中图分类号：TK172　　论文编号：1028703 10-0003
学科分类号：081101

硕士学位论文

基于 Rhapsody 的飞行控制系统数字化设计研究

研究生姓名　何火军
学科、专业　控制理论与控制工程
研 究 方 向　系统建模与仿真
指 导 教 师　曹云峰 研究员

南京航空航天大学

研究生院　自动化学院

二〇一〇年一月

Nanjing University of Aeronautics and Astronautics
The Graduate School
College of Automation Engineering

Research on Digitized Design of Flight Control System Based on Rhapsody

A Thesis in

Control Science and Engineering

by

He Huojun

Advised by

Prof. Cao Yunfeng

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

Master of Engineering

January, 2010

承诺书

本人声明所呈交的硕士学位论文是本人在导师指导下进行的研究工作及取得的研究成果。除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得南京航空航天大学或其他教育机构的学位或证书而使用过的材料。

本人授权南京航空航天大学可以将学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本承诺书）

作者签名：何火军

日　　期：2010年3月18日

摘 要

本课题的研究工作依托于“综合系统顶层设计与数字仿真软件包”项目，以某型无人机飞行控制系统为研究对象，重点研究了该系统的数字化设计。通过运用面向虚拟样机的数字化设计方法，在虚拟样机平台上设计了某型无人机飞行控制系统的虚拟样机，即建立数字化模型，从而实现了飞行控制系统的数字化设计。论文主要完成的工作包括以下几方面：

1. 简要介绍了课题的研究背景和国内外的研究现状，同时介绍了课题的来源和研究意义。

2. 首先在总体上提出了某型无人机飞行控制系统的面向虚拟样机的数字化设计方法，并设计出适用于该方法的虚拟样机平台；再根据某型无人机的要求，设计出某型无人机飞行控制系统的基本结构；然后设计出飞行控制系统所包含的功能模块；最后重点给出各模块的具体功能和性能要求。

3. 研究将设计模式用于 UML 建模的好处以及如何将设计模式用于 UML 建模。在对各种设计模式进行研究的基础上，选用适用于无人机飞行控制系统建模的设计模式。

4. 在虚拟样机平台的 Rhapsody 软件模块上，运用模型驱动式柔性开发方法设计某型无人机飞行控制系统的数字化模型（即虚拟样机），并进行仿真验证。

5. 在虚拟样机平台的 Simulink 软件模块上，建立某型无人机的动力学模型。将动力学模型导入到 Rhapsody 工程，构成一个带有动力学模型的飞控系统数字化模型，并进行仿真验证。

关键词：无人机，飞行控制，数字化设计，仿真，设计模式，统一建模语言，虚拟样机

Abstract

This thesis was based on the research program, High-level Design and Digital simulation for Integrated System. Digital Design on a certain type UAV (Unmanned Aerial Vehicles) flight control system was studied. In order to achieve Digital Design on flight control system, Digital design methods oriented to virtual prototype was used to design Virtual Prototype of a certain type UAV flight control system. The major research work was as following.

1、This paper briefly described the research background and current situation of the research program at home and abroad. Also, the program sources and research significance were introduced in this paper.

2、Firstly, digital design method oriented to virtual prototype was put forward and Virtual Prototype Platform for UAV flight control system was designed. Secondly, basic structure of the UAV flight control system was designed according to the requirements of a certain UAV. Thirdly, Function modules of the UAV flight control system were designed. Finally, specific features and performance requirements of the UAV flight control system was given.

3、The benefits of design patterns as well as how to apply design patterns to UML modeling were discussed. Based on the research on various design patterns, the suitable design pattern was chosen to match UAV flight control system modeling.

4、UAV flight control system was developed with model-driven & Flexible development approach, which is based on Rhapsody software module of the Virtual Prototype Platform. The model was verified by the simulation.

5、UAV Kinetic model was developed with the Simulink module of the Virtual Prototype Platform. The model was verified by the simulation. The Kinetic model and continuous UAV flight control system model were aggregated into an organic whole. A complete model of the system was verified by the simulation.

Keywords: UAV, flight control system, digital design, simulation, design patterns, UML, virtual prototype

目 录

第一章 绪论 1
1.1 课题研究背景 1
1.2 国内外相关研究工作 2
1.3 课题来源及研究意义 5
1.3.1 课题来源 5
1.3.2 课题研究意义 6
1.4 本文的内容结构安排 7
第二章 基于 Rhapsody 的虚拟样机平台的搭建 9
2.1 飞行控制系统设计方法 9
2.1.1 飞行控制系统传统设计方法 9
2.1.2 面向虚拟样机的数字化设计方法 10
2.2 虚拟样机平台的设计 11
2.2.1 虚拟样机平台的作用 11
2.2.2 虚拟样机平台的基本功能要求 11
2.2.3 虚拟样机平台的搭建 12
2.3 飞行控制系统的功能需求 14
2.4 飞行控制系统的结构组成 15
2.5 飞行控制系统的基本工作原理及控制策略 17
2.6 本章小结 18
第三章 设计模式的研究 19
3.1 设计模式概述 19
3.1.1 设计模式的基本概念 19
3.1.2 设计模式的描述 20
3.1.3 设计模式的分类 20
3.2 设计模式与框架的区别 21
3.3 设计模式的应用研究 22
3.3.1 设计模式的优点和研究意义 22
3.3.2 设计模式应遵循的原则 22
3.4 应用设计模式设计飞行控制系统 23

3.4.1 简单工厂模式的应用研究 23
3.4.2 工厂方法模式的应用研究 26
3.4.3 适配器模式的应用研究 27
3.4.4 外观模式的应用研究 29
3.4.5 状态模式的应用研究 31
3.4.6 MVC 模式的应用研究 33
3.5 设计模式指导 UML 建模 34
3.5.1 UML 建模方法的不足 34
3.5.2 设计模式在建立 UML 模型中的应用 35
3.5.3 设计模式指导 UML 建模的步骤 35
3.6 本章小结 36
第四章 某型无人机飞行控制系统的虚拟样机设计 37
4.1 模型驱动式柔性开发方法 37
4.1.1 模型驱动方法 37
4.1.2 柔性开发模式 37
4.1.3 模型驱动式柔性开发方法 38
4.2 系统分析 39
4.2.1 用例建模的主要目标 39
4.2.2 用例建模的步骤 39
4.2.3 建立系统的需求模型 41
4.3 系统设计 43
4.3.1 系统的类与对象的确定 43
4.3.2 系统静态模型的设计 44
4.3.3 系统动态模型的设计 49
4.3.4 模型的阶段性检测 56
4.4 系统测试 57
4.4.1 语法和语义的检测 58
4.4.2 基于人机交互界面的仿真 58
4.5 代码编写 61
4.6 本章小结 63
第五章 某型无人机动力学模型的导入 65
5.1 某型无人机动力学模型的设计 65

5.2 将动力学模型导入 Rhapsody 工程 69
5.2.1 建立导入到 Rhapsody 工程的 Simulink 动力学模型 69
5.2.2 将修改后的动力学模型导入 Rhapsody 工程 71
5.3 系统虚拟样机的仿真验证 78
5.3.1 语法和语义的检测 78
5.3.2 基于 DOS 界面的仿真 79
5.4 生成模型的代码 79
5.5 本章小结 79
第六章 总结与展望 81
6.1 工作总结 81
6.2 前景展望 83
参考文献 85
致 谢 89
在学期间的研究成果及发表的学术论文 91

图表清单

图 1.1 飞行控制系统传统设计流程 1
图 1.2 建模与仿真的基本组成与相互关系 4
图 2.1 飞行控制系统传统设计方法流程图 9
图 2.2 面向虚拟样机的数字化设计方法的流程图 10
图 2.3 虚拟样机平台 12
图 2.4 飞行控制系统结构图 15
图 2.5 飞行控制系统控制回路图 15
图 2.6 飞行控制系统详细结构图 16
图 3.1 简单工厂模式的类图结构 24
图 3.2 飞控计算机与传感器之间的类图结构 24
图 3.3 飞控计算机与舵机之间的类图结构 25
图 3.4 飞控计算机与电位器之间的类图结构 25
图 3.5 工厂方法模式的类图结构 26
图 3.6 舵机电位器之间的类图结构 27
图 3.7 适配器模式的类图结构 28
图 3.8 电源适配的类图结构 28
图 3.9 外观模式的类图结构 29
图 3.10 子系统与外部系统的类图结构 30
图 3.11 飞控计算机与外部交互的类图结构 30
图 3.12 状态模式的类图结构 31
图 3.13 控制模态的结构关系图 32
图 3.14 控制模态的状态图 32
图 3.15 MVC 模式应用图 33
图 4.1 柔性开发模式 38
图 4.2 模型驱动式柔性开发流程 38
图 4.3 飞行控制系统的需求模型 41
图 4.4 飞控计算机系统的需求模型 42
图 4.5 无人机的需求模型 43
图 4.6 无人机静态结构图 45
图 4.7 飞行控制系统顶层设计结构图 45
图 4.8 飞控计算机的结构图 46

图 4.9 电位机、飞控计算机和舵机之间的类图 47
图 4.10 飞控计算机与传感器之间的类图 47
图 4.11 电源转换的类图 48
图 4.12 飞控系统与硬件平台的接口图 48
图 4.13 VC 软件的接口图 48
图 4.14 飞控计算机各模块的类图 49
图 4.15 无人机的行为功能图 50
图 4.16 飞行状态的子状态图 51
图 4.17 无人机飞行状态的状态图 52
图 4.18 CPU 模块的状态图 53
图 4.19 模拟输入通道的状态图 53
图 4.20 串行通道的状态图 54
图 4.21 定时计数器的状态图 54
图 4.22 离散通道的状态图 55
图 4.23 前置信号处理模块的状态图 55
图 4.24 电源转换模块的状态图 56
图 4.25 无人机横侧向的初始状态 57
图 4.26 数字仿真的 DOS 界面显示 57
图 4.27 系统虚拟样机的编译结果 58
图 4.28 交互机制图 59
图 4.29 人机交互界面图 59
图 4.30 操纵 VC 交互界面 60
图 4.31 statechart 响应图 61
图 4.32 自动生成的 C++代码的头文件和源文件 62
图 4.33 自动生成的类 UAV 的代码 63
图 5.1 无人机动力学模型的详细设计流程图 65
图 5.2 无人机动力学模型的模块图 68
图 5.3 导入 Rhapsody 工程的无人机动力学模型的模块图 71
图 5.4 新建工程对话框 72
图 5.5 对象模型图 72
图 5.6 Features 对话框 73
图 5.7 Motor 类的对象图 73
图 5.8 导入 Simulink 模型的对话框 74
图 5.9 选择 Simulink 模块图导入的对话框 74
图 5.10 导入 Simulink 模块图后的对话框 75

图 5. 11 导入 Simulink 代码后的对话框 75
图 5. 12 导入 Simulink 模型后的对象模型图 76
图 5. 13 更改后的 FCC 类的对象模型图 76
图 5. 14 导入 Simulink 动力学模型后的对象模型图 78
图 5. 15 系统虚拟样机的编译结果 78
图 5. 16 航向保持控制仿真模块的输出 79
图 6. 1 半物理仿真平台的总体功能结构图 82
图 6. 2 飞行控制系统硬件平台的总体结构 83
表 1. 1 建模与仿真的历史发展 4
表 3. 1 设计模式的分类 21

注释表

1. 缩略词与名称

基本缩略词	全称	中文名称
UAV	Unmanned Aerial Vehicle	无人机
GPS	Global Position System	全球定位系统
UML	Unified Modeling Language	统一建模语言
MDD	Model-Driven Development	模型驱动开发
SysML	Systems Modeling Language	系统建模语言
OXF	Object Executable Framework	可执行对象框架
RTOS	Real-time operating system	实时操作系统
MVC	Model-Veiw-Controller	模型/视图/控制器
MDA	Model-Driven Approach	模型驱动方法

2. 基本符号与名称

	序号	数学符号	描述	单位
构造参数	1	m	质量	千克(kg)
	2	I_x	绕OX轴的转动惯量	千克·米^2($kg\cdot m^2$)
	3	I_y	绕OY轴的转动惯量	千克·米^2($kg\cdot m^2$)
	4	I_z	绕OZ轴的转动惯量	千克·米^2($kg\cdot m^2$)
	5	I_{xz}	惯性积	千克·米^2($kg\cdot m^2$)
力和力矩	1	G	重力	牛(N)
	2	T	拉力	牛(N)
	3	D	阻力	牛(N)
	4	L	绕OX轴的力矩	牛·米($N\cdot m$)
	5	M	绕OY轴的力矩	牛·米($N\cdot m$)

	6	N	绕OZ轴的力矩	牛·米($N\cdot m$)
状态变量	1	V	地速	米/秒(m/s)
	2	$\bar{V}$	当前速度与基准速度的比V/V_0	无量纲
	3	α	迎角	弧度(rad)
	4	β	侧滑角	弧度(rad)
	5	γ	爬升角	弧度(rad)
	6	ϕ	滚转角	弧度(rad)
	7	θ	俯仰角	弧度(rad)
	8	ψ	偏航角	弧度(rad)
	9	p	滚转角速度	弧度/秒 (rad/s)
	10	q	俯仰角速度	弧度/秒(rad/s)
	11	r	偏航角速度	弧度/秒(rad/s)
	12	H	高度	米(m)
控制输入变量	1	δ_e	升降舵	弧度(rad)
	2	δ_a	副翼	弧度(rad)
	3	δ_T	发动机状态	百分比(%)
	4	δ_r	方向舵	弧度(rad)

