源基数	
最大源基数	
约束	

五、UDM数据类型

在UDM中使用下列数据类型来描述属性：

Date：表示日期，分为年、月、日三部分，都用正整数表示；

Integer：有符号整数，值域从-2^{31}到$2^{31}-1$；

String：最大长度为1023的一串可打印字符；

Text：任意长度的一串可打印或不可打印的字符。

六、UDM类简介

在UDM的类层次中包含五个类，其中一个是父类，四个是子类。这四个子类代表UDM进行建模的实体，即：软件资产，元素(其中包括文件)，库，组织(其中包括个人)。

RIG对象(RIGObject)是类层次中的根类，它为UDM中的其它类提供了基础，在其中定义的局部属性被其它类所继承。它不能用来实例化自己的对象。

软件资产(Asset)为可复用实体建模，提供可复用实体的描述信息。一个可复用实体至少包含一个软件资产对象。

元素(Element)为软件资产的构成成分建模。这些成分包括文档、需求规约、测试用例、源代码、安装信息和说明文件。软件资产对象至少包含一个元素对象。在一些库中文档、需求规约、测试用例等可能本身是软件资产而不是软件资产的元素。

库(Library)为包含可复用实体的库建模。这里只提供在交换软件资产的过程中必需的信息。

组织(Organization)描述如人、公司、委员会等实体。在交换软件资产的过程中不需要的信息不包括在内。

七、UDM属性简介

UDM中的每个类都具有一些属性，以下对每个类分别进行简要介绍。

1. **RIGObject(RIG对象)**：RIGObject类的属性包括标识(Identifier)和名称(Name)。其中标识是对象的唯一标识。这个标识在软件资产交换的环境中是唯一的。名称是对象的名称或标题。一个对象可以有多个名称。这两个属性都被子类所继承。

2. **Asset(软件资产)**：Asset类的属性包括：

摘要(Abstract)：软件资产的一般定义和(或)

解释；

接受日期(AcceptanceDate)：指示软件资产被初次接受并加入库中的日期；

代理发布号(AgencyPublicationNumber)：由代理赋于软件资产的标识号或字符串；

认证(Certification)：包括对评价(Evaluation)或验证(Qualify)软件资产的方法的讨论、评价或验证的结果、实施认证的日期、实施认证的人员。可能是对库的认证方法(CertificationMethods)属性的引用；

