一种图形化对象式需求定义语言的设计*

张家重 吕 建 王志坚 徐家福

(南京大学计算机软件研究所 南京 210093)

(南京大学计算机软件新技术国家重点实验室 南京 210093)

摘要 本文简要讨论了软件需求定义及其语言的有关基本概念,重点介绍了图形化对象式需求定义语言 NDORL 的设计思想以及主要的语言结构与成分. 该语言是一种以面向对象方法支持软件需求定义的半形式化语言,具有形象易读、表达力强和便于实现到形式功能规约的转换等特点,并提供了方便的机器支撑.

关键词 面向对象,需求模型,需求定义,需求定义语言.

80 年代中期以来,特别是在近几年,国际上计算机科学界掀起了对需求工程的研究热潮.[1~3]作为其主要内容的需求定义与分析是软件开发过程的第 1 阶段,它的成功与否对目标软件的开发和维护起着举足轻重的作用. 但在计算机发展的早期,由于所求解问题的规模较小,需求定义与分析也因此而被忽视. 自 1968 年提出软件工程以来,才逐渐认识到需求定义与分析的重要性. 随着软件系统复杂性的提高和规模的增大,需求定义与分析愈加困难和耗时,使得它在软件开发中所占的工作量更加突出. 为此,人们试图通过对需求工程及其自动化技术的研究,以克服现有困难、提高生产效率、改进软件质量.

我们的研究目标是在已有软件生产自动化技术及系统,特别是在功能、设计和实现级软件自动化的研究基础之上,探索需求级软件自动化技术. 作为需求级软件自动化技术及系统研究的重要部分,我们近期设计了一种图形化对象式需求定义语言 NDORL,至今通过若干实例的试用已对初始版本进行了较大程度的改进. 本文首先讨论了需求定义及其语言的有关基本概念,然后重点介绍了 NDORL 的设计思想及其主要的语言成分.

1 有关的基本概念

1.1 需求定义及其作用

软件需求定义是软件需求的完整陈述. 它是软件开发人员与用户密切合作,了解用户的需要、目的和期望,并进一步表述而成的定义性陈述. 有时它又称为软件需求规约. 概略地

---

* 本文研究得到国家 863 高科技项目基金资助以及跨世纪优秀人才基金的部分资助.作者张家重,1965 年生,副教授,主要研究领域为软件自动化等. 吕建,1960 年生,博士,教授,主要研究领域为软件自动化等. 王志坚,1959 年生,博士,教授,主要研究领域为软件自动化等. 徐家福,1925 年生,教授,主要研究领域为软件自动化,新型程序设计等.

本文通讯联系人:张家重,南京 210093,南京大学计算机软件研究所

本文 1995 11-20 收到修改稿

说,其基本内容有2个部分:

(1)功能性需求:表达了软件系统"做什么",描述系统的输入、输出及其关系的信息,完整地刻画系统的行为,因此它是整个软件需求的核心.

(2)非功能性需求:定义系统的属性,包括系统的有效性、可靠性、安全性,易维护性和可见性等标准或限制.

一般说来,它的作用主要包括3个方面:

(1)它可作为系统开发者和用户之间的合约,使得双方对待解问题有一个共同理解,实现用户、分析员和设计人员之间的通讯.

(2)它可作为系统的开发依据,开发人员以此为基础进行软件设计和实现.

(3)它可作为系统确认和验证的基础,一方面,可以确认需求定义是否满足用户的要求,另一方面,可验证软件的设计与实现是否正确.

1.2 需求定义语言及其分类

需求定义语言是指用于书写用户对所需软件系统各种需求的语言.按照形式化程度,需求定义语言可分为3类:

(1)非形式需求定义语言:指未作任何限制的自然语言.其主要特征是非形式性,即具有歧义性、不完备性和模糊性.一般说来,非形式需求定义语言具有易理解、易使用的特点,易于为一般用户所接受.但是由于自然语言的非形式性而使得需求定义中常出现错误,并且难以用计算机系统提供自动化的支持.

(2)半形式化需求定义语言:指在宏观上对语言的语法和语义有较精确的描述,而在某些局部方面则允许使用非形式的自然语言.例如在结构化方法(SA)中,用图形化的数据流图来刻画系统的总体结构,即系统由哪几部分组成、各部分之间的关系如何等,而系统中各个成分的含义则通常用自然语言或专门设计的小说明语言来描述.其主要特点是既便于用户表达和理解相应的需求定义,又可在某种程度上用机器对相应的需求定义进行管理和部分正确性检查.

