Software Engineering and Applications 软件工程与应用, 2020, 9(5), 337-344
Published Online October 2020 in Hans. http://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2020.95038

Hans汉斯

面向对象分析的发展现状

冀付军，程　诺

首都经济贸易大学，管理工程学院，北京
Email: nuonuo0814@163.com

收稿日期：2020年9月22日；录用日期：2020年10月5日；发布日期：2020年10月12日

摘　要

面向对象是目前应用较为广泛的一种软件开发思想，该思想已经涉及到了软件开发的各个方面，如面向对象的分析、面向对象的设计以及面向对象的编程实现等。其中，面向对象分析与设计的开发语言以UML统一建模语言占据主导地位。本文通过对真实案例进行面向对象的分析，通过UML语言进行设计建模，进一步地体会到了面向对象分析与设计思想的优势和创新，并对面向对象未来的发展提出了若干建议。

关键词

面向对象，用例图，类图，时序图

The Development of Object-Oriented Analysis

Fujun Ji, Nuo Cheng

School of Management Engineering, Capital University of Economics and Business, Beijing
Email: nuonuo0814@163.com

Received: Sep. 22nd, 2020; accepted: Oct. 5th, 2020; published: Oct. 12th, 2020

Abstract

Object oriented is a widely used software development idea, which has involved all aspects of software development, such as object-oriented analysis, object-oriented design and object-oriented programming. Among them, UML is the dominant language for object-oriented analysis and design. This paper, through the object-oriented analysis of real cases, through UML language design modeling, further experiences the advantages and innovation of object-oriented analysis and design ideas, and puts forward some suggestions for the future development of object-oriented.

文章引用: 冀付军, 程诺. 面向对象分析的发展现状[J]. 软件工程与应用, 2020, 9(5): 337-344.
DOI: 10.12677/sea.2020.95038

Keywords

Object-Oriented, Use Case Diagram, Class Diagram, Sequence Diagram

1. 面向对象概述

(一) 概念

1. 面向对象

面向对象(Object-Oriented)是一种目前非常流行的软件开发思想，也是目前应用领域广泛、应用范围逐渐扩大的一种开发方法。面向对象是相对于面向过程而言的，其二者最根本的区别是思考、解决问题的方式不同。面向过程，顾名思义就是通过分析出待解决的问题需要的所有步骤，然后用函数将这些步骤一一实现，运行时只需要依次调用即可，总的来说是一个自顶向下的编程过程。而面向对象则是分析出待解决的问题中涉及的所有对象或者涉及到的所有功能，然后按照功能或者行为来进行设计。面向对象思想是从问题域出发，将问题域中客观存在的事物抽象成对象，每个对象都有其属性和操作[1]。

2. 面向对象的特征

面向对象具有四大特征：抽象性、封装性、继承性和多态性。

抽象性指的是将一类对象的共同特征提取出来，构造出一个类，类中包含有该对象的数据抽象和行为抽象，其中包含着对象的属性和操作，而不关注操作的具体细节。

封装性指的是将数据和代码捆绑到一起，对象的某些属性和功能是私有的，某些属性和功能则是公共的，系统可以被设计成对不同的用户拥有不同的访问权限，而用户只需要调用其中的功能，不需要看到内部的结构和代码。封装可以防止程序由于依赖性而带来的变动影响，同时还能够进行信息隐藏操作，增加了系统的安全性和稳定性。

继承性指的是子类可以使用父类的某些属性和功能，是一种层次模型。在层次模型中，上层具有通用性，下层具有特殊性，使得类的属性和方法能够从上层部分继承一些方法和变量。下层在继承之余还可以进行添加、删除和修改，这样能够有效提高工作效率，避免重复性工作。继承可以看作是一种从一般到特殊的演绎过程，这样设计可以减少信息冗余，方便数据库的维护和修正，非常适用于复杂的系统[2]。

多态性指的是面对不同的事物系统有不同的表现形式，多态机制能够使具有不同内部结构的对象共享相同的外部接口，这样能够降低代码的复杂度，真正的实现一个接口，多种方式。

(二) 面向对象发展历史

面向对象的发展历史基本可以分为三个阶段。

其雏形阶段可以追溯到 1960 年，当时的程序设计领域正面临着一种危机：在软硬件环境逐步复杂的情况下，如何更好的维护软件？对此，挪威计算中心开发的 Simula 67 首次引入了类的概念和继承机制，它是面向对象发展史上的第一个里程碑。随后设计出的众多语言都加入了相关概念，对抽象数据类型理论的发展起到了推动作用，并且均支持数据与操作的封装，例如 20 世纪 70 年代的 CLU、Ada、Pascal 以及 Alan Kay 设计出的 Flex 语言。直到 1972 年出现的 Smalltalk 72，标志着面向对象方法的雏形形成，