第一章 绪论

无人驾驶飞机(Unmanned Aerial Vehicle)简称无人机(UAV)，又被称为空中机器人，是一种由动力驱动、机上无人驾驶、可重复使用的航空器[1]。无人机结构简单、造价低廉、生存能力强、机动性好、能够很好地完成有人驾驶飞机不能完成的任务。无人机的研制受到了各国政府的高度重视。飞行控制系统是现代无人机的核心[2]，对飞行控制系统的研究也就成为无人机研制的重点所在。本章主要介绍课题的研究背景和国内外研究现状，同时介绍了课题的来源和研究意义，并对所做的研究工作和论文的结构做了一个概括性的介绍。

1.1 课题研究背景

飞行器控制是当前控制工程中的一个专门的、重要的高技术领域。飞行控制技术的产生和发展决定着飞行器的产生和发展。纵览当今世界，飞行器的发展已经涉足到交通、运输、航空航天、国防等诸多领域，有着无可替代的地位。进行飞行控制系统的分析和设计是飞行器设计中的重要一环。

飞行控制系统传统的开发方法主要以瀑布模型为代表，其设计流程从方案设想到验证分析共有32个步骤[3]。 图1 .1显示了其中的部分步骤[4]。

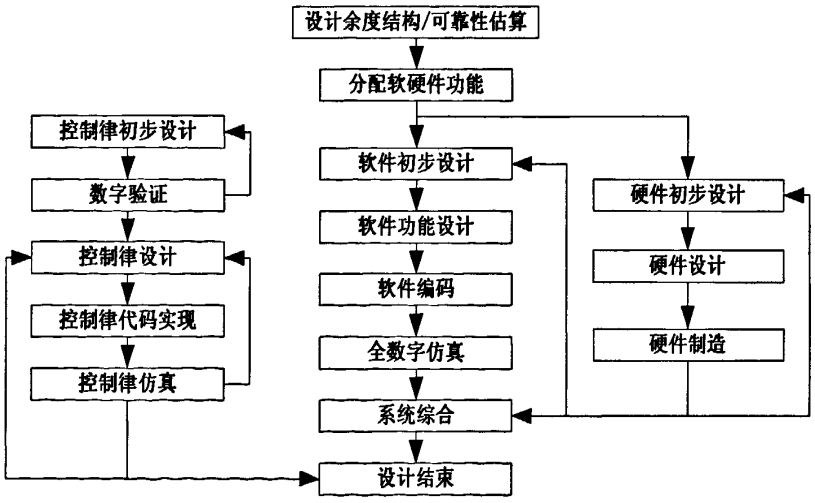


图 1. 1 飞行控制系统传统设计流程

设计流程从制定系统全部设计规范开始，规划着系统的全部功能。在综合考虑可靠性要求、软硬件各自承担的功能等基础上，进行了系统的顶层结构设计。顶层结构设计完成之后，分别进入软件模块和硬件模块的设计和测试。在同一技术状态循环内，控制律、软件和硬件的设计是并行的，各自独立的，直到系统数字仿真和综合模拟试验时才进行综合和测试。可以看出，

传统的系统设计流程存在以下的缺陷[5]：

（1）信息孤岛的出现。系统设计总体上是串行单向的，其设计流程都是相对独立的过程，各自成为了信息孤岛。设计一旦出现更改，将带来成本的大幅增长，非常不利于飞行控制系统的发展。

（2）文档驱动的缺陷。传统开发方法主要是文档驱动的。文档主要是以文字描述的，文字描述最大缺点是具有二义性。对同样一个功能或行为的描述，不同的阅读者可能有不同的理解，造成的后果就是不同阶段的开发人员对系统功能、行为的理解不完全一致。

（3）系统规模增大和规范的不可测试。以无人机为例，随着无人机技战指标的要求越来越高，飞行控制系统规模越来越大，系统中各部件、子系统之间的交互越来越多，其复杂度超出了人所能控制的范围。传统的文档式系统规范是不可测试的，而规范内部不可避免的隐藏着歧义和错误，依据规范构造出的系统可能不满足用户的需求，甚至系统本身就不能正常运行。在传统的系统开发方法中，除控制律具有两个修改循环外，由于缺少测试手段，其余部分均要等到数字仿真和系统综合时才能进行测试和验证。设计的反馈周期长，设计时需要的背景知识库丰富，设计一旦出现更改，将带来成本的大幅增长。因此迫切需要在系统开发早期即顶层设计阶段就能对系统的功能、行为规约进行测试，消除歧义和错误。

（4）缺乏快速反应的能力。系统规范的不完善，常常导致复杂系统的设计要经历很多次的修正和改进，系统一项设计内容的修改也很难在其它地方自动及时地得到刷新或更正。设计文档完全依赖于人为编写，无法自动生成。因此，系统开发效率不高，研制周期较长（普遍达数年），新产品研发缓慢，这些都越来越难以适应市场对产品的要求。

（5）维护量大。传统的系统维护工作集中在系统开发的最后某个阶段，工作量大。

传统设计方法的不足，促使新的数字化设计方法的出现，以解决传统设计方法的缺陷。在这种形势下，诞生了我的课题：基于 Rhapsody 的飞行控制系统数字化设计研究。

1.2 国内外相关研究工作

伴随着信息时代的来临，全球进入了数字化时代。数字化时代是数字化技术在生产、生活、经济、社会、科技、文化、教育和国防等各个领域不断扩大应用并取得日益显著效益的时代。一系列数字概念如数字图书馆、数码城等与日俱增，数字化对企业研发、设计、制造和试验所带来的冲击是巨大的。

数字化技术作为先进设计制造技术的基础核心，国外自 20 世纪 70 年代以来，经历了从点(数字化单项技术应用)到线(型号研制的全过程数字化)到面(数字化企业)的发展。目前，从联合攻击战斗机（JSF）、美国国家航天局的载人探测飞行器等项目研制所采用的数字化技术可看出，国外数字化技术的研究应用已经达到了较高的水平和规模。主要体现在：虚拟样机在型号研制

中得到普遍应用，集成化的数字化虚拟产品研制环境的显现，异地联合数字化产品研制模式的产生等。

在国内，数字化技术研究与应用开始于20世纪70年代末、80年代初。在国家“863”、“973”等重大项目的支持下，各研究院所、高等院校和航空航天等工业部门在数字样机、多学科优化设计、并行工程、企业信息集成技术、虚拟制造等方面开展了大量理论和应用研究。主要体现在：数字化单项技术普遍应用；集成与协同技术应用初见成效；数字化核心技术研究成果初显；数字化支撑平台体系尚待提升等。

未来5—10年，数字化技术将继续向集成化、网络化、虚拟化、智能化的方向发展。以NASA、波音公司、洛克希德·马丁公司为代表的国外军工企业将逐步实现数字化企业的发展目标，主要有三个方向：全产品数字化、全过程数字化、全方位数字化[6]。

数字化技术与各种专业技术相融合形成了各种数字化专业技术，如数字化制造技术、数字化设计技术、数字化视听技术等。其中数字化设计技术的诞生标志着设计理念发生了革命性的变化。数字化设计技术是指将计算机技术应用于产品设计领域。最初的计算机是被设计用来进行科学计算的，随着计算机和外部设备的发展，很快人们发现它还可以用在其他的领域，其中之一是用于产品辅助设计。计算机辅助设计(computer aided design, CAD)便是数字化设计最初的应用。随着研究的深入，数字化设计已经不仅仅局限在CAD。对于数字化设计技术比较完备的定义是：数字化设计技术是基于产品描述的数字化平台，建立基于计算机的数字化产品模型，并在产品开发全程采用，达到减少或避免使用实物模型的一种产品开发技术。由数字化设计方法设计的数字式飞行控制系统自20世纪70年代初大规模集成电路出现后，便得到了迅速发展。

虚拟样机技术(virtual prototyping, VP)近年来作为数字化设计的一个重要分支，得到了人们的普遍关注。目前，虚拟样机技术在一些较发达国家，如美国、德国、日本等已得到广泛的应用，应用领域从汽车制造业、工程机械、航空航天业、造船业、机械电子工业、国防工业、通用机械到人机工程学、生物力学、医学以及工程咨询等很多方面。我们熟知波音777飞机是世界上首架以无纸方式研发及制造的飞机，其设计、装配、性能评价及分析就是采用了虚拟样机技术。这不但使研发周期(8年缩短为5年)大大缩短，研发成本大大降低(设计费用降低94%，设计更改降低93%)，而且确保了最终产品一次装机成功。通用动力公司1997年建成了第一个全数字化机车虚拟样机，并行地进行产品的设计、分析、制造及夹具、模具工装设计和可维修性设计。

在国内，虚拟样机技术的应用尚处于起步阶段，主要是引用国外的先进技术，但也取得了很大的进展，代表性的有中航第一飞机研究院成功推出了国内首架飞机全机规模电子样机；863项目“月球表面探测机器人方案研究”则运用虚拟样机技术构造虚拟月球面计算仿真环境，并对涉及到的多项关键技术进行了深入研究，取得了很好的成果；军事院校已经把虚拟样机技术应用到火炮等武器的研制生产当中。

虚拟样机技术的本质是建模与仿真，而建模与仿真技术却是实现虚拟样机的主要途径。建模与仿真技术的不断发展能够推动虚拟样机技术的不断发展。建模与仿真是指构造现实世界实际系统的模型和在计算机上进行仿真的有关复杂活动，它主要包括实际系统、模型和计算机等三个基本部分，同时考虑三个基本部分之间的关系，即建模关系和仿真关系，如图1.2所示。建模关系主要研究实际系统与模型之间的关系，它通过对实际系统的观测和检测，在忽略次要因素及不可检测变量的基础上，用数学方法进行描述，从而获得实际系统的简化近似模型。仿真关系主要研究计算机的程序实现与模型之间的关系，其程序能被计算机所接受并在计算机上运行[7]。

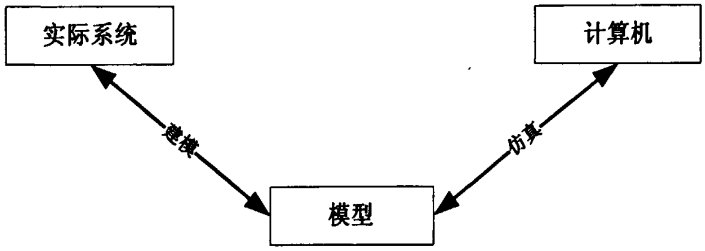


图 1.2 建模与仿真的基本组成与相互关系

建模和仿真是紧密联系的，其中，建立的模型既是系统仿真的重要内容，又是系统仿真的前提条件。当前系统仿真主要包括以下几个方面的研究：①研究新的建模方法；②创造先进的建模环境，如计算机辅助建模，利用先进仿真语言建模等；③重视专家系统在建模中的作用，不断完善专家系统的建模知识库，并致力于研制专用于系统建模的专家系统；④加强建模的薄弱环节，如模型校验、验证等的技术研究工作[8]。

建模与仿真的历史发展进程如表1.1[9]

表 1.1 建模与仿真的历史发展

年　　代	发 展 的 主 要 特 点
1600-1940年	在物理学基础上的建模
20世纪40年代	电子计算机的出现
20世纪50年代中期	仿真用于航空领域
20世纪60年代	工业操作过程的仿真
20世纪70年代	包括经济、社会和环境因素的大系统仿真
20世纪70年代中期	系统与仿真的结合，系统仿真与更高级的决策结合
20世纪80年代中期	集成化建模与仿真环境
20世纪90年代	可视化建模与仿真，虚拟现实仿真，分布交互仿真

从80年代起，国外一些国家对军事电子信息系统和武器装备的建模与仿真，研究得十分广

泛和深入，既有方法论的研究，又有实战的仿真。90 年代以来，建模与仿真技术受到发达国家的高度重视，尤其是美国和日本。美国国防部一直将建模与仿真技术作为“国防关键技术计划”的重点项目，尤其是经海湾战争实战验证后，更增强了对该方面的组织领导，不仅颁布了“国防建模及仿真倡议”，批准了“建模与仿真管理计划”，新组建了“国防建模与仿真办公室”，还提出了新的建模与仿真的投资战略[10]。近年来，仿真技术向着分布式交互仿真的方向发展，美国是发展分布式交互仿真最早的国家，目前技术水平及应用水平处于世界领先地位，分布式交互仿真是在飞行仿真器网络技术的基础上发展起来的，它以通信网络为基础，又包含多个武器平台及各种虚拟作战环境的交互作用。目前，在武器系统仿真技术研究方面，美国已研制出系列化的飞行仿真器[11]和阵列式目标仿真器，研制出万亿次每秒的高速数字机用于核武器的仿真研究，并将仿真技术用于战争的预先推算。

我国建模与仿真技术的研究与应用开展得较早，并且发展迅速。自20世纪50年代开始，在运动体自动控制领域首先采用仿真技术，面向方程建模和采用模拟计算机的数学仿真获得较普遍的应用，同时采用由自行研制的三轴模拟转台等参与的半实物仿真试验已开始应用于飞机、导弹的工程型号研制中。60年代末，在开展连续系统仿真的同时，已开始对离散事件系统的仿真进行研究。70年代，我国训练仿真器获得迅速发展，我国自行设计的飞行仿真器、火电机组培训仿真系统、机车培训仿真器、坦克仿真器、汽车仿真器等相继研制成功，并形成一定市场，在操作人员培训中起了很大作用。80年代，我国建设了一批水平高、规模大的半实物仿真系统，如鱼雷半实物仿真系统等，这些半实物仿真系统在武器型号研制中发挥了重大作用。90年代，我国开始对分布交互式仿真、虚拟现实等先进仿真技术及其应用进行研究，开展了较大规模的复杂系统仿真，由单个武器平台的性能仿真发展为多武器平台在作战环境下的对抗仿真[12]。