(3)形式化需求定义语言:指其语法和语义均精确定义的语言.这类语言通常以一定的数学理论为基础.目前,常用的数学理论有2类,一类是逻辑,代表性的工作有Z语言及其面向对象扩充版本(OOZE,Z^{++}等)和VDM-SL;另一类是代数,代表性的工作有OBJ语言和CLEAR语言.一般说来,形式化需求定义具有良好的数学性质,易于分析需求定义的各种性质,如一致性、完备性等,并可对其提供自动化支持,有利于在某种意义下保证后续开发步骤的正确性.它的缺点是难于理解.

2 NDORL语言的设计思想

2.1 基本设计准则

一般地,需求定义语言的设计应根据其基本内容考虑需求模型、语言结构和语用分析3个方面.对应于这三者,NDORL的基本设计准则为:

(1)需求模型的选择:需求模型指相应软件开发风范与方法在语言中的具体体现.它反映了软件需求的获取方式,决定了参加需求分析的各类人员的思维方式.我们通过剖析现有代表性的面向对象分析方法,如文献[4~7],提出了一个层次化对象式需求模型NDHORM

作为需求模型. 该模型及其构模方法的具体设计有另文介绍.

(2)语言结构的选择：语言结构提供用于刻画系统需求的具体手段，主要包括需求模型的描述、数据描述、控制描述、抽象机制以及项目相关信息描述等. 通过权衡各类形式的语言的利弊，并考虑到各层次用户和后续软件开发工作等的不同需要，决定采用图形、文本和形式化相结合的描述方式.

(3)语用分析的确定：主要讨论需求定义语言的适用领域、易扩展性等性质以及相应的方法与工具支持. 确定 NDORL 主要面向 MIS 系统的需求定义，并设计与实现其支撑系统.

据此，通过对需求定义语言有关概念的剖析，并考虑到从非形式的软件需求定义到形式的功能规约(自动)转换的需要，结合各种语言研究与设计[8~10]的经验，利用机器支撑的便利，将 NDORL 设计为一种以面向对象方法支持软件需求定义的图形化语言，即属半形式化需求定义语言.

2.2　对象及其关系的描述

软件领域中的对象概念应区分为 3 类，即需求时刻的对象、设计时刻的对象和实现时刻的对象.[8]由于不同阶段的对象所处的领域及侧重点不同，它们的含义就有所区别. 需求定义中的对象，即需求时刻的对象，亦称问题对象，是指需求领域中一切被明确定义边界的现实世界实体；“需求领域”指排除其它时刻的对象；“明确定义边界”指对象需与待解问题有关，排除那些被误认为是对象的情况.

依据 NDHORM 模型及方法，需求构模的第 1 步是构造与现实问题结构直接对应的对象关系图. 其目标是，识别并描述出问题对象是什么、其组成为何、对象间存在的关系怎样. 对象的识别与描述应遵循如下几点：(a)现实世界实体或概念；(b)与问题领域有关；(c)具有明确边界定义；(d)被封装的属性和操作(或称行为)应易于理解. 在 NDORL 中，对于属性和操作，根据需要可以描述成不同的抽象层次，既可以是非形式的，也可以是形式的. 由于需求定义只关心操作的功能行为“做什么”，不考虑其“如何做”，故形式的方法采用了前、后置断言的方式来刻画对象在其生存期内的动态行为，即状态空间的变化情况.

NDORL 中描述的对象间的关系刻画了对象之间的交互通信及消息传递情况，实际上是一种动态的行为关系，称之为事件. 这种关系主要靠分析问题领域中现实世界实体间(是否)存在什么样的关系来识别. 其描述采用与属性和操作一致的描述形式.

一般地，人们认识与分析问题的过程是由抽象到具体而逐步精化的，那么，人们对于问题对象及其关系的识别，即对象关系模型的构造，也应按照同样的过程. 为此，NDORL 提供了对象精化机制作为刻画这一过程的手段，使得所构造的对象关系模型是层次式的，即各层对象关系图之间有一定的层次精化关系，且把被精化了的对象，即具有子层对象关系图的对象称为聚合对象；其它对象称为原子对象. 事实上，对象精化是一个事件(或行为)制导的对象识别与构模过程，它一方面符合人们自然的思维方式，另一方面有助于组织需求定义的结构和控制其规模与复杂性.