该语言正式使用了“面向对象”这个术语[3]。

面向对象的完善阶段标志性的语言是 1981 年推出的 Smalltalk 80。这是第一个完善的、能够实际应用的面向对象语言，被认为是面向对象发展史上最重要的一个里程碑。Smalltalk 80 基本具备绝大多数面向对象的基本概念以及支持机制。

随后面向对象的发展进入到了繁荣阶段，相继涌现了大批实用型的编程语言，如 C++、Object Pascal、Eiffel 等。现在则普遍采用语言、类库和可视化环境相结合的方式进行编程，如 Visual C++、Delphi 等。如今面向对象的思想也涉及到了整个软件的生命周期以及软件开发的各个方面，如面向对象分析(Object-oriented analysis, OOA)、面对对象设计(Object-oriented design, OOD)和面向对象编程(Object-oriented programming, OOP) [4]。

到 20 世纪 90 年代，面向对象分析与涉及方法已多达几十种，国际上也有很多面向对象产品出现，其中典型的以 Booch 方法、Coad 方法、OMT 方法和 UML 方法为主，Booch 方法最早出现，描述了面对对象的软件开发方法的基础问题，指出面向对象开发是一种根本不同于传统功能分解型的设计方法，功能分解只能通过问题空间的转换来获得，而面向对象更接近与对问题域中客观事物的理解来进行分析和设计。Coad 方法则结合了一部分面向对象概念和一些系统的开发经验，在对象、结构、属性、动作等方面提出了一套系统的原则。Coad 方法完成了从需求角度将类和类层次进行进一步的划分，基本初步体现出了类和类层次结构之间的特征。OMT 方法则是在 Coad 方法之后提出的一种新兴的面向对象的开发方法，该方法将真实世界的客观对象进行建模，然后围绕这些分析模型进行设计，这种建模设计方法将需求分析层面理解得更加深入，这样更有利于开发出更清晰、更人性化、更易维护的软件系统，该方法力求寻找问题求解的实际方法，为软件开发领域提供了一种实际的、高效的保证。

(三) 面向对象与面向过程

面向对象和面向过程都是软件程序分析与设计的方法。面向过程(Procedure Oriented)是以过程为核心的，它以什么正在发生为主要目标进行分析与设计，解决问题的一般思路就是计算得出问题的求解步骤，然后逐一按次序调用程序来解决问题、输出结果。面向对象是以对象为核心的，把构成问题事务分解成各个对象，更加关注问题解决过程中各个对象的行为。面向对象相较传统的面向过程分析优势在于提高了软件程序的可维护性，即可理解、可测试和可修改。面向对象分析与设计提高了软件程序的层次感，降低了模块之间的耦合程度，更加利于快速开发和降低开发成本。

据国际数据公司报告，在全球范围内的十余种新兴技术中，面向对象编程技术的应用位于第四，而其他应用范围较广的技术都与面向对象有关。从方法论意义上来讲，面向对象的思想可以说是一场方法论的革命，它打破了传统延续了多年的编程思想，从一个全新的角度重新分析系统——从对象出发，将客观事物的属性及其行为看作一个整体，通过抽象将一个一个这样的对象映射成类，运用到具体的建模过程中。面向过程的思想主要关注的是“如何做”的问题，以系统的数据和对数据的处理过程为中心进行分析和建模，而面向对象的思想主要关注的是“做什么”的问题，以封装好的数据、对数据操作的对象以及不同对象之间的相互关系为中心，是一种跟前者完全不同的分析、设计和编程思想。面向对象思想的提出，从根本上改变了人们看待软件系统甚至客观世界的角度，它不单单只针对软件开发这一应用领域，如今已经几乎覆盖了计算机系统及应用的所有领域，包括证券、医疗、企业管理、交通运输、数据库以及人工智能等等。

2. 面向对象分析与设计

统一建模语言(Unified Modeling Language, UML)是一种为面向对象系统的产品进行说明、可视化和编制文档的一种标准语言，是面向对象的建模工具，独立于任何程序设计语言。UML 是在不同角度描述