作为建模的主要工具之一的Rhapsody软件，是美国TeleLogix公司(现已被IBM公司收购)开发的一种遵循UML标准的面向对象设计工具，主要针对嵌入式系统的设计。它为嵌入式系统的开发提供了一个“四化”的支撑平台，即可视化、工程化、自动化和团队化。目前，Rhapsody软件在美国和日本应用较为广泛，其中在日本的很多大学都开设有关于Rhapsody软件的课程，并且有日文版的Rhapsody软件。在国内，Rhapsody软件也逐步走入科研院所、企业等的视线，目前，国内的601研究所，611研究所，华为，华中科技大学等有关于Rhapsody软件的研究工作。

1.3 课题来源及研究意义

1.3.1 课题来源

本课题来源于中国航空工业第 601 研究所的“综合系统顶层设计与数字仿真软件包”项目，该项目是以基于 UML 语言的系统级建模与设计工具 Rhapsody 作为开发平台，利用其提供的图形化建模方式，对某型无人机飞行控制系统的行为、结构、数据交联关系及关键控制算法进行

建模和仿真，并作为系统虚拟样机软件集成的核心，与 MATLAB 等相关软件配合工作，生成系统分布仿真模型和仿真代码，实现模型驱动开发（MDD）能力。

1.3.2 课题研究意义

本文所要完成的飞行控制系统数字化设计，就是通过运用建模与仿真技术设计飞行控制系统的虚拟样机，即建立飞行控制系统的数字化模型来实现的。

建模与仿真技术得以发展的主要原因，是它所带来的巨大社会经济效益。在军事领域中，仿真技术已成为武器系统研制与试验中不可缺少的工具。国外统计数据表明，在研制导弹的过程中，由于使用了仿真技术，导弹飞行试验的数量可减少 30%－60%，研制费用可节省 10%－40%，研制周期可缩短 30%－40%，从而使型号研制得到很高的效费比。

不管系统多么复杂，只要能正确地建立起系统的模型，就可利用仿真技术对系统进行充分地研究。仿真模型一旦建立，可以重复使用，而且改变灵活，便于更新。经过仿真逐步修正，从而深化对其内在规律和外部联系及相互作用的了解，以采用相应地控制和决策，使系统处于科学化的控制与管理之下。

在无人机的研发过程中，采用建模与仿真技术建立系统的数字化模型，即设计系统的虚拟样机，从而实现无人机的数字化设计，这样不仅可以降低研发成本、提高总体研发质量，还能缩短研发周期。因此，在无人机的研发过程中采用建模与仿真技术是大势所趋，这样将有利于数字化设计的实现。飞行控制系统是无人机系统的核心子系统，同样采用建模与仿真技术进行开发，实现飞行控制系统的数字化设计。建模与仿真技术是面向虚拟样机的数字化设计方法中设计虚拟样机的核心技术。

本文主要是在虚拟样机平台上对某型无人机飞行控制系统建模，同时进行数字化仿真和半物理化仿真，即设计系统的虚拟样机。这种在虚拟样机平台上对某型无人机飞行控制系统建模与传统的建模不同，它是一种数字化建模。这种数字化建模与设计在本质上是相同的，即建模的过程就是设计的过程，设计的过程就是建模的过程。这种建模是一种面向虚拟样机的数字化设计方法，它以 UML 语言为主要建模语言，以 Rhapsody 软件为其协同设计平台的核心模块，其目的是建立系统的虚拟样机，依赖制造的虚拟样机进行系统的设计。使用这种数字化设计方法能够明显弥补传统设计方法的不足，具体如下：

（1）能够进行模型的测试。Rhapsody 软件提供了强大的模型测试功能，可以执行和调试系统模型[13]，在项目开发的早期就能够对系统进行可视化仿真和测试系统行为，以便在开发生命周期中尽早排除错误，提供一个清晰、一致和完备的需求，这对项目以后阶段的开发提供了正确的指导，促使项目的顺利进行，从而提高系统开发的效率和质量，减少开发成本。

（2）是一种可视化的建模方法。UML 是一种可视化建模语言，可用图形表示模型，表达

能力强[14]。运用 UML 语言的图形化方式提出需求，将会使需求分析更加生动形象，更加易于理解和交流。这样就能避免以文档的形式提出需求所产生的二义性。

（3）可以边设计边调试。所谓设计级调试能力就是允许在设计的同时对设计进行调试和验证，这样设计者可以在更短的时间内验证设计方案是否正确。通过使用可运行的设计模型，Rhapsody 软件使用户在调试和验证阶段仍然可以将工作重点放在设计上，消除了冗长乏味的代码级调试时间。

（4）模型与代码双向自动关联。代码的修改会导致模型的改变，模型的改变也会导致代码的更改，这种双向自动关联保证了模型和代码间的一致性，从而使得系统的扩充、增删和维护工作顺利进行[15]。

（5）可维护性强。使用 UML 语言建模，可将系统的维护工作分散在系统开发生存期的各个阶段，大大减少了系统开发后期的维护工作量[14]。

1.4 本文的内容结构安排

第一章：绪论

主要介绍课题的研究背景和国内外研究现状，同时介绍了课题的来源和研究意义，并对所做的研究工作与论文的结构做了一个概括性的阐述。

第二章：基于 Rhapsody 的虚拟样机平台的搭建

主要任务是搭建以 Rhapsody 为核心的虚拟样机平台，系统的虚拟样机设计(即数字化模型的建立)就是在虚拟样机平台上完成的。在搭建虚拟样机平台之前，在总体上提出某型无人机飞行控制系统的面向虚拟样机的数字化设计方法。虚拟样机平台搭建好后，需要在平台上完成具体系统的虚拟样机设计，因此。紧接着提出了某型无人机飞行控制系统功能需求，再设计出某型无人机飞行控制系统的组成结构，然后提出飞行控制系统所包含的功能模块，最后重点给出各组成部分的具体功能与性能要求。

第三章：设计模式的研究

首先，本章对设计模式做了一个基本介绍；其次，研究将设计模式用于 UML 建模的优点以及如何将设计模式用于 UML 建模；最后，在研究各种设计模式的基础上，选用适用于无人机飞行控制系统建模的设计模式，并研究了使用这些模式后的优点。

第四章：某型无人机飞行控制系统的虚拟样机设计

本章通过把模型驱动方法和柔性开发模式相结合，得到用于系统设计的模型驱动式柔性开发方法。运用该方法和设计模式，在搭建的虚拟样机平台的 Rhapsody 软件模块上完成了无人机飞行控制系统的虚拟样机(即数字化模型)设计，其中飞控系统的数字化模型包括需求模型、静态模型和动态模型等。在设计模型时，可以边设计边仿真验证，做到尽早、及时的排除错误。

完成系统的虚拟样机设计后，运用两种方式对虚拟样机进行阶段性测试。首先是利用 GMR 方式进行语法和语义的检测；其次是基于人机交互界面的交互式仿真验证。模型通过测试后，利用 Rhapsody 自动生成代码的功能生成产品级代码，为后续的半物理仿真奠定基础。

第五章：无人机动力学模型的导入

本章先是在虚拟样机平台的 Simulink 软件模块上建立无人机的动力学线性模型，再通过 Rhapsody 软件与 Simulink 软件的接口，采用“黑盒子”模式把建立的动力学线性模型导入 Rhapsody 工程(即在 Rhapsody 软件上建立的飞行控制系统数字化模型)，然后生成带有动力学模型的飞行控制系统的代码。生成的系统代码用于下载到系统的硬件平台，进行系统的半物理仿真验证。

第六章：总结与展望

总结本文已经完成的工作和本课题有待进一步完成的工作。

第二章 基于 Rhapsody 的虚拟样机平台的搭建

本章的主要任务是搭建以 Rhapsody 为核心的虚拟样机平台，虚拟样机平台用于设计系统的虚拟样机，即建立系统的数字化模型。在搭建虚拟样机平台之前，提出了某型无人机飞行控制系统的具体设计方案。虚拟样机平台搭建好后，需要在平台上完成具体系统的虚拟样机设计，由此提出了某型无人机飞行控制系统的功能需求，根据提出的功能需求，设计出了飞行控制系统的基本结构，并明确了结构的各部分功能。最后，本章简单介绍了飞行控制系统的基本工作原理及其飞行控制策略。

2.1 飞行控制系统设计方法

2.1.1 飞行控制系统传统设计方法

飞行控制系统的传统设计方法除了具备以瀑布模型为开发方法的特征外，还具备依赖制造物理样机进行开发的特点。在本章中，给出依赖制造物理样机进行开发的大致流程图。

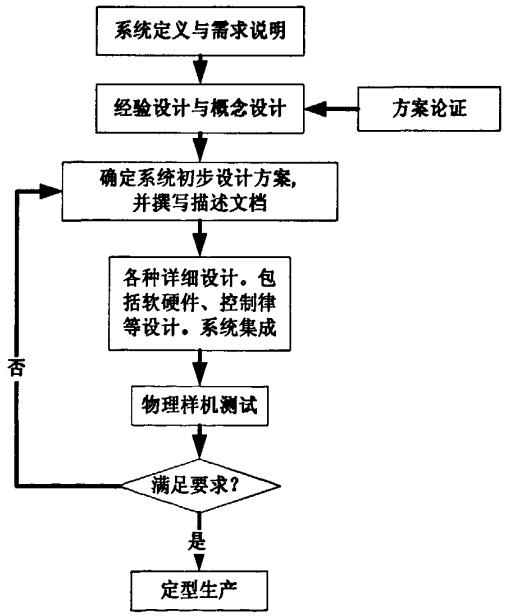


图 2.1 飞行控制系统传统设计方法流程图

从传统的设计过程来看，一个新系统的设计首先要经过需求分析，再进行系统初步方案的设计，待设计方案通过论证之后，才可确定各设计细节，并进行详细设计。系统详细设计完成之后，为了验证设计的正确性以及功能实现与否，往往需要制造物理样机进行性能测试。如果测试结果满意则投入生产实现，反之，则将对设计详细内容进行修改，并再次制造物理样机进

行测试，直至测试结果满意为止。在大多数情况下，样机的试制并不是一两次就能达到设计要求的，往往需要经过多次试制。实践表明，传统的设计方法存在很多弊端。除了前面所介绍的不足外，传统的设计方法还存在以下弊端[16]：

（1）物理样机的生产制造需要大量的时间和费用，即设计成本高、周期长；

（2）在某些情况下，物理样机的试验是破坏性的，甚至非常危险，比如飞行员的安全性试验和汽车的碰撞试验等；

（3）传统的设计过程大都采用传统的计算方法，计算速度慢、精度低，难以进行多种方案的分析对比；

（4）一些企业为了避免物理样机试制的繁琐以及费用方面的问题而省略样机测试这个环节，或者偷工减料、敷衍了事，这势必会降低产品的性能，甚至满足不了设计的要求。

2.1.2 面向虚拟样机的数字化设计方法

针对飞行控制系统传统设计方法存在的诸多弊端，本文提出了飞行控制系统的面向虚拟样机的数字化设计方法。图 2.2 为面向虚拟样机的数字化设计方法的流程图。

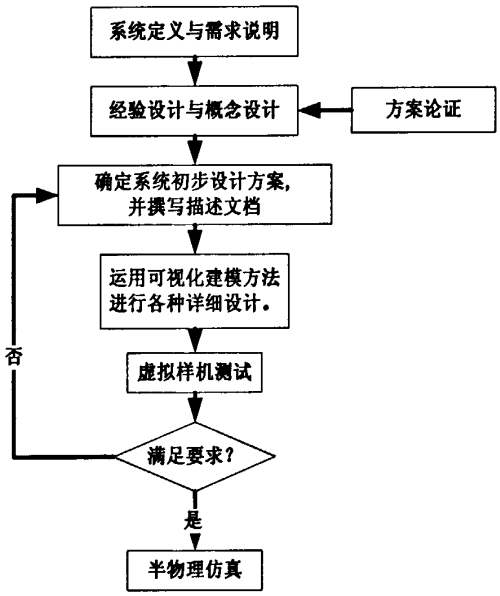


图 2.2 面向虚拟样机的数字化设计方法的流程图

该方法以虚拟样机代替物理样机完成系统整个设计流程中的样机测试。虚拟样机技术是一种崭新的产品开发方法，是一种基于产品的计算机仿真模型的数字化设计方法。这些数字化模型即虚拟样机将不同工程领域的开发模型结合在一起，从外观、功能和行为上模拟真实产品，并支持并行工程方法学。虚拟样机技术涉及到多体系统运动学、动力学建模理论及其技术实现，是一种基于先进的建模技术、多领域仿真技术、信息管理技术、交互式用户界面技术和虚拟现实技术的综合应用技术[17]。面向虚拟样机的数字化设计具有以下优点：

（1）是一种新的研发模式。传统的研发方法是一个串行过程，而虚拟样机技术真正地实现了系统角度的产品优化。它基于并行工程，使产品在概念设计阶段就可以迅速地分析、比较多种设计方案，确定影响性能的敏感参数，并通过可视化技术设计产品、预测产品在真实工况下的特征以及所具有的响应，直至获得最优的工作性能。