2.3　类及其关系的描述

对象关系图虽然与现实问题结构具有直接的对应关系，但对于需求定义的总目标而言，它仅反映了问题的总体结构，并不能或不便于完整而精确地刻画问题需求定义的全部，为此引入并借助类关系图来弥补其不足.

NDORL 中把每个对象都归为某个类,将类看作是一组具有共同属性和行为的对象的抽象[8],且作为刻画对象的模板,并称聚合对象所属的类为聚合类;称原子对象所属的类为原子类.

NDORL 把类间关系归为聚合关系和子类关系 2 种. 聚合(Aggregation) 关系也称整体部分(Whole-Part)关系,指一些类是另一个类的组成部分,它对应于对象关系图中的对象精化,即层次精化关系. 子类关系也称类的层次关系(Is-A),指一个或一些类是另一个类的子类,后者称为父类. 父类具有所有子类中共同的属性与行为;子类至少具有父类中的全部属性与行为,并可能增加新的成分. 事实上,子类关系是组织需求定义,实现软件功能规约复用的一种有效手段,它与面向对象程序设计中的"继承"机制相对应,之所以不称其为"继承",是为了避免与"继承的目的是实现代码共享"的这种解释相混淆,而作为需求级概念的子类关系的主要目的是实现软件功能规约的共享.

与其它一些面向对象方法相同,NDORL 用类及其关系来组织与刻画对象的主要好处在于:(a)类便于刻画一组相似对象的共同性质;(b)类间关系便于实现需求定义的结构化以及未来软件设计过程中功能规约的复用;(c)完整描述了的类将作为软件功能规约的基本构造单位,易于实现需求定义到功能规约的自然过渡.

3 NDORL 的语言成分

3.1 需求定义的结构及其描述

采用 NDORL 语言所描述的软件需求定义的结构形式为:

```
REQUIREMENTS〈软件需求定义名称〉
    系统名称〈系统名称〉
    需求简介〈需求简介〉
    非功能性需求〈非功能性需求〉
    功能性需求
        对象关系图〈对象关系图〉
        类关系图〈类关系图〉
        类字典〈类字典〉
    其它〈其它〉
END REQUIREMENTS
```

其中〈需求简介〉包括应用范围、用户特性、功能简介、常识、一般约束、假设和依赖关系等的描述;〈非功能性需求〉包括软件接口、硬件接口、通信接口、设计约束、安全性、易维护性、易移植性和其它约束等的描述.

上述结构中除了〈功能性需求〉部分外,其它各部分均用非形式化的自然语言来描述. 〈功能性需求〉所包括的各部分主要采用图形化与形式化相结合的方式描述. 本文以下将重点介绍后者.

3.2 数据类型、变量和表达式

数据类型、变量和表达式的定义是半形式化或形式化表示的基础. 数据类型用于刻画需求定义中所涉及的数据及其上操作. 和其它软件规约语言一样,NDORL 定义了一组常见的数据类型,包括3种简单类型:整型(INT)、实型(REAL) 和布尔型(BOOL);3种复合类型:序列型(SEQ)、集合型(SET)和枚举型(ENUM)以及1种类类型. 它们统称为基本数据类型. 此外,还提供了具有2种形式的指定集合类型.

类类型即将类视作类型,可以把变量定义为具有某个类所表达的类型.类类型的名是类名,值集是其描述的对象集,操作集是类中的行为集合.以类为基类型的集合型和序列型,其值集分别是由以对象为元素的集合或序列构成的集合,其作用相当于"元类",这是定义复杂对象的重要手段.

指定集合类型借鉴并扩充了 Z 语言中的自由类型(Free Type)而定义的.它对应于目标软件系统中的数据库及其记录.一种形式是,每个指定集合类型说明引进1个或多个类型名,表示1个或多个指定集合类型.如[COMPANY],[PERSON]分别定义了2个指定集合类型 COMPANY 和 PERSON,它们也可以采用复合定义的方式,等价于[COMPANY, PERSON].另一种形式是,每个指定集合类型说明引进1个类型名,表示1个指定集合类型,并借助域类型表来说明其域.其中表有"*"的域表明它的值又是集合型,此处的域类型名即代表域名,又代表指定集合型.如[STUDENT # NUM,NAME,COURSE*]定义了指定集合类型 STUDENT,其域分别为 NUM,NAME,COURSE 并分别以 STUDENT.NUM, STUDENT.NAME, STUDENT.COURSE 的方式来引用,其中 COURSE 的值是集合型.