一个系统的功能和特征，也适用于描述软件各个开发阶段中的不同状态。通过建立需求模型、逻辑模型、设计模型和实现模型等，反应事物的实体、性质、关系、结构、状态和动态变化过程。

UML 在开发系统时常用到三种主要的模型：功能模型、对象模型和动态模型。其中，功能模型是从用户的角度展示该系统的主要功能，例如用例图；对象模型是采用对象、属性、操作等概念展示系统的结构，例如类图；动态模型展示的是系统运行时的动态行为，包括时序图、活动图和协作图等。

(一) 建立需求模型——用例图

1. 确定系统边界和对象

系统边界指的是一个系统所包含的所有系统成分跟系统之外的各种事物的分界线。对象是指现实生活中客观存在的事物，在面向对象分析中，是系统用来描述客观事物的封装体，是数据和动作的一个封装。对象不仅能够进行操作，并且能及时记录下操作的结果。选择对象的标准是首先保证这个对象能够参与程序的运行，其次保证这个对象有对象的基本特征，也就是能够进行封装，最后要保证对象必须是强相关的对象。

2. 绘制用例图

用例图包括参与者、用例以及用例与参与者之间的关系。这些关系包括参与者与用例之间的关联关系、参与者间的继承关系和用例之间的包含、继承和扩展关系。参与者通常分为三类：用户、设备或者外部系统。

3. 用例之间的关系

用例之间的关系包括包含关系、扩展关系和继承关系。

(二) 以医院病房系统为案例进行用例图分析

1. 医院病房监护系统的系统要求为：

1) 将病症监视器安装在每个病房，使病人的病症信号能够实时传送到中央控制系统进行分析和处理；

2) 在中心值班室内，值班护士使用中央监控系统对病人的情况进行监控，根据医生的要求进行打印病情报告、更新病历等操作；

3) 当病症出现异常时，系统会自动报警，并实时打印病人的病情报告，同时更新病历。

2. 分析系统的参与者

通过题目要求可以分析出主要参与者：

病人：提供病症信号；

值班护士：监视病情、查看病情报告、打印病情报告；

医生：查看病情报告、打印病情报告、查看病历、打印病历。

3. 分析系统的详细功能

病症监护：将病人反馈的病症信号格式化后实时传送的中央监护；

中央监护：对比信号，将收到的信号与标准病症信号进行对比；报警，若通过对比发现信号异常则自动报警；更新病历，若发现信号异常则自动更新病历；打印报告，若发现信号异常则自动打印病情报告；