（2）设计过程中减少或避免实物模型的制造，即依赖虚拟样机进行系统的数字化设计。传统设计在产品定型生产前需经过“物理样机生产——物理样机测试——系统修改设计”的过程，且需反复多次，这不仅耗费物力、财力，还使得产品上市周期延长。数字化设计则在制造实物模型之前，先通过计算机技术建立产品的数字化模型，可以完成无数次物理样机无法进行的虚拟试验，排除某些设计的不合理性，从而无需制造及试验物理样机就可获得最优方案，这样不但减少了物理样机的数量、缩短了研发周期，而且提高了产品质量。

（3）实现并行设计。一项设计工作可由多个设计队伍在不同的地域分头并行设计、共同装配，形成一个可完成强度、制造性、成本和功能测试的完整的数字化模型。

2.2 虚拟样机平台的设计

面向虚拟样机的数字化设计方法的核心思想就是以虚拟样机代替物理样机完成系统设计流程中的样机测试，实现飞行控制系统的数字化设计。因此，设计出飞行控制系统的虚拟样机就成为设计的核心，而能否搭建起一个合适的虚拟样机平台成为能否成功设计出虚拟样机的关键。本文的一个主要任务就是搭建以 Rhapsody 软件模块为核心的虚拟样机平台。

2.2.1 虚拟样机平台的作用

虚拟样机平台主要是用于设计系统的虚拟样机，以取代样机测试中的物理样机。本文的虚拟样机平台主要是用于设计某型无人机飞行控制系统的虚拟样机，即建立某型无人机飞行控制系统的数字化模型。设计飞行控制系统的虚拟样机，实质上就是对飞行控制系统进行数字化建模。本文已经强调：本课题组所提出的数字化建模与设计在本质上是相同的，即建模的过程就是设计的过程，设计的过程就是建模的过程；数字化建模本身是一种数字化设计方法。

第四章和第五章的主要工作就是在本章搭建的虚拟样机平台上分别设计飞行控制系统的虚拟样机和无人机的动力学模型，即分别对飞行控制系统的虚拟样机和导入动力学模型后的飞控系统模型进行建模与仿真验证。

2.2.2 虚拟样机平台的基本功能要求[18]

虚拟样机平台的作用是建立起系统的虚拟样机（即数字化模型）并完成仿真验证实验，其主要功能包括样机建模与设计、样机的仿真和调试。利用这一平台，可以从无到有逐步建立不同粒度的系统样机模型。

建模与仿真是虚拟样机平台最基本、也是体现样机水平的组成部分。目前用于系统和分系统的 CAD 软件已经相当丰富，给样机建模提供了技术支持。但其主要的困难是如何把不同设计平台的结果进行无缝连接，并在统一的设计平台上进行仿真运行和试验。

为了满足建立和运行虚拟样机的需要，虚拟样机平台应具备的基本功能如下：

（1）适应不同类型的飞行控制系统样机的建模需要，具有可视化建模手段，自动生成运行策略和执行仿真。

（2）采用多层模型来表示虚拟样机的复杂模型，并具有无限制的分层功能。

（3）支持高级语言建模。同时为了建模的高效率，软件框架能够直接调用综合工程软件包的高级语言组件。

（4）完善的模型排错手段。在仿真过程中可进行中断调试和分析，并提供丰富的仿真算法。

（5）完善的模型库交互管理功能。模型库的补充方便、快捷，使用户可以很容易地添加新的模型，以促进系统的完善和扩充。

（6）精确逼真的图形显示功能，能实时表示系统部件及内部的物理运动或虚拟模型。

2.2.3 虚拟样机平台的搭建

根据虚拟样机平台的要求，结合飞行控制系统的性能，本课题组采用以下虚拟样机平台的构架，如图 2.3[15]。搭建虚拟样机平台的构架运用了 MVC 模式。如何运用 MVC 模式搭建虚拟样机平台的构架，详见本论文的第三章。

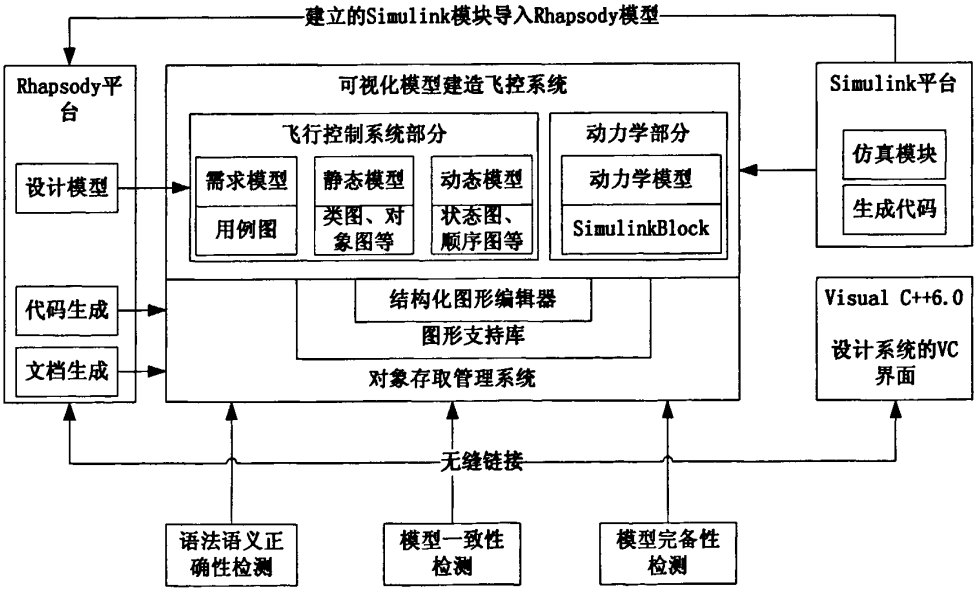


图 2.3 虚拟样机平台

虚拟样机平台支持 UML 高级建模语言，主要由 Rhapsody、Simulink、VC++等软件构成。

其中 Rhapsody 软件是整个平台构架的核心，它与其它软件无缝连接，从而构成了一个有机的总体。在该平台上进行虚拟样机设计所具有的优点在第一章绪论中已有简单的介绍，在本章中将做更详细的介绍。

面向虚拟样机的数字化设计方法的优点主要是通过虚拟样机平台的特点来体现，本文所搭建的虚拟样机平台具有以下特点：

（1）具有可视化建模手段。采用可视化建模语言 UML，使得系统建模过程变得可视和可监控。

（2）具有连续性。UML 语言具有连续性，即前一个阶段使用的模型与下一个阶段使用的模型紧密衔接，甚至前一个阶段使用的模型可以直接被下一个阶段使用，使用的技术经过生存期的每一个阶段后不发生改变。

（3）选用 Rhapsody 作为核心模块。Rhapsody 是基于内置 OXF 实时框架的软件。OXF 框架面向对象和跨 RTOS 平台，它很好地将 UML 的概念移植实时嵌入式系统中来，为嵌入式系统的开发提供了清晰的结构以及可复用的软件模块，能够切实提高实时嵌入式系统的开发效率和可维护性[19-20]。

（4）实现 Rhapsody 和 Simulink 的协同工作。利用 Rhapsody 与各领域工具之间的现有接口及中间件技术，实现了 Rhapsody 和 Simulink 的无缝连接，达到了流程与数据模型共享。

（5）以模型驱动的思想为指导。面向对象的方法缺乏贯穿整个系统设计流程的线索，模型驱动弥补了这种不足。模型驱动使得系统开发的整个开发过程是由对系统的建模行为驱动的，其生命周期各个阶段产生的工件是能够被计算机理解的形式化模型。模型在整个生命周期中处于核心地位，贯穿系统整个开发过程[21]。Rhapsody 和 Simulink 的协同工作扩展了模型驱动。

（6）具有共同的标准环境。能使不同的模型在共同的标准环境里进行沟通，从而打破了开发各阶段的隔膜，实现了在生命周期各阶段间的合作与信息共享，使各阶段紧密集成。Rhapsody 是一种基于 UML2.0 和 SysML（Systems Modeling Language）统一标准环境的嵌入式系统工具[22]。它使用的 UML 语言能尽可能保持语言简单，其目标是使所有的方法使用一个共享的模型语言，使其达到普通化。

（7）具有完善的模型排错手段。Rhapsody 具有边设计边调试验证的功能，能尽早、及时的排除错误，降低系统风险。

（8）能对系统的静态结构和动态行为生成代码。由于 Rhapsody 能够自动生成大部分代码[22]，省去了一部分编写代码的繁琐工作，从而提高了代码质量和开发效率。

（9）提高了被设计系统的快速反应能力。模型与代码双向自动关联[22]：当用户对生成的代码进行修改后，代码的修改转换到模型上；当用户对模型进行修改，生成的代码也会相应的改变。从而保证了模型和代码间的一致性，使系统的扩充、增删和维护工作顺利进行[15]。

（10）具有强大的模型仿真验证功能。在虚拟样机平台上进行虚拟样机模型的设计时，不仅可以在设计过程中做到边设计边仿真验证，还可以在完成系统模型的整体设计以后，对模型进行综合仿真验证。Rhapsody提供了三种不同的验证方式：1)利用GMR(G表示代码生成、M表示编译、R表示运行)方式对模型进行形式化语法和语义进行检测；2)基于VC界面的交互式仿真，验证系统行为和功能，从而达到对设计模型的一致性和完备性的检测；3)同dSp-ace等商用平台联合进行快速原型验证。在本论文中采用前两种仿真验证方式对建立的飞行控制系统模型进行了仿真验证。

（11）可以自动生成多种形式的文档。在 Rhapsody 软件上，可以生成 word、ppt、html 等多种形式的文档。

（12）便于扩展。虚拟样机平台的核心模块 Rhapsody 软件提供了与众多的第三方软件连接的接口(如与 Simulink、DOORS、Rational Rose 等软件的接口)，以便日后扩展现有的虚拟样机平台，满足更为复杂的设计要求。

（13）虚拟样机平台结合了虚拟样机和协同设计的理念，并集中发挥了它们各自的优点，把协同设计的高效率与虚拟样机的可视化仿真结合了起来。不但提高了设计工作的速度，也给产品的管理带来了令人耳目一新的革新和直观性[23-24]。

2.3 飞行控制系统的功能需求

某型无人机飞行控制与任务设备管理系统（以下简称飞行控制系统）是一个以高性能嵌入式计算机系统为核心的数字化飞行控制系统，用于实现无人机从起飞到回收着陆的全飞行过程的飞行控制与任务设备管理功能。飞行控制系统接收从地面测控站上行信道经机载无线电测控终端送来的控制命令及数据，机载传感器测量各个有关的无人机运动状态参数，经机载飞行控制计算机处理，输出控制指令到各个执行机构及有关设备，实现对无人机各种飞行模态的控制和任务设备的管理。同时，飞行控制系统亦把无人机的飞行状态数据及发动机、机载电源系统、任务设备工作状态参数实时传给无线电终端，经无线电下行信道发送回地面站测控站，为地面控制人员提供无人机及其任务设备的有关状态信息，以实时控制引导无人机完成飞行计划任务。某型无人机飞行控制系统的主要功能包括：

●接收并处理无线电测控终端的飞行控制命令；

●接收并处理无线电测控终端的任务设备控制命令；

●接收并记录无线电测控终端传送的预置航迹与装定数据；

●返回给无线电测控终端各种飞行状态、发动机工作状态、电源系统工作状态、任务设备工作状态等遥测信息；

●各传感器信息的采集和处理；

●磁航向解算；

●横侧向、纵向控制律的解算；

●导航参数的计算；

●时序控制与逻辑处理；

●任务设备的控制。

2.4 飞行控制系统的结构组成

飞行控制系统主要由机载传感器、执行机构和飞控计算机三部分构成，系统结构如图 2.4 所示：

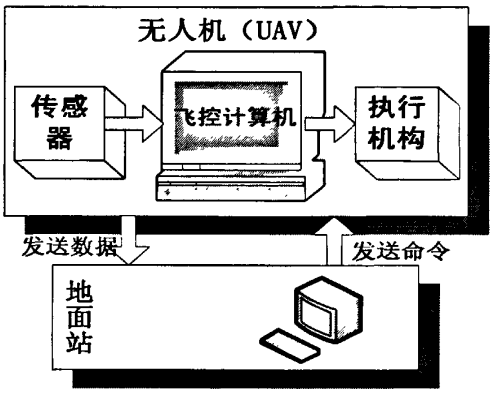


图 2.4 飞行控制系统结构图

机载传感器实时测得无人机的飞行状态，传入飞控计算机，飞控计算机根据采集的传感器信息，存储有关状态，同时接收无线电测控终端发送来的遥控指令，经判决、分析和解算后，输出指令给执行机构(即舵面)，控制无人机的舵面、发动机风门和前轮的偏转，从而控制无人机的飞行或地面滑跑。

其控制回路如图 2.5 所示：

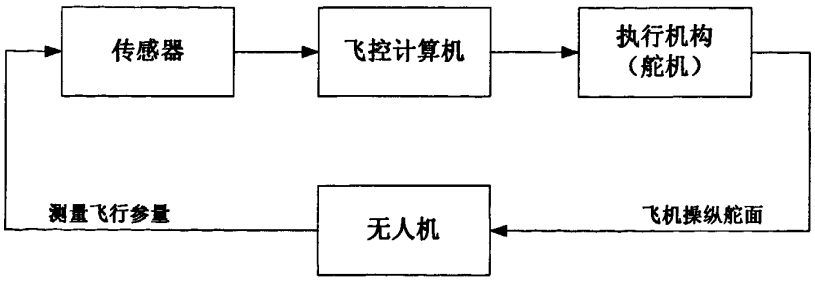


图 2.5 飞行控制系统控制回路图

可以看出这是一个反馈系统，即闭环系统，其一端是测量飞机参数的传感器，另一端是控制飞机操纵界面的伺服机构。当飞机偏离原始状态时，敏感元件感受方向和大小，输出相应的信号给飞控计算机；信号经放大、计算处理后输出给执行机构，操纵执行机构偏转，结果使得