限于篇幅,不再细述各种数据类型所定义的操作.对于变量与表达式的定义,NDORL 与 FGSPEC[9]基本相同.

3.3　半形式化成分

NDORL 语言采用图形化方式刻画需求模型中的对象关系模型和类关系模型的总体结构及其成分间的关系,用形式化或自然语言的方式描述各成分的具体内容.本节及下一节分别介绍其图形化成分和形式化成分.

(1)对象关系图

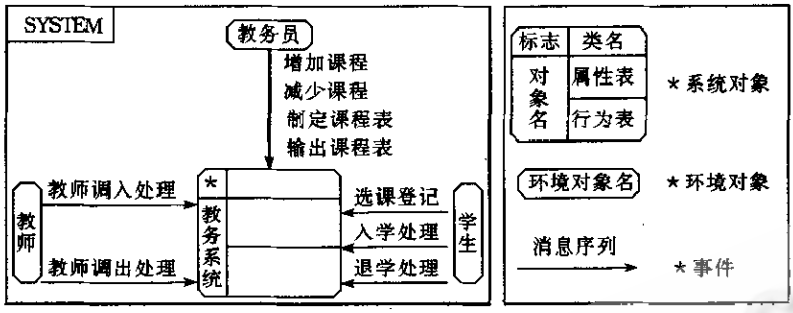
它分为若干不同层次,各层间的联系与区分是借助于一个称为"被精化对象"的对象名字来实现的.每层图由对象、事件及相应的图符等组成.当"被精化对象"为关键字 SYSTEM 时,表明是最顶层,并称之为目标软件系统的环境图,其中的环境对象是指目标软件系统的外部环境中的有关对象,如系统用户等,且约定它们仅出现在顶层中,顶层图如图1(a)所示.

非顶层对象关系图中仅含系统对象,即问题领域中欲构模的问题对象,它是通过精化某层某对象而得到(即识别)出来的.非顶层图如图1(b)所示,它是精化图1(a)中对象"教务系统"而得到的,图中仅与一个对象连接的事件符表示其另一端的缺省对象为其对应的被精化对象.

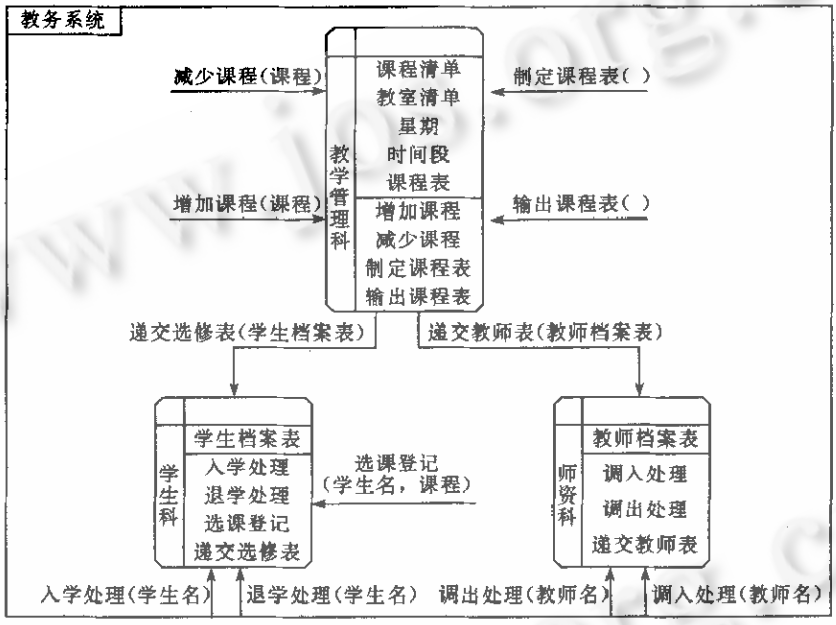
对象关系图中,对象可以缺省属性和行为的说明,因为它们可以在类或类字典中进行说明.如果对象"属性表"中具有形如〈属性名〉="值"的说明,则它指明该对象的创建约束,即表明此属性值在对象生存期内保持不变.若"类名"省略,表明该对象属于同名的类.具有"*"标志的对象为聚合对象,说明存在其子层对象关系图.