病情报告管理：医生和护士均可以查看、打印病情报告；

病历管理：更新病历，信号异常时对病历进行自动更新、定期对病历进行更新；查看病历，医生可以查看病历；打印病历，医生可以打印病历；

自动更新病历。

根据以上分析，绘制出的用例图如图 1 所示。

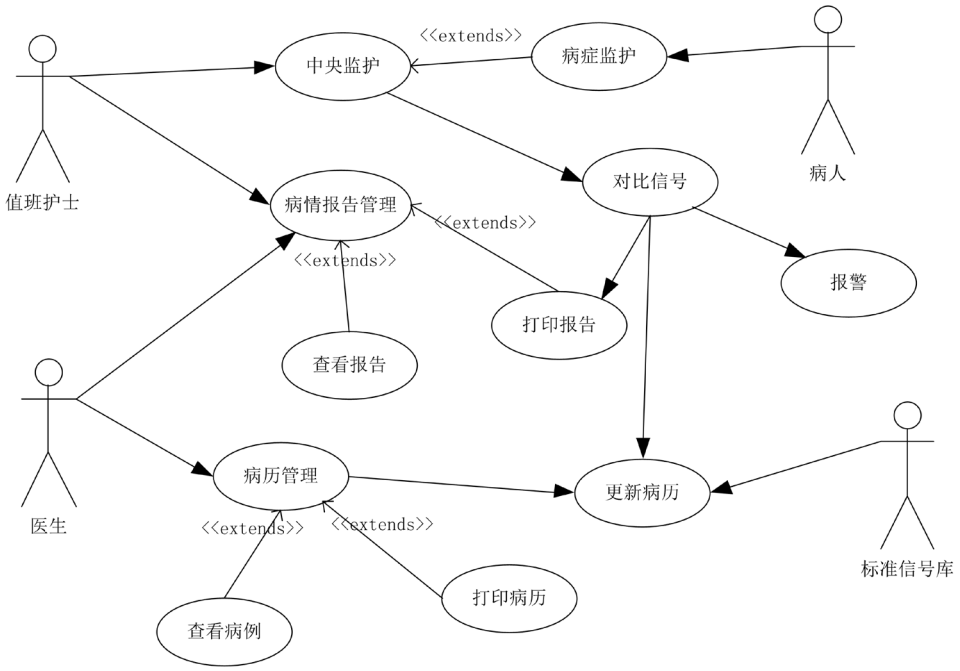

Figure 1. Use case diagram of ward monitoring system
图 1. 病房监护系统用例图

(三) 建立基本模型——类图

1. 发现对象，定义他们的类

类是现实世界中实体的形式化描述，实体的数据和操作封装起来，多个对象实体抽象成一个类。类实际上是一种数据类型，具有属性和方法，属性由数据结构进行描述、方法用操作名和实现该操作的方法进行描述。类图描述的是系统的静态模型类型，有助于在正式编写代码进行开发之前，对系统整体有一个全面的认识。类图表示类、接口和他们之间的协作关系。

2. 类与类之间的结构关系

结构是指所要研究的问题域的复杂度和连接关系，也就是类和对象之间的关系。类和对象之间的关系主要有四种：关联关系、依赖关系、泛化关系和实现关系。其中，关联关系还分为双向关联、单向关联、自关联、多重关联、聚合关系和组合关系。

泛化关系也称继承关系，指的是一个子类或者子接口继承另外的一个父类或者父接口的功能。如果继承的父类是抽象类，并且有抽象方法，那么抽象方法必须在子类中实现。实现关系指的是一个类实现一个或多个接口的功能，是类与接口之间最常见的关系。在 UML 图中通常用一个带空心三角箭头的虚线，从类指向实现的接口。依赖关系指的是两个相对独立的对象，一个对象负责构造了另一个对象的实例或者依赖另一个对象的服务时，二者表现出依赖关系，也就是一个类使用了另一个类的方法参数，二者就是依赖关系。关联关系是一种比依赖关系更强并且是存在长期性的一种关系，当一个类的实例使用了另一个类的实例时，二者形成关联关系。聚合关系是关联关系的一种，其耦合度更强，体现的是整体与部分的关系，是一种“整体－个体”之间的相互关系，在 UML 中通常使用空心菱形实线箭头表示。组合关系是关联关系的一种特殊形式，也成为强聚合关系，同样体现了整体与部分之间的关系，但区别在于，组合关系中整体与部分是不可分割的，也就是加入整体的生命周期结束，意味着部分的生命周期也结束，在 UML 中通常使用实心菱形和实线箭头表示。

(四) 分析建模在线商城的类图

以在线商城为研究对象，得出该系统的用户主要包括顾客和管理员两类，顾客分为普通用户和注册成会员的用户。普通用户可以进行商品查询、浏览商品详情等基本浏览网页操作，会员用户可以在登录后进行添加购物车、移除购物车内商品、下订单以及付款等操作。网站管理员负责系统后台的维护，包括管理会员信息、管理商品信息、管理订单、订单维护、订单执行发货等操作。