飞机回到原始状态。其中，信号的处理、变换、调整、控制律的计算、工作模态的转换及状态参数的逻辑判断等等都在飞行控制计算机内由飞行控制软件来完成。

根据飞行控制系统的功能要求，给出了飞行控制系统的更为详细的结构图。如图 2.6 所示：

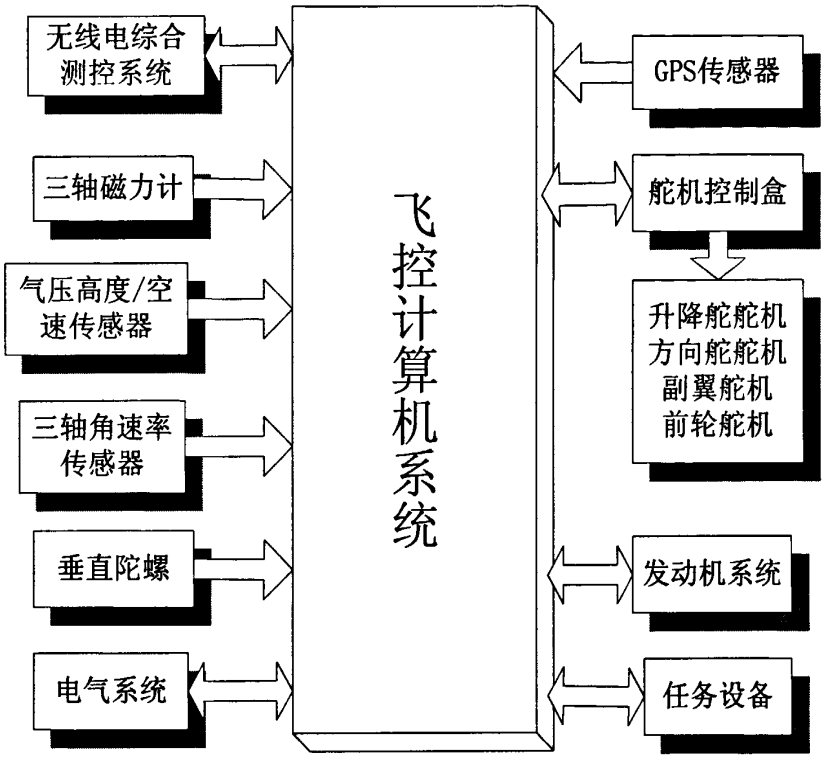


图 2.6 飞行控制系统详细结构图

根据图2.6，对设计的某型无人机飞行控制系统的主要组成部分做功能说明：

1、机载传感器

无人机的传感器用于测量无人机的飞行控制系统的飞行姿态、飞行速度、航向、高度、位置等状态信息，包括有：

●垂直陀螺：测量飞机的俯仰/横滚姿态角；

●三轴角速率传感器：测量飞机沿三个机体轴的旋转角速率；

●高度空速传感器：测量飞机的动静压，并依此解算出飞机的气压高度和空速；

●三轴磁力计：测量地磁场沿机体轴的三维向量分布；

●GPS 导航接收机：接收 GPS 卫星导航信息，以获取飞机的导航参数(经度，纬度)；

2、执行机构

执行机构即舵机系统，主要包括 6 个小型直流电动机舵机。6 个直流电动舵机分别为：

●升降舵机：操纵飞机的升降舵上下偏转；

●左/右副翼舵机：操纵飞机的左/右副翼做上下偏转；

●左/右方向舵机：操纵飞机的左/右副翼做左右偏转；

●前轮舵机：操纵飞机的前轮做左右偏转；

●风门舵机：操纵发动机风门的开度大小。对风门的控制如果由 PFM(脉冲频率调制)改为由发动机 ECU 直接控制风门开度，则风门舵机可以不再使用。

3、飞控计算机[25]

飞控计算机的发展是从模拟式开始的，随着飞行控制系统功能的日益增多，飞行包线的不断扩大，尤其是综合控制技术的发展，目前的飞控计算机已发展成为以数字机为基础的、可实现多种组合的余度计算机系统。

飞控计算机是整个飞行控制系统的核心，担负着系统数据采集，余度管理，控制律计算等重要任务。它性能的优劣直接关系到飞行控制系统乃至整个无人机的性能。飞控计算机实时采集各个传感器测量的无人机飞行状态数据、接收无线电测控终端传输的由地面测控站上行信道送来的控制命令及数据，经计算处理，输出控制指令给执行机构，通过操纵无人机舵面的偏转和控制发动机风门的开度来控制无人机的空中飞行、起飞和着陆。同时，飞控计算机亦把无人机的飞行状态数据及发动机系统、机载电源系统、任务设备的工作状态参数实时传递给无线电测控终端，经无线电下行信道发送给地面站。

本文所设计的飞控计算机系统以嵌入式计算机系统为核心，主要包含以下功能组合模块：

●CPU 模块：飞控计算机系统的核心，用于所有信息数据的存储、运算和处理；

●A/D 转换模块：用于模拟信号的模/数转换；

●串行通信模块：提供连接传感器、机载设备等的串行通道；

●定时计数模块：提供定时计数功能；

●离散 I/O 模块：输入离散状态信号，输出开关控制信号；

●模拟信号前置处理模块：对模拟输入信号进行滤波和电平转换，将不同的模拟信号电压转换为 A/D 模块所要求的电压范围；

●电源转换模块：实现电源转换功能，把机载电压转换成计算机，传感器和舵机等要求的不同数值的电压。

2.5 飞行控制系统的基本工作原理及控制策略

飞行控制系统是一个多通道的控制系统，即多输入/多输出控制系统。根据某型无人机飞行控制系统设计要求及所要完成的功能，可将无人机飞行控制分为姿态保持与控制模态、航向保持与控制模态、高度保持与控制模态、速度控制模态、自主导航模态、自动起飞模态及自主着陆模态，这些模态控制都是通过控制回路来实现的。

飞行控制系统的核心控制回路是以姿态角(俯仰角，滚转角)信号反馈为基础构成的飞行姿

态稳定与控制回路，称为内回路。内回路是飞行高度、航向控制等外回路的基础。无人机转弯、爬升、下滑飞行模态是由内回路给定相应的俯仰/滚转参考角来实现的[26]。

总的来说，无人机的飞行控制可以分为相对独立的纵向控制和横侧向控制。因此，在对某型无人机飞行控制系统部分和某型无人机动力学部分的建模过程中，都有纵向和横侧向模型，尤其是在设计无人机动力学部分的模型时，就是将其解耦为纵向和横侧向两部分，分别进行模型的设计。

在设计无人机的动力学模型时，将会涉及到飞行控制系统的控制律设计。

2.6 本章小结

为了设计某型无人机飞行控制系统的虚拟样机，本章给出了面向虚拟样机设计的数字化设计方法，搭建了适用于该方法的虚拟样机平台，提出了系统的功能需求。在功能需求的基础上，我们设计出了飞行控制系统的基本硬件结构，并明确了硬件结构的各部分功能。在本章的结尾，简单介绍了飞行控制系统的基本工作原理及其飞行控制策略。本章是后续工作的基础，在此基础上，设计了飞行控制系统的虚拟样机，并进行了数字化仿真。

第三章 设计模式的研究

设计模式是面向对象软件工程中的一个重要概念，是从软件模式分支中衍生出来的一个解决具体问题的重要方案之一。设计模式使人们可以更加简单方便地复用成功的设计案例和体系结构，帮助设计者更快更好的完成系统设计。本章重点研究了 GOF 的 23 种设计模式和 MVC 设计模式。通过对设计模式的研究，选取合适的设计模式用于飞行控制系统建模。

3.1 设计模式概述

模式概念的提出最初源于建筑学领域，由建筑工程师 Christopher Alexander 于 1977 年在他的描述建筑体系结构模式的著作中首次提出。由于 Alexander 在计算机科学领域的影响较大，因此，他的思想被许多计算机科学家所接受，并逐渐引入到面向对象的软件开发过程中。1994 年，由 Eric Gamma、Richard Helm、Ralph Johnson 和 John Vlissides（世人称“Gang of Four”—GOF 四人帮）合著的《设计模式：可复用面向对象软件的基础》（该书中文版已由机械工业出版社出版）详细描述了设计模式的概念，介绍了 23 种基本设计模式及其实现代码。1996 年由 Frank Buschmann 等五人合著的《面向模式的软件体系结构》（Patterns Oriented Software Architecture, 简称“POSA”）是模式发展史上的一个重要里程碑。

3.1.1 设计模式的基本概念[14]

设计模式的核心思想是总结和积累前人成功的设计经验，通过对这些经验的学习，使得人们在面对新的设计问题时不用重复所有环节，而是尽量套用已有的模式实施，以提高系统设计效率。广义上讲，设计模式是对被用来在特定场景下解决一般设计问题的类和相互通信的对象的描述。从狭义的角度理解，设计模式就是对特定问题的描述或解决方案。实质上设计模式就是解决特定问题的经验，是一种被复用的方案。

对设计模式的理解，更多人认为是对特定问题的解决方案，即在设计过程中出现了问题后，才会考虑到是否有现存的设计模式可以解决这种问题。但通过研究，本文认为设计模式应该还具备指导设计的作用，即套用已有的设计模式指导系统某部分设计，而不仅仅是运用设计模式解决已经出现的问题。

本章既包括用设计模式指导飞行控制系统的数字化建模（即数字化设计），又包括使用设计模式解决特定的问题。

3.1.2 设计模式的描述[27]

一般而言，一个设计模式有四个基本要素：

（1）模式名称（pattern name）：一个助记名，它用一两个词描述模式的问题、解决方案和效果。

（2）问题（problem）：描述了应该在何时使用模式。

（3）解决方案（solution）：描述了设计的组成部分，它们之间的相互关系及各自的职责和协作方式。

（4）效果（consequences）：描述了模式应用的效果及使用模式应权衡的问题。

在实际运用中，可以将上述四个基本要素细化，分门别类的详尽表述。通常情况下，一个设计模式可以按如下内容表述：

（1）模式名和分类：模式名称简洁地描述了模式的本质，模式的分类将在 3.1.3 节介绍。

（2）意图：明确其作用、基本原理以及要解决的特定问题。

（3）别名：模式的其它名称，易于为人理解。

（4）动机：用于说明一个设计问题和如何用模式中的类、对象来解决该问题的特定情景。

（5）适用性：什么情况下可以使用，有什么地方需要改进，以及如何识别。

（6）结构：用图形进行直观描述。

（7）参与者：指设计模式中的类和对象以及它们的各自职责。

（8）协作：设计模式的参与者怎样协作以实现它们的职责。

（9）效果：所需做的权衡取舍，以及实现中的优点与不足。

（10）实现：实现模式时需要知道一些提示、技术要点及应避免的缺陷，以及是否存在某些特定于实现语言的问题。

（11）代码实例：一般以动机中提出的问题为例、给出设计模式实现的代码例子。

（12）已知应用：实际系统中发现模式的例子。

（13）相关模式：与这个设计模式紧密相关的其它设计模式。

3.1.3 设计模式的分类[27]

根据以下两条准则对设计模式进行分类：

第一是目的准则，即设计模式是用来完成什么工作的。设计模式依据其目的可分为创建型（Creational）、结构型（Structural）和行为型（Behavioral）等三种。

第二是范围准则，指定设计模式主要是用于类还是用于对象。

按照上述两条准则对设计模式进行划分，如表 3.1 所示

表 3.1 设计模式的分类

目 的 准 则				
		创 建 型	结 构 型	行 为 型
范围准则	类	Factory Method（工厂方法）	Adapter(适配器)	Interpreter（翻译） Template Method（模板方法）
	对象	Abstract Factory(抽象工厂) Builder（建造） Prototype（原型） Singleton(单例)	Adapter(适配器) Bridge(桥接) Composite（组合） Decorator（装饰） Façade(外观) Flyweight（享元） Proxy(代理)	Chain of Responsibility(职责链) Command(命令) Iterator(迭代器) Mediator(中介) Memento（备忘录） Observer(观察者) State(状态) Strategy(策略) Visitor(访问者)

3.2 设计模式与框架的区别

设计模式和框架是记录或重用某些设计元素的两种方式，有时常会混淆。为了更好的运用设计模式，本文深入研究了设计模式和框架的区别，得到设计模式和框架的三点不同之处：

（1）设计模式与框架利用的规模不同。框架定义了一个完整应用或相关的一类应用构架；设计模式描述的是单个设计问题的解决方案，它可用于许多应用或框架中。如在 Rhapsody 软件上建模就是基于 OXF 框架，在此框架中运用设计模式解决特定问题，并指导建模。

（2）设计模式和框架有不同的内容。一个设计模式是纯粹的设计思路，所以设计模式可以指导建模。设计模式可用不同语言以不同方法改写和实现。如可用 UML 语言，Java 语言，C++语言等实现设计模式。本文主要是用 UML 语言建模，因此，使用 UML 语言实现设计模式。运用 UML 语言实现设计模式的描述后，还可以生成 C++代码，以 C++代码的形式体现设计模式。框架典型的是设计和代码的混合物，它可以由应用程序以各种方式扩展。