(2)类关系图

整个系统只用一个类关系图刻画,它由类、聚合关系和子类关系组成.图2是对应于图1的类关系图示意及其图例说明.其中聚合关系对应于前述的对象精化层次而产生,聚合类可以没有属性和行为说明,而原子类的属性和行为除了在图中列出外,还应根据实际需要决定是否需在类字典中以后述的形式化方式详细刻画.



(a) 顶层对象关系图及图例说明



(b) 非顶层对象关系图

图1

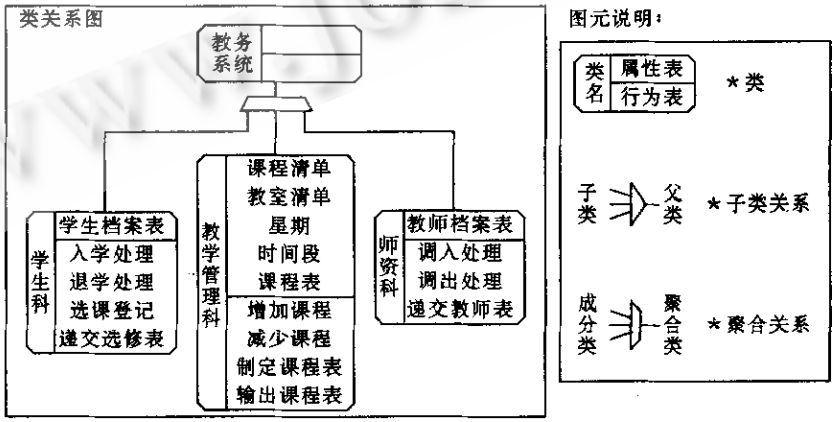


图2 类关系图示意及其图例说明

3.4 形式化成分——类字典

NDORL 的形式化成分便于作为软件设计的依据和转换的基础，主要体现在类字典的描述中. 类字典详细刻画类的有关信息，包括每个类的属性和行为的含义，它主要采用形式化方法来描述，其中所允许的非形式描述仅作为概念解释，以便于非专业人员理解.

类字典所描述的每个类皆允许有其常量说明和类型定义. 如果所说明的类不是 system 类，那么，常量说明和类型定义只在当前类中有效，否则，将被认为是全局的，即对所有类均有效. 类字典条目的形式概略地描述如下：

```
〈类条目〉::=CLASS〈类名〉
            [〈常量说明〉]
            [〈类型定义〉]
            [〈非形式化描述〉]
            [〈属性描述〉〈行为说明〉]
            ENDCLASS
〈属性描述〉::=ATTRIBUTE〈属性名〉:〈类型说明〉
              [CONSTRAINT〈约束条件〉]
〈行为说明〉::=BEHAVIOR〈行为名〉
                INPUT[〈输入参数表〉]
                OUTPUT[〈输出参数表〉]
                [DESCRIPTION〈语义解释〉]
                [〈断言式〉|CALL〈子行为引用表〉]
〈断言式〉::=PRE〈前断言〉,
            POST〈后断言〉
```

其中〈非形式化描述〉和〈语义解释〉采用自然语言来描述，以解释该类或行为在问题领域中的含义，其它部分均采用形式化方法来描述. 〈断言式〉是与 Z 语言具有基本相同约定的一阶谓词公式，用以刻画原子行为. 〈前断言〉中自由变量只能是相应〈输入参数表〉中变量的子集，〈后断言〉中自由变量只能是相应〈输入参数表〉中变量、〈输出参数表〉中变量、属性或标有撇号(′)的属性. 〈子行为引用表〉用以刻画复杂行为与其被精化而得到的若干子行为之间的关系. 〈断言式〉和〈子行为引用表〉二者只具其一.

以下是前例中类“师资科”的部分内容的形式化描述，其中大写词为保留字或新定义的数据类型.

```
CLASS TeacherOffice //师资科//
  [COURSE]
  [TEACHER#NUM,NAME,SEX,AGE] //定义指定集合类型//
  ATTRIBUTE teachers: SET (TEACHER) //教师档案表//
    CONSTRAINT
      ∀ t:TEACHER,t∈teachers⇒
                (t.NUM〈〉“ ”)∧(t.NAME〈〉“ ”)
  BEHAVIOR addteacher //调入处理//
    INPUT name:NAME, num:NUM, sex:SEX, age:AGE
    OUTPUT
    PRE ∀ t:TEACHER, t∈teachers⇒(num〈〉t.NUM)
    POST teachers′=teachers∪{(num,name,sex,age,)}
```

```
BEHAVIOR deleteteacher //调出处理//
  INPUT num: NUM,name: NAME
  OUTPUT
  PRE ∃ ! t: TEACHER, t∈teachers⇒(num=t.NUM) ∧ (name=t.NAME)
  POST teachers'=teachers\{(num,name,,)}
ENDCLASS
```