通过以上分析，可以总结出的类有：用户类、管理员类、顾客类、订单类、商品信息维护类。在线商城的类图如图 2 所示。

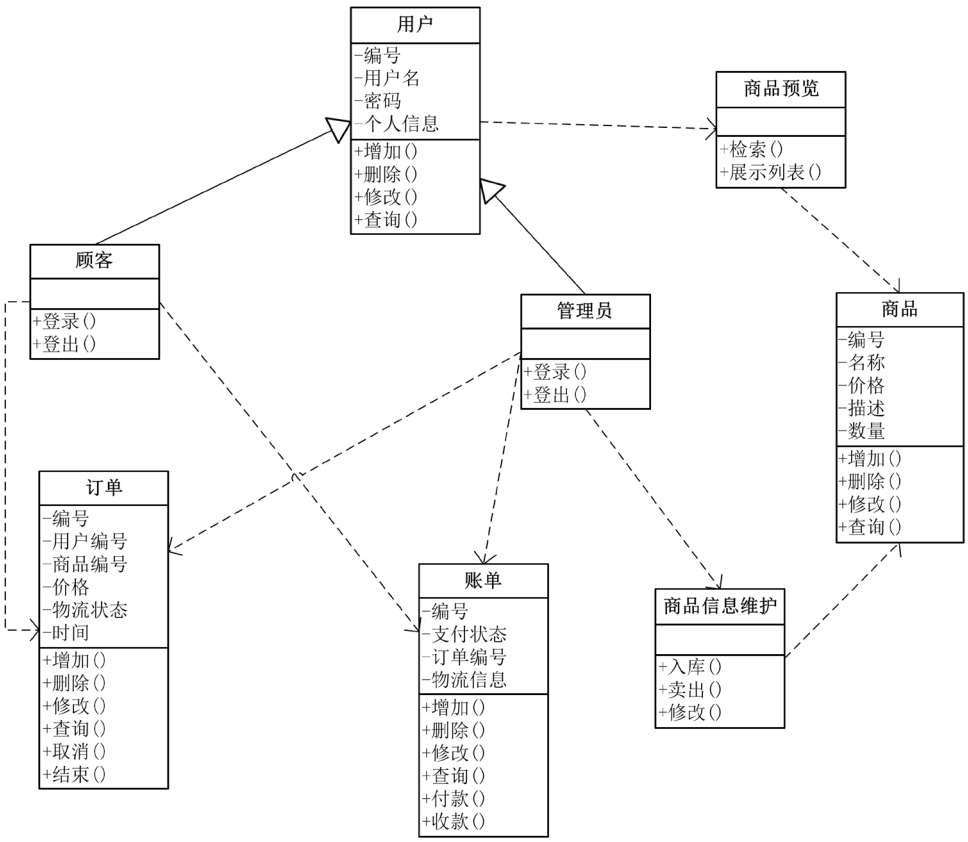

Figure 2. Class diagram of online mall system
图 2. 在线商城系统类图

(五) 确定动态模型——时序图

1. 组成元素

一个完整的时序图需要包含七种元素：Actor (角色)、Object (对象)、Life Line (生命线)、Activation (动作)、Message (消息)、自关联消息以及组合片段。其中角色主要指的是系统角色，可以是人或者系统的抽象表示。对象位于时序图顶部，用矩形表示，通常对象的命名方式可以同时显示对象名和类名，也可以只显示其中一个。生命线是时序图中每个对象下面一条垂直的虚线，代表着对象的时间线。动作表示时序图中对象在某段时间内进行的操作。

2. 信息

消息指的是对象在进行操作的同时所传递的信息，这种信息分为同步、异步和返回信息。同步信息用实线和实心箭头表示，代表消息随着操作同时传送给信息的接收者并且等待信息返回，所谓同步指的

是操作和等待信息的同步。异步信息相对于同步信息，指的是消息传送给信息的接收者后继续进行自己的活动，不再等待返回消息，消息的发送者和接收者是同时并发工作的，异步信息用一条实线和大于号表示。而返回信息表示的是从过程调用返回信息，用虚线和小于号表示。

自关联消息指的是方法的自身调用或者一个对象内的方法调用另一方法，在 UML 中用使用半闭合的矩形和实心箭头表示。

3. 结构

时序图通常描述的是对象之间的交互情况，也就是描述在一个完整的操作中，信息在对象之间的传递情况。时序图有两个坐标轴，横坐标显示对象，用矩形表示，矩形框中写有对象名或者类名；纵坐标表示时间，每一个对象下面的纵向虚线用来表示对象的执行情况，这条虚线称作生命线。当对象接收到消息时，该对象相对应的操作就会被调用，这一过程称作激活。

4. 创建步骤

时序图的创建大概分为六个步骤：

1) 确定交互过程的上下文；
2) 识别参与活动的交互对象；
3) 为参与活动的对象设置生命线；
4) 按照实际活动顺序，依次从初始消息画到最后；
5) 标记消息发生的时间点；
6) 说明时间约束的地点。

(六) 基于在线商城案例分析时序图

通过用例图和类图的分析，我们可以确定了参与者和基本的活动，在静态建模的基础上，UML 建模的下一步就是根据分析出系统的基本活动绘制出时序图，从动态建模角度将进一步分析系统的功能实现和结构。在线商城的案例中，我们可以通过分析顾客购买商品的行为，画出相应的时序图，见图 3。