分类机制(ClassificationMachanism)：分类软件资产的方法；

与标准的一致性(ComplianceToStandards)：软件资产所符合的标准；

价格(Cost)：订户为获得复用软件资产的权利所必须付出的费用的种类和数量；

信息的修改日期(DateOfInformation)：关于软件资产的信息的最后修改日期；

历史(History)：软件资产演化的完整描述。其中可能包括名称和演化过程中主要步骤的日期。即软件资产的"家谱"；

关键词(Keyword)：描述软件资产某一方面性质的词或短语，例如软件资产具有的功能，操作的对象，这种描述有利于产生分类信息；

限制和约束(LimitationsAndConstraints)：限制软件资产的可复用性的因素。应考虑算法、编译器、可移植性、副作用、环境等；

参考(Reference)：对包含有关软件资产的有用信息的信息源的声明；

限定(Restriction)：关于软件资产使用的法律信息，应考虑版权、数据权限、不承诺信息、出口限制、许可等；

目标环境(TargetEnvironment)：软件资产所针对开发的计算机系统、操作系统、编译器的名称；

传输规模(TransferSize)：以电子形式存在的软件资产包含的元素在进行电子传输时需要的字节数；

唯一标识(UniqueID)：指示构成软件资产实体的数据集的名称或句柄。它不仅在软件资产传输的环境中唯一，并且在所有环境和库中唯一；

版本(Version)：软件资产版本的指示；

版本日期(VersionDate)：软件资产的当前版本完成的时间，更进一步地，是库认为信息可供公共使用的日期。发布日期；

保证(Warranttes)：描述软件资产的组织保证的说明，应证明软件资产的完整和可靠性。另外，应说明所有者维护和替换有缺陷的软件资产的责任。

3 **Element(元素)**：Element 类的属性包括：

是否加密(Encrypted)：描述元素是否使用加密算法生成；

格式(Format)：元素存储的格式，特别是是否需要特殊的工具解释和处理其内容。

4 **Library(库)**：Library 类的属性包括：

认证方法(CertificationMethods)：对库中的软件资产进行评价和验证的方法的讨论，其中包括评价和验证的结果的范围，进行认证的时间、方式和人员；

电子地址(ElectronicAddress)：库的电子地址；

库分类机制(LibraryClassificationMechanism)：描述库存储和分类软件资产的方法；

库度量(LibraryMetrics)：反映库属性的度量，可包括软件资产的数量，订户的数量，提取软件资产的数量，可互操作性度量；

库限定(LibraryRestrictions)：关于库的使用的法律信息，应考虑数据权限、不承诺信息、出口限制、许可、安全性。

5 **Organization(组织)**：Organization 类的属性包括：

地址(Address)：人或组织的通讯地址；

电子函件(Email)：组织的电子函件地址；

电传(Fax)：组织的传真号码；

电话(Telephone)：组织的电话号码。

八、UDM 关系简介

UDM 中的类之间可能存在双向的关系，在 UDM 中，为简便起见，这些关系是定义在源类中的，以下对每个类中定义的关系分别进行简要介绍。

1. **Asset**

Asset 类中定义的关系包括：

派生自(DerivedFrom)：是与 Asset 类之间的关系，标识软件资产所派生自的软件资产，典型情况下是软件资产的较老版本；

由…组成(IsComposedOf)：是与 Asset 类之间的关系，标识软件资产的子资产或组成部分。软件资产间的组成关系是独立于库的；

位于(IsLocatedIn)：是与 Library 类的关系，标识软件资产所在的库。库既包含软件资产的描述，也包含其本身；

(下转第 70 页)

21 ISO 9000-3, Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software, ISO/IEC, 1991

22 Jacobson I. et al Object-Oriented Software Engineering A Use Case Driven Approach. Addison-Wesley, Reading, MA, 1992. 524

23 Jones C Applied Software Measurement. Assuring Productivity and Quality McGraw-Hill, NY, 1991

24 Software Reuse: A Holistic Approach-Measuring the Effect of Reuse Chapter. Even-Andre Karlsson, ed. Chichester; New York. Wiley, c1995, published by Wiley & Sons, Ltd Baffins Lane, Chichester Sussex PO 191 UD, England

25 Kearney J K, et al. Software Complexity Measurement, Comm ACM, 1986, 29: 1044～1050

26 Lewis J A, et al. On the Relationship between the Object-Oriented Paradigm and Software Reuse: An Empirical Investigation. J. Object-Oriented Programming, 1992, 5(4) 35～41

27 Lorenz M Object-Oriented Software Development: A Practical Guide, Prentice Hall, NJ, 1993

28 McCabe T J. A Complexity Measure, IEEE Trans Software Eng., 1976, 2(4): 308～320

29 Mili Hafedh, et al. Reusing Software: Issues and Research Directions, IEEE Trans. On SE, 1995, 21(6): 528～562

30 Piper Joanne C, Barner Wanda L The RAPID Center Reusable Components (RSCs) Certification Process. U. S. Army Information Systems Software Development Center-Washing, Ft Belvoir, VA

31 Poulin Jeffey S. Measuring Software Reusability Third International Conference on Software Reuse, 1994

32 Prather R E An Axiomatic Theory of Software Complexity Measures, Comput. J., 1984, 27 340～346

33 Rajaraman C, Lyu M R. Some Coupling Measures for C++ Programs. In Procs. TOOLS USA'92, Prentice Hall, Englewood Cliffs, NJ, 1992. 225～234

34 Tegarden D P, Sheetz S D. Object-Oriented System Complexity: An Integrated Model of Structure and Perceptions. In OOPSLA'92 Workshop on Metrics for Object-Oriented Software Development Washington DC, 1992

35 Vessey I, Weber R. Research on Structured Programming. An Empiricist's Evaluation, IEEE Trans. Software Eng., 1984, 9-10: 394～407

36 Weyuker E Evaluating Software Complexity Measures, IEEE Trans. On Software Eng., 1988, 14: 1357～1365

37 Zues H. Software Complexity: Measures and Methods, Walter de Gruyter, Berlin, 1990, 605

(上接第 20 页)

包含(IsMadeOf):是与 Element 类之间的关系,标识构成软件资产的元素;

需要(Requires):是与 Asset 类之间的关系,标识相互依赖的软件资产;

另见(SeeAlso):是与 Asset 类之间的关系。标识对本软件资产的理解、使用等可能有帮助的其它软件资产;

由…制作(WasCreateBy):是与 Organization 类之间的关系。标识软件资产的制作者。

2 Element

Element 类中定义的关系包括:

由…提供(ProvidedBy):是与 Organization 类之间的关系。标识元素的提供者。

3. Library

Library 类中定义的关系包括:

联系(ContactIs):是与 Organization 类之间的关系。标识对库负责的组织。组织可能是个人。

结论:UDM 为在库间传送软件资产提供了一个通用的表示,消除了每个库为所有其它库产生映射的需要。UDM 也标识出了描述可复用软件资产的一些重要的特性,如价格、认证、语言、保证、目标环境、位置、作者、内容等。UDM 还提供了一个包括 UDM 属性、关系和属性值列表的术语集,库可以将这个术语集与各自的术语建立联系。

参 考 文 献

1 RIG Basic Interoperability Data Model (BIDM): [RPS-0001]: Reuse Library Interoperability Group, April 1993

2 RIG Uniform Data Model for Reuse Libraries (UDM): [RPS-0002]. Reuse Library Interoperability Group, January 1994

3 RIG Glossary: [SDR-0001]. Reuse Library Interoperability Group, April 1993