以上各部分介绍了NDORL的有关成分,它们分别以不同的抽象层次刻画了同一个问题的不同侧面,而如何保证不同侧面的完备性和它们之间的一致性也是NDORL需要解决的问题.为此,NDORL提出了若干相关的正确性准则,并提供了较好的机器支撑,旨在一定程度上保证需求定义的正确性.因篇幅关系,不再细述.

4 结束语

作为"需求级软件生产自动化技术及系统"研究的首要部分,通过对图书馆图书流通管理系统等若干实例进行需求定义的试用,已逐步改进了NDORL语言版本,并在SPARC 490工作站上实现了其支撑系统.概括起来,它的主要特点有:(a)采用面向对象分析方法,提供了有效的需求定义组织手段;(b)由抽象到具体、逐步精化的层次构模成分,有利于控制需求定义的结构和规模;(c)图形、文本和形式化相结合的方式适于各类人员之间的交流;(d)具有较强的表达能力,并提供了方便的机器支撑.此外,采用NDORL进行分析而得到的需求定义具有形象直观、易理解、易维护、便于复用等特点,而且其中大量烦琐、易出错的部分由计算机支撑系统辅助完成,从而在一定程度上保证了需求定义的正确性,减轻了分析人员的负担,有助于提高软件的质量和生产率.

NDORL是我们在需求级软件自动化研究方面的初步尝试,尚有欠妥之处.进一步工作是通过更多、更广的实例应用来完善NDORL语言成分,实现从采用NDORL描述的需求定义到用OOZE语言描述的形式功能规约自动转换的软件自动化系统.

致谢 特别感谢课题组的伊波博士、朱鸿博士、金凌紫博士、费宗铭博士等.

参考文献

1 Stephen F Fickes. Proceedings of the IEEE international symposium on requirements engineering. Los Alamitos, California: IEEE Computer Society Press, 1993.

2 Jawed Siddiqi. Proceedings of the first international conference on requirements engineering. Los Alamitos, California: IEEE Computer Society Press, 1994.

3 Pamela Zave. Proceedings of the second IEEE international symposium on requirements. Los Alamitos, California: IEEE Computer Society Press, 1995.

4 Coad P, Yourdon E. Object-oriented analysis, 2nd ed. New Jersey: Yourdon Press, Printice Hall, 1991.

5 Rumbaugh J *et al*. Object-oriented modeling and design. New Jersey: Yourdon Press, 1991.

6 Shlaer S, Mellor S. Object-oriented systems analysis: modeling the world in data. New Jersey: Yourdon Press, 1988.

7 Embly D *et al*. Object-oriented systems analysis: a model-driven approach. New Jersey: Yourdon Press, 1992.

8 徐家福,王志坚等.对象式程序设计语言.南京:南京大学出版社,1992.

9　吕建. 广谱规格说明语言 FGSPEC 的设计. 计算机研究与发展，1991，28(2)：1～5.

10　徐家福著. 系统程序设计语言. 北京：科学出版社，1983.

11　Antonio J S M de Alencar. OOZE：an object-oriented Z environment[Ph D. Dissertation]. Oxford：University of Oxford，1994.

ON THE DESIGN OF A GRAPHICAL OBJECT-ORIENTED REQUIREMENTS DEFINITION LANGUAGE

Zhang Jiazhong　Lü Jian　Wang Zhijian　Xu Jiafu

(*Institute of Computer Software　Nanjing University　Nanjing*　210093)

(*State Key Laborotary for Novel Software Technology　Nanjing University　Nanjing*　210093)

Abstract　This paper discusses some fundamental concepts of the software requirements definition, outlines the design rationale for a graphical object-oriented requirements definition language NDORL which is a semi-formal language that employs the object-oriented method to support requirements definitions, and also introduces its main constructs in detail. The language is concise, very intuitive, and powerfully expressive. It is, moreover, easy to be transformed to a formal functional specification language automatically.

Key words　Object-oriented, requirements model, requirements definition, requirements definition language.