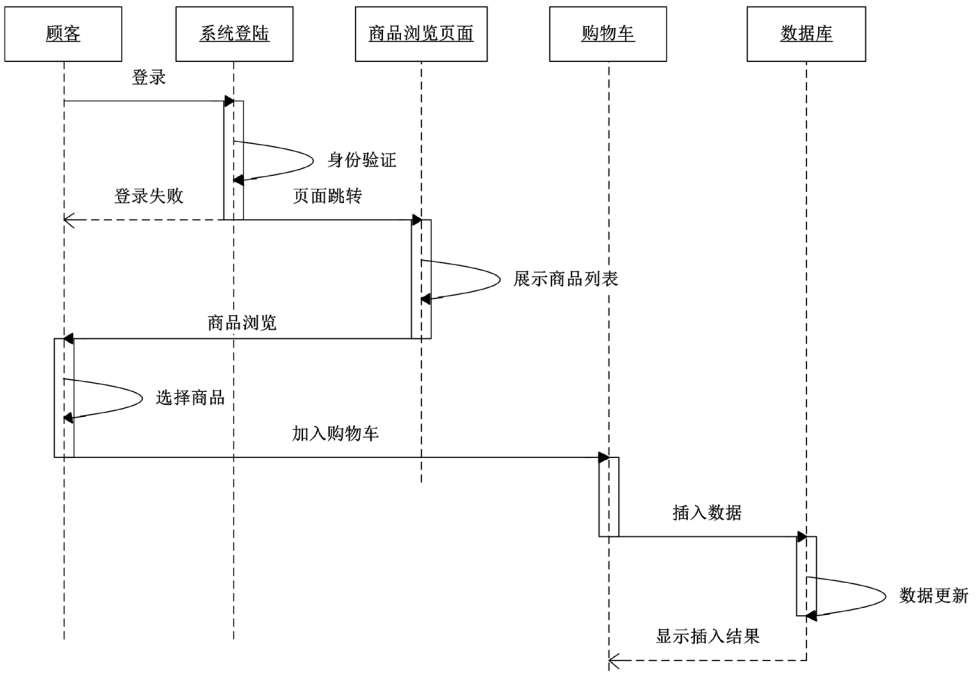

Figure 3. Sequence diagram of customer product preview
图 3. 顾客商品预览时序图

分析可知，该行为的参与者是顾客，要想做出完整的购买商品的行为，需要进行登陆系统、浏览商品、将商品加入购物车。

(七) 静态结构和动态行为的组合——协作图

协作图又称通信图，表示不同事物之间相互协作完成一个复杂的功能，是面向对象动态建模的另一种图。与时序图不同，协作图展示的是对象之间的关系，有利于分析出所有对象之间的相互影响关系，通常适用于过程设计，适合用来描述少量对象之间的简单交互。

协作图通常由三种元素组成：对象、连接和消息。对象由矩形框表示，代表协作图中用来进行交互的角色，与时序图中对象的概念相近。连接表示对象之间相互通信的路径，用一条连接两个角色的实线表示。消息代表协作图中对象之间通过连接所传递的信息，连接的实线上方标有消息的序列号和对象之间发送的消息，其中，一条消息就会触发接收对象中的一项操作。

协作图中的消息类型基本与时序图相同，只不过需要为消息添加顺序号以说明交互过程中的时间顺序。在 UML 中，时序图可以与协作图进行相互转换。

3. 结语

面向对象思想打破了传统的软件开发思维模式，通过封装、继承等操作，将客观对象封装成类进行进一步开发，但其效率问题始终存在不足。当系统的规模足够庞大时，开发效率的弊端就显现了出来，并且这种开发思想要求程序员对类库较为熟悉，如果开发设计人员对类库的掌握不够，设计过程存在漏洞，这就可能导致整个开发过程的失败。随着技术的革新和对缺陷的不断修复，面向对象程序设计会成为推动人工智能、科技金融、自动化、无人驾驶等众多领域的主力军，用自身的优势和特点进一步推动我国科学技术的发展进步。

参考文献

[1] 张菲, 郭庆峰, 张帅, 高阿曼. 基于 UML 状态图测试用例生成的策略研究[J]. 计算机与数字工程, 2020(2): 72-74.

[2] 栾家伟, 吴陈. 基于 UML 状态图测试用例生成的策略研究[J]. 计算机与数字工程, 2020, 48(2): 409-411+432.

[3] 张日如. 基于 UML 的图书管理系统的设计[J]. 电脑知识与技术, 2019, 15(10): 81-83.

[4] 袁中臣, 马宗民. 基于语义和结构的 UML 类图的检索[J]. 东北大学学报(自然科学版), 2020(1): 23-28.