（3）设计模式比框架更关注可移植性。框架是已经实现了的，并没有必要组成一个完整的应用，因此常受限于一个单一的实现环境。设计模式是独立于语言的，可以在不同情况中得到广泛的应用[28]。

研究清楚了设计模式与框架的区别，有利于更好的应用设计模式。

3.3 设计模式的应用研究

为了更好的应用设计模式，对设计模式的优点与意义进行了研究，同时研究了应用设计模式应该遵循的原则。

3.3.1 设计模式的优点和研究意义

通过设计模式的实际应用，参照文献[14], 提炼出设计模式具有以下优点：

（1）巧妙。设计模式是一些优雅的解决方案，一般很难立刻设计出来。

（2）通用。设计模式通常不依赖某个特定的系统类型、程序设计语言或应用领域，它们是通用的。设计模式在实际系统和面向对象系统中得到广泛应用，它们并不仅仅停留在理论上。

（3）简单。设计模式通常都非常小，只涉及很少一些类。为了构建更多更复杂的解决方案，可以把不同的设计模式与应用代码结合或混合起来使用。

（4）可复用的解决方案。通过复用已经公认的设计，能够在解决问题时取得先发的优势，而且避免重蹈前人覆辙，从学习他人的经验中获益。由于是复用，在遇到问题时，无需每次都从底层做起，而是可以从标准解决方案——设计模式入手，并对它改编，使之适应特殊问题的需要，这样既能节省时间，又能提高开发的质量。

（5）面向对象。设计模式是用最基本的面向对象机制（如类、对象、通用化和多态等）构造的。

（6）确立通用术语。开发中的交流和协作都需要共同的词汇基础和对问题的共识。设计模式为项目分析和设计阶段的研究提供了共同的基准点[29]。

本文研究设计模式的意义就在于本文需要使用设计模式。那么为什么要使用设计模式呢？通过对设计模式优点的研究，本文已经给出了答案。

3.3.2 设计模式应遵循的原则[14][29]

为了更好的复用设计模式，提高复用设计模式之后的可维护性，前人在提出设计模式的同时，也规定了设计模式应该遵循的原则。设计模式应该遵循的原则如下：

（1）开闭原则。一个系统实体应当对扩展开放，而对修改关闭。系统开发过程中或完成后，其需求是不断变化的，功能可能增加或减少。如果修改系统，就可能给整个系统带来问题，包括编译、测试等方面。如果新功能能够作为单独的模块插入系统，则可以降低集成的成本，只需编译、测试该模块即可。

（2）从背景设计原则，即在设计各部分所呈现的细节之前先创建总体概念。在总体的大背景下设计各个部分。

（3）依赖倒置原则。要依赖于抽象，不要依赖于具体。即高层模块不应该依赖于底层模块，

细节应该依赖于抽象类。这一原则意味着使用对象和被使用对象之间只能在概念层次存在耦合，而非实现层次。即对象之间通过接口或抽象类联系，并不直接通讯。

（4）封装变化原则，即一个类只封装一个变化。如果一个类包含多个变化，随着功能的增加，类的继承层次就会越来越多，对象间的关系、逻辑随之愈发复杂。这增加了变化之间的耦合，降低了内聚性。系统的可靠性大大降低。

（5）理性怀疑原则，即不要过分依赖模式。模式都是有益的，但并非颠扑不灭的“真理”。模式应该作为思考问题的一种辅助手段，而不是解决问题的处方。

3.4 应用设计模式设计飞行控制系统[30][31]

本文的绪论已经阐明：本文所说的建模是一种数字化建模。这种数字化建模与设计在本质上是相同的，即建模的过程就是设计的过程，设计的过程就是建模的过程。应用设计模式设计飞行控制系统，实质就是运用设计模式指导飞行控制系统建模和解决建模中的特定的问题。通过对 GOF 的 23 种设计模式和 MVC 模式的深入研究，选择了几种合适的设计模式应用到飞行控制系统建模中，以下就是对这几种设计模式的重点研究。

3.4.1 简单工厂模式的应用研究

简单工厂模式是一种类的创建模式，又称为静态工厂方法模式。简单工厂模式包含以下三种角色：

（1）工厂类角色（Creator）：这是本模式的核心，含有一定的商业逻辑和判断逻辑。它往往由一个具体类实现。

（2）抽象产品角色（Product）：一般它是具体产品继承的父类或者实现的接口。抽象产品角色可由一个接口或者抽象类来实现。

（3）具体产品角色（ConcreteProduct）：工厂类所创建的对象就是此角色的实例。具体产品角色由一个具体类来实现。

简单工厂模式的类图结构如下：

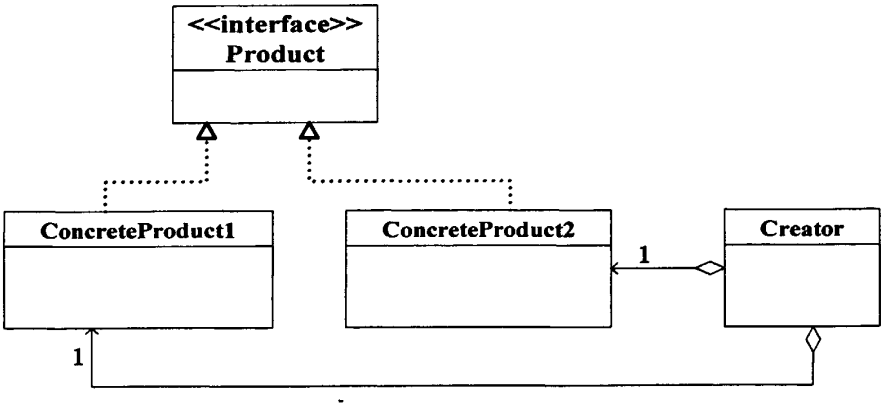


图 3.1 简单工厂模式的类图结构

依据第二章的分析得知：在设计飞行控制系统时，将会涉及到垂直陀螺，GPS 传感器，三轴角速率陀螺，高度/空速计，磁力计等五种传感器。这些传感器将会与飞控计算机进行数据的传输。在飞控系统的建模中，本文运用简单工厂模式表示这种关系。具体分析如下：

（1）工厂类角色（Creator）：飞控计算机（FCC），这是本模式的核心。

（2）抽象产品角色（Product）：传感器（Sensor）。

（3）具体产品角色（ConcreteProduct）：垂直陀螺（GyroSensor），GPS 传感器(GPS)，三轴角速率陀螺(RotationSensor)，高度/空速计(HVSensor)，磁力计(Magnetometer)。

运用简单工厂模式设计出的类图结构如下：

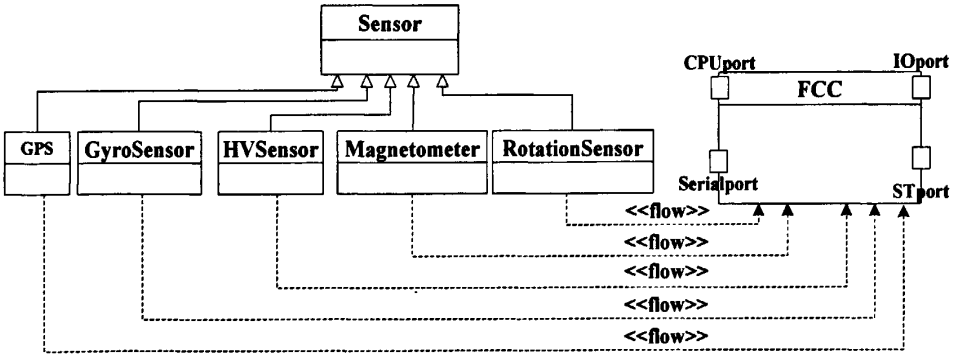


图 3.2 飞控计算机与传感器之间的类图结构

除了传感器与飞控计算机之间的关系可用简单工厂模式表示，舵机与飞控计算机之间的关系，舵机电位器与飞控计算机之间的关系都同样可以采用简单工厂模式。舵机与飞控计算机之间的各类角色如下：

（1）工厂类角色（Creator）：飞控计算机（FCC）。

（2）抽象产品角色（Product）：舵机（Gear）。

（3）具体产品角色（ConcreteProduct）：方向舵机(Rudder)，升降舵机(Elevator)，副翼(Aileron)，前轮舵机(Frontwheel)，油门舵机(Throttle)。

运用简单工厂模式设计出的类图结构如下：

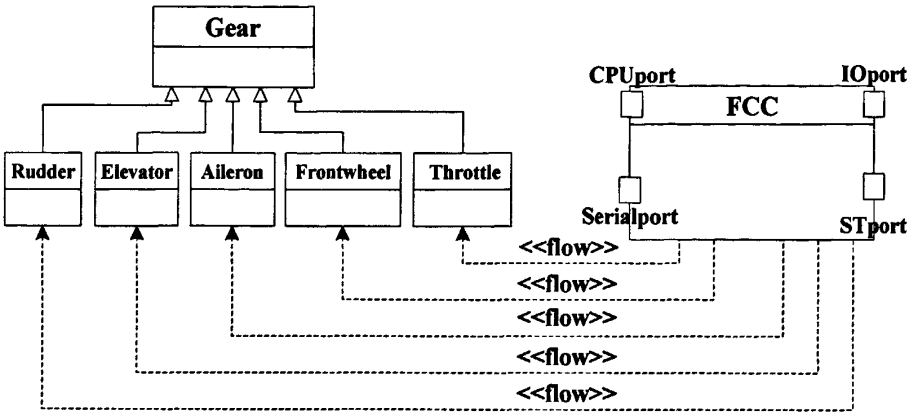


图 3.3 飞控计算机与舵机之间的类图结构

舵机电位器与飞控计算机之间的各类角色如下：

（1）工厂类角色（Creator）：飞控计算机（FCC）。

（2）抽象产品角色（Product）：电位计（Potentiometer）。

（3）具体产品角色（ConcreteProduct）：方向舵机电位计（RPotentiometer），升降舵机电位计（EPotentiometer），副翼电位计（APotentiometer），前轮舵机电位计（FPotentiometer），油门舵机电位计（TPotentiometer）。

运用简单工厂模式设计出的类图结构如下：

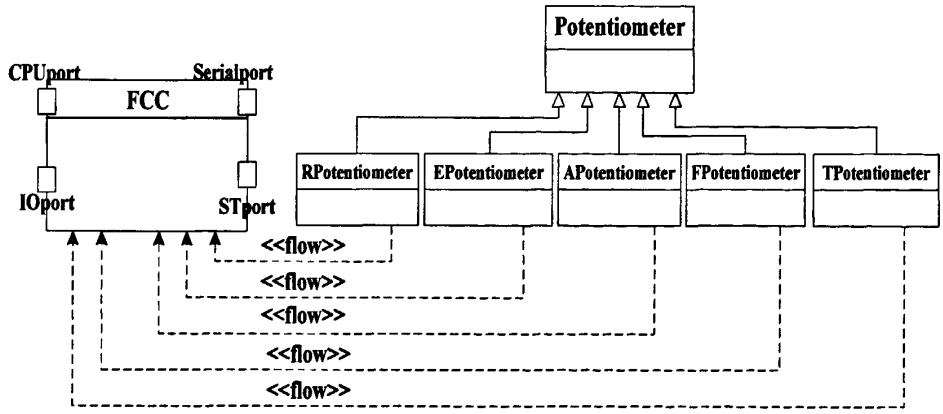


图 3.4 飞控计算机与电位器之间的类图结构

简单工厂模式是最简单的设计模式，因此能够很方便的被复用。

3.4.2 工厂方法模式的应用研究

工厂方法模式去掉了简单工厂模式中工厂方法的静态属性，使得它可以被子类继承。这样在简单工厂模式里集中在工厂方法上的压力可以由工厂方法模式里不同的工厂子类来分担。工厂方法模式组成如下：

（1）抽象工厂角色（Creator）： 这是工厂方法模式的核心。它与应用程序无关，是具体工厂角色必须实现的接口或者必须继承的父类。它由抽象类或者接口来实现。

（2）具体工厂角色(ConcreteCreator)：它含有和具体业务逻辑有关的代码。由应用程序调用以创建对应的具体产品的对象。

（3）抽象产品角色(Product)：它是具体产品继承的父类或者是实现的接口。它一般由抽象类或者接口来实现。

（4）具体产品角色(ConcreteProduct)：具体工厂角色所创建的对象就是此角色的实例。一般由具体的类来实现。

用类图来清晰的表示下的它们之间的关系：

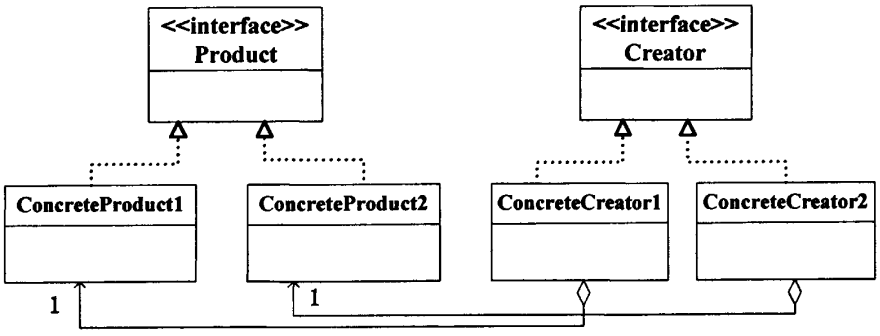


图 3.5 工厂方法模式的类图结构

运用工厂方法模式后的优点体现在：

（1）便于系统的扩展；

（2）降低了类之间的耦合度，增加了可维护性；

（3）各具体工厂角色各司其职，与简单工厂模式相比，减轻了工厂角色负担，提高了效率；

（4）复用设计模式，提高了系统设计效率。

反馈舵机的舵面位置是由其对应的电位器测量得到。在飞控系统的建模中，可运用工厂方法模式表示这种关系。具体分析如下：

（1）抽象工厂角色（Creator）： 舵机类（Gear）。这是所有具体舵机的抽象。

（2）具体工厂角色(ConcreteCreator)：飞控系统所包含的所有具体舵机（执行机构）。包括副翼（Ailerons），升降舵机（Eleavator），方向舵机（Rudder），前轮舵机(FrontWheel)，油门舵机(Throttle)。

（3）抽象产品角色(Product)：电位器（Potentiometer）。

（4）具体产品角色（ConcreteProduct）：各个舵机所对应的电位器。包括升降舵机电位器（EPotentiometer），副翼电位器（APotentiometer），方向舵机电位器（RPotentiometer），前轮舵机电位器(FPotentiometer)，油门舵机电位器(TPotentiometer)。

采用工厂方法模式后的类图结构如下：

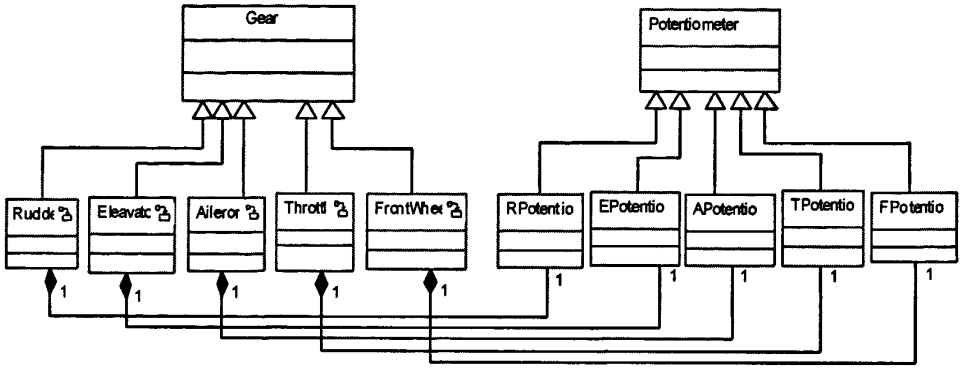


图 3.6 舵机电位器之间的类图结构

运用工厂方法模式建立舵机与电位器之间的模型，其优点具体如下：

（1）降低了类 Gear 和电位器之间的耦合度，当需要修改某个舵机或电位器时，将不会影响到其它的舵机或电位器；

（2）便于以后扩展。例如在飞控系统中增加一个鸭翼舵机，只需在舵机的基础上增加一些操作和属性，不会影响到其它舵机，同时能生成其对应的电位器类，也不会影响到其它舵机的电位器；

（3）每个舵机各自使用自己电位器，并行运行，不会出现当两个舵机同时使用电位器时，因抢占资源或等待而降低使用效率。

3.4.3 适配器模式的应用研究

适配器模式可以将某个类的接口转换成客户希望的另外一个类的接口。适配器模式使得原本由于接口不兼容而不能一起工作的那些类可以一起工作，它又名包装器。适配器的组成如下：

（1）目标（Target）角色：定义 Client 使用的接口。

（2）被适配（Adaptee）角色：这个角色有一个已存在并使用了的接口，而这个接口是需要我们适配的。

（3）适配器（Adapter）角色：这个是适配器模式的核心。它将被适配角色已有的接口转换为目标角色希望的接口。

适配器模式有类适配器和对象适配器两种。类适配器使用一个具体的 Adapter 类对 Adaptee 和 Target 接口进行适配，结果是一个适配器 Adapter 只能与一个 Adaptee 同时工作。对象适配

器则允许一个适配器 Adapter 能与多个 Adaptee 同时工作。因此，在本文的建模中我们采用的是对象适配器模式。对象适配器模式的类图结构如下：

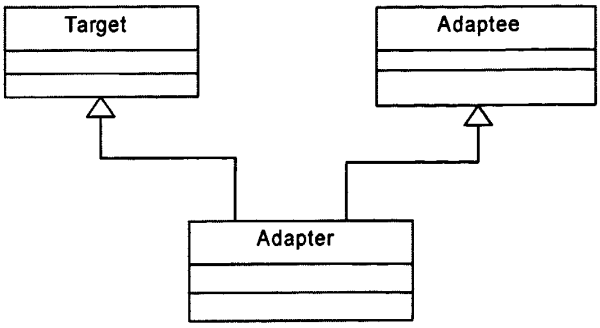


图 3.7 适配器模式的类图结构

运用适配器模式后的优点体现在：

（1）运用适配器模式，可以精简系统结构，无需为所有的类都分别建立一个对应接口；

（2）可以使由于接口不兼容而不能一起工作的类能够一起工作；

（3）降低了类之间的耦合，便于扩展。

在本文的某型无人机飞行控制系统设计过程中，飞控计算机、离散输入输出模块采用+5V 的电源，ADC 转换器采用+12V 的电源，垂直陀螺采用+10V 的电源，三轴角速率陀螺采用+15V 电源等。如果分别提供+5V，+10V，+12V，+15V 的电源，将会增大硬件成本和硬件结构，同时不利于扩展。因此，当只提供一个 27V 的电源时，就需要使用电源转换板了，这就是适配器模式。电源转换器便是一个适配器。在飞控系统的建模中可使用适配器模式表示出这种适配关系。具体分析如下：

（1）目标（Target）角色：舵机（Gear），飞控计算机（FCC），A/D 转换器（AD）。

（2）被适配（Adaptee）角色：机载电源（Power）。

（3）适配器（Adapter）角色：电源转换器（PowerCBoard）。

采用适配器模式后的类图结构如下：

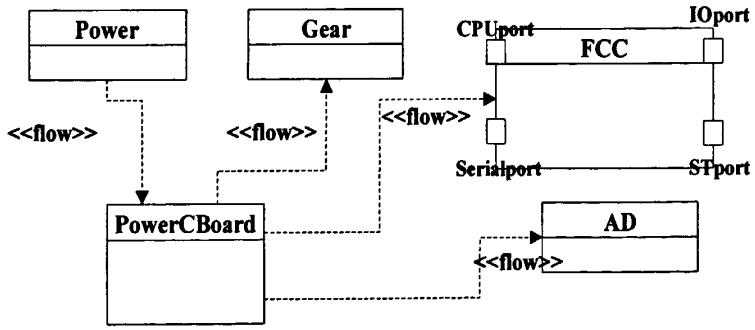


图 3.8 电源适配的类图结构

在飞控系统建模中采用适配器模式，其优点具体如下：

（1）精简了系统的建模结构，不用为舵机等这类对象分别设计一个电源接口。在硬件实现上也只需一个电源就可以，不用使用多个电源。因此能节约成本，简化硬件结构。

（2）降低了电源类和舵机等对象之间的耦合，便于扩展。以后有新的电压出现我们只需要在适配器上增加相应的电源转换接口就行，不会影响到其它。

3.4.4 外观模式的应用研究

外观模式（facade）又称门面模式，它为子系统中的一组接口提供了一个一致的界面。外观模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。子系统是指在设计中为了降低系统的复杂性，根据一定的规则（比如业务、功能）对系统进行划分而产生的新系统。子系统中封装了一些类。外观模式的组成如下：

（1）外观角色（facade）：这是外观模式的核心。它被客户角色调用，因此它熟悉子系统的功能。它内部根据客户角色已有的需求预定了几种功能组合。

（2）子系统角色：实现了子系统的功能。对它而言，facade 角色就和客户角色一样是未知的，它没有任何 facade 角色的信息和链接。

（3）客户角色：调用 facade 角色来完成要得到的功能。

外观模式的类图结构如下：

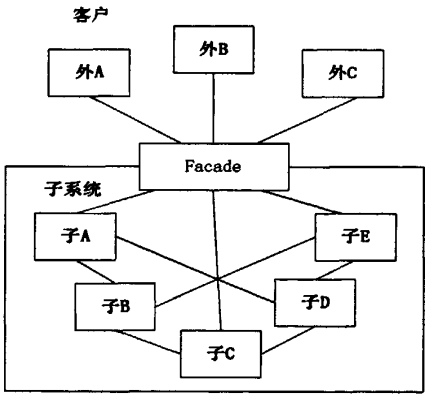


图 3.9 外观模式的类图结构

使用外观模式的优点：

（1）为复杂的子系统提供简单的接口；

（2）降低了子系统与外部系统的耦合，提高了子系统的独立性与可移植性，便于扩展和更改；

（3）外观模式能明显改变图 3.10 所示的子系统与外部机构的零乱关系。通过比较图 3.10（没有使用外观模式的类图结构）与图 3.9（使用外观模式后的类图结构）得到：外观模式可以使系统的内外联系更加有条理，更加便于控制，因此，系统也更加稳定。

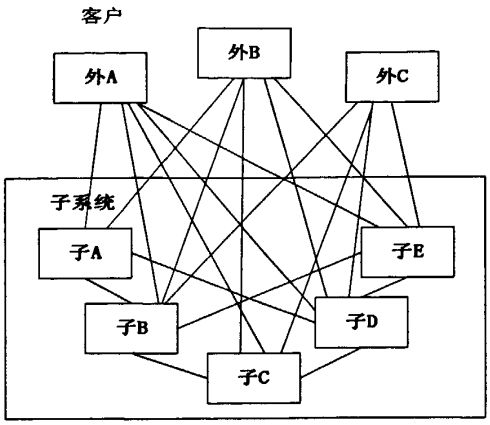


图 3. 10 子系统与外部系统的类图结构

飞控计算机内部的 CPU 模块、定时计数器模块、A/D 转换模块等与外部传感器等的详细关系，就可以用外观模式描述，具体分析如下：

（1）外观角色（facade）：RS232/RS485/RS422 等标准接口，它包括主模块接口(CPUPort)，串行接口(SerialPort)，输入输出接口(IOPort)，处理模块接口(STPort)等。

（2）飞控计算机系统(FCC)的子系统角色： CPU 模块，定时计数器模块(TCDevice)，IO 离散输入输出模块（IOChannel），串行通道模块（UART），A/D 转换模块（ADC），标准总线（STD_BUS）。

（3）客户角色：舵机(Gear)，垂直陀螺(GyroSensor)，GPS 传感器(GPS)，任务设备（DIS），三轴角速率陀螺(RotationSensor)，高度/空速计(hvSensor)，磁力计(Magnetometer)，终端测试设备（TestTerminal）。

对应的类图结构如下：

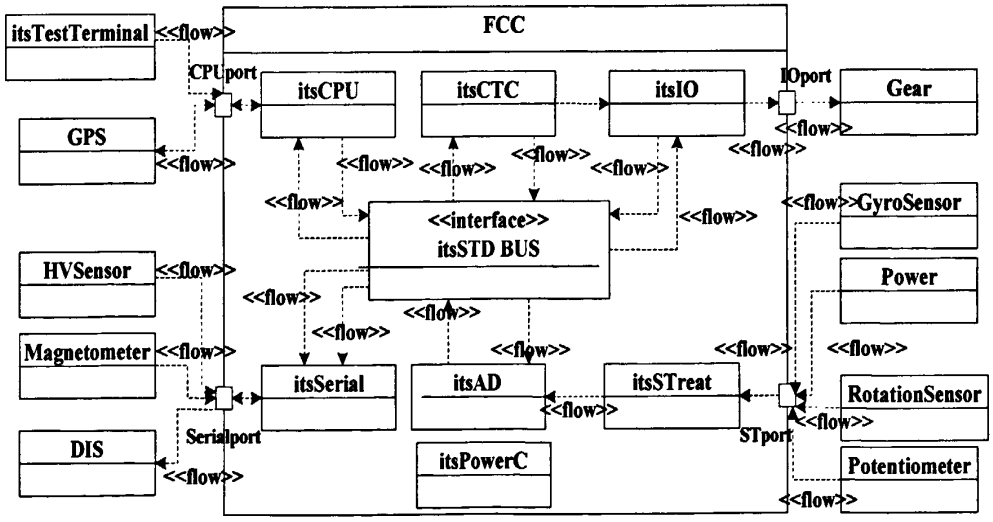


图 3. 11 飞控计算机与外部交互的类图结构

在飞控系统建模中采用外观模式后，其优点具体如下：

（1）为复杂的飞控计算机子系统提供简单的 RS232/RS485/RS422 等标准接口。

（2）降低了飞控计算机子系统与外部机构的耦合，提高了飞控计算机子系统的独立性与可移植性，便于扩展和更改；

（3）采用外观模式后，飞控计算机内外部界限一目了然。飞控计算机内部机构与传感器、舵机等的数据传输也相当清晰。整个系统的内外联系变得非常有条理。

3.4.5 状态模式的应用研究

状态模式（state）允许一个对象在其内部状态改变时改变它的行为。状态模式的各组成角色为：

（1）使用环境(Context)角色：客户程序是通过它来满足自己的需求。它定义了客户程序需要的接口，并且维护一个具体状态角色的实例，这个实例来决定当前的状态。

（2）状态(State)角色：定义一个接口以封装与使用环境角色的一个特定状态相关的行为。

（3）具体状态(Concrete State)角色：实现状态角色定义的接口。类图如下，其结构与策略模式非常相似。

状态模式的类图结构如下：

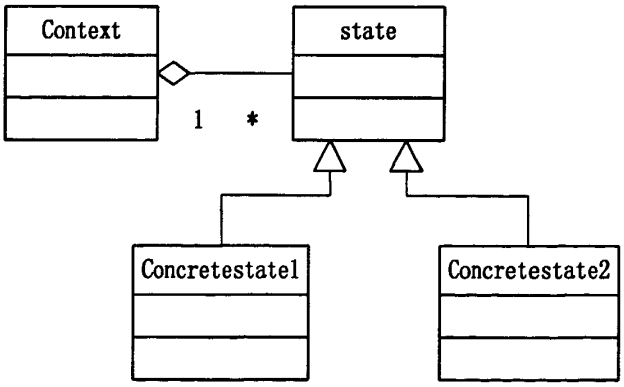


图 3.12 状态模式的类图结构

从前面几种设计模式的应用和目前关于设计模式的研究来看，似乎设计模式只是在系统的结构建模中得到复用，也即是在 UML 建模中对类或对象之间的结构关系进行复用。这也许跟某些设计模式的本身有关，如工厂方法模式就有两层含义：第一是抽象的类产生具体的对象；第二是具体的对象作用于其它具体对象。其两层含义的实质就是体现工厂方法模式对类或对象之间的结构关系进行描述。那么是否有这样的设计模式，可以在行为状态建模中复用？通过研究应用发现，本节的状态模式就可以在行为状态建模中复用。

根据状态模式的定义：允许一个对象在其内部状态改变时改变它的行为。无人机的行为取

决于它的飞行模态，并且运行时根据飞行模态改变它的行为。因此，可以在建立无人机飞行模态的状态图时复用状态模式，结构关系如图 3.13 所。该设计模式的具体角色如下：

（1）使用环境（Context）角色：飞控系统。

（2）状态（State）角色：控制模态。

（3）具体状态（Concrete State）角色：内控模态，外控模态，混控模态。

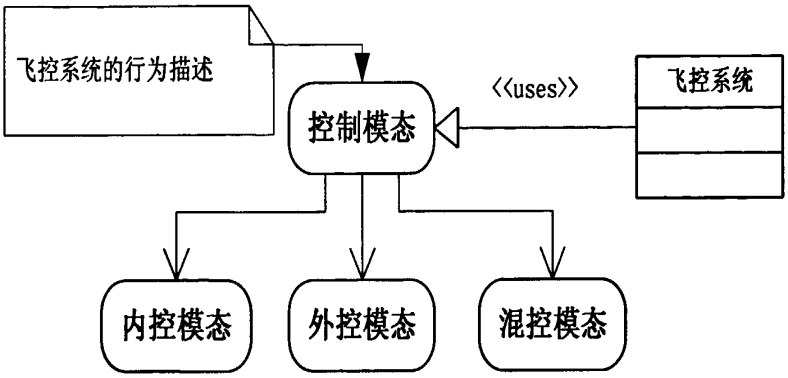


图 3. 13 控制模态的结构关系图

根据图 3.13 所示的状态模式的结构关系图建立控制模态的状态图，如图 3.14 所示。

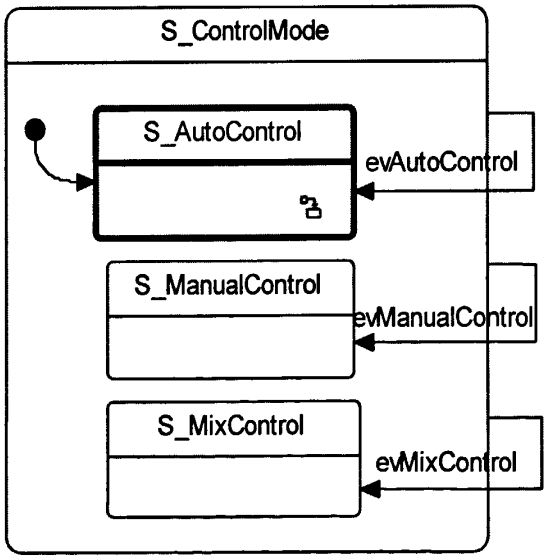


图 3. 14 控制模态的状态图

图中控制模态(S_ControlMode)包括以下各子状态：内控状态(S_AutoControl)，外控状态(S_ManualControl)，混控状态(S_MixControl)。通过发送 evAutoControl、evManualControl、evMixControl 等触发消息，可以控制无人机处于内控、外控、混控等三种不同的控制模态。当无人机处于其中任意一种模态时，只能响应该模态的行为，其它模态的行为不能响应。

使用状态模式的优点如下：

（1）对于用户来说，状态变化是透明的。用户可以通过状态图清晰地看到状态的转变。

（2）简化了系统设计，多个状态对应一个状态类，无须为每一个状态都设计一个状态类。上述的内控状态，外控状态，混控状态都有各自的行为，可以建立多个行为状态图。虽然有多个行为状态图，但它们都是飞行控制系统类的状态图。

3.4.6 MVC 模式的应用研究[27][32][33]

类的 MVC 模式（模型/视图/控制器，Model-Veiw-Controller）是独立于 GOF 的 23 种设计模式之外的设计模式，其基本思想是将模型、视图和控制器三者分离开来，各自独立的完成自己的功能，使其中任何一部分变化对其余部分的影响降到最低，甚至消除。 MVC 设计模式可用于搭建嵌入式系统框架。

在对飞行控制系统进行设计的过程中，采用 MVC 模式搭建系统数字化设计方法中的虚拟样机平台构架。具体角色如下：

（1）控制器（Controller）：主要由 Rhapsody 软件担当，它既能启动 VC 界面（View 角色），达到控制视图的功能，又能建立 UML 模型（Model 角色）并触发模型，达到控制模型的功能。

（2）模型（Model）：主要是飞控系统的 UML 模型和动力学部分的 Simulink 模型。

（3）视图（View）：在 VC 软件中建立的人机交互界面。

这种 MVC 设计模式的详细结构图如下：

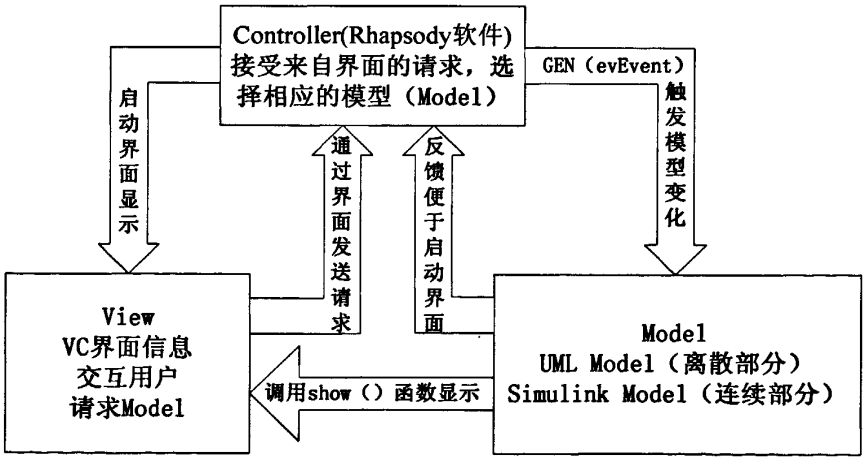


图 3.15 MVC 模式应用图

在通常的应用实例中，MVC 设计模式的精髓体现在以下三点：

（1）模型 Model 只有一个，视图 View 有多个

（2）视图必须保证它的显示能正确地反映模型的状态，一旦模型的数据发生了变化，视图要做相应的改变。这样可以为一个模型提供多个不同的视图表现形式，也能够为一个模型创建新的视图而无需重建模型。

（3）控制器控制着视图和模型，既有主导作用，又有中介的角色。

本文所使用的 MVC 设计模式与通常所使用的 MVC 设计模式有所不同，主要体现在：

（1）模型 Model 有多个，视图 View 只有一个。

（2）模型必须保证它的变化能正确地反映视图（界面）的转变。因为在飞控系统设计中，我们将会设计一个人机交互界面，这个人机界面是相对稳定不变的。当我们操纵界面，通过界面让无人机处于不同的模态，或发送不同的指令时，模型要做相应的改变，达到控制无人机的目的。这样可以为一个视图提供不同的模型来分别响应，也能够为一个视图创建新的模型而无需设计视图，最多做一些局部的改变。

使用 MVC 设计模式后的优点体现在：

（1）低耦合，高内聚。系统的虚拟样机既包括在 Rhapsody 软件上建立的离散模型，还包括在 Simulink 软件上建立的连续模型。为了不致使单独建立两个模型时对所建立的 VC 界面造成影响，使用 MVC 设计模式，使得模型与界面相对独立，具有松耦合关系。而它们各自高内聚，如无人机的动力学部分（Simulink 模型）和无人机飞控系统部分（UML 模型）在 Rhapsody 软件模块中通过接口紧密衔接，达到高内聚，从而保证了系统稳定性。

（2）便于扩展，方便更改。无论是对模型还是对视图（界面）作更改或扩展，都不用全盘修改或进行大量测试，就能顺利进行更改和扩展。

3.5 设计模式指导 UML 建模

在运用设计模式进行实际工程设计的基础上，研究如何应用设计模式指导 UML 建模。

3.5.1 UML 建模方法的不足

本文主要通过使用 UML 建模的方法来设计飞行控制系统。前面已经阐明，本文的建模和设计的本质是一样的。本文的第四章主要就是对飞行控制系统建模，即对飞行控制系统进行设计。使用 UML 建模方法存在以下不足：

（1）UML 缺乏设计方法指导

由于 UML 只是一种建模方法的表示，所以缺乏对过程和方法的支持。当然这有其客观原因：过程难以推广，标准的交流工具显得比较急迫。但在使用 UML 过程中，人们却无法回避这种问题：虽然 UML 有很强的表现力，可以精确而方便地表达设计意图，但在系统设计过程中却没有现成方法的指导，而显得力不从心。况且仅有 UML 也不能开发出真正好的系统，不能保证系统的质量，所以方法的指导是必不可少的。

（2）UML 过于复杂

UML 除了具有丰富的表达力，其复杂性也是惊人的。虽然 UML 的开发者称：只要 UML 中 20%的内容就可以表达实际应用中 80%的系统，但 UML 中并未给出哪些内容是必要的，哪

些是可选的。这给用户的使用带来了不小的麻烦。从另一个角度说，UML 虽然包罗万象，但也不可能将全部好的建模技术包括在内，比如 CRC 卡技术，虽然并没有在标准 UML 中出现，但在分析类责任的时候仍是一种比较实用的技术[34]。

（3）UML 元素间缺乏平滑的过渡

领域知识是十分复杂的。要全面、准确地描述它，只有从不同的角度来考察，才能真正把握领域的需求。为了达到这一目的，UML 定义了九种不同类型的图形，希望通过它们之间的有机结合来构造一个完整、一致的系统。分析建模过程及各模型图之间的关系会发现，虽然大多数模型图之间都保持着一定的内在联系，但静态图却无法向动态图平滑过渡，这不仅会带来随设计者不同而产生的随意性，还难以保证建模过程的连贯性和一致性。UML 中将类图分为三个不同的层次：观念层、说明层和实现层，以试图解决这一问题。这三个层次的类图随开发过程逐渐深入而采用迭代的方法来深化设计细节，但是，这一层次的概念固有的模糊性和复杂性，本身就难以理解和区分，使其在实际的应用过程中往往耗费大量的精力和时间。

3.5.2 设计模式在建立 UML 模型中的应用

设计模式既可以解决系统建模中的特定问题，又可以指导系统的 UML 建模。设计模式和 UML 建模方法都是面向对象的产物，而且彼此的思想是完全统一的，最终都是为了高效高质地建造一个可复用可维护的系统。针对 UML 建模方法的不足，设计模式从以下几个方面弥补了 UML 的缺陷。

（1）设计模式本身就是优秀的设计思想和设计方法，提供了设计过程中出现的问题的解决方案。设计模式本身就可以作为 UML 建模方法的指导。

（2）它捕获了面向对象设计的有价值的经验，并且用简洁、可复用的形式表达出来。

（3）设计模式已经被证明是一种行之有效的面向对象分析和设计的手段。它描述了模式的问题和使用条件，为其方便快捷的交流和使用指明了方向。同时设计模式为 UML 建模提供了方法的指导，而又没有因此影响到 UML 的功能。

（4）设计模式是前人经过反复验证的解决方案的结晶，并且有非常详细的描述和说明。由于它来自于从实际设计过程中，针对某一个具体方案和问题而提出的，使得设计的重点不在于如何使用 UML 的诸多元素，而在于如何体现设计思想，从而避免了在繁多的 UML 细节上花费过多的精力，并且使交流变得更加通畅。使用设计模式可以防止过早地陷入实现的细节。

设计模式能够明显弥补 UML 建模方法的不足，这是在飞控系统的 UML 建模中应用设计模式的一个极其重要的原因。

3.5.3 设计模式指导 UML 建模的步骤

根据前面设计模式的实际应用，对应用情况进行了大致归纳，提出了使用设计模式指导

UML 建模的步骤：

（1）研究特定的问题和需求场景，选择合适的设计模式。

（2）研究设计模式的结构图，确定参与者、协作者以及它们之间的关系。

设计模式的主要优点就是其良好的设计结构，因此，如果对某一个设计模式的结构不熟悉，将很难复用。同时还需要考虑设计模式的其它优点和不足，是否能改善以后的设计或影响将来的设计问题的解决，即要权衡使用该设计模式的利弊。如果应用该设计模式结构后能解决当前问题，但有可能对将来的设计造成很大的问题，则应当考虑使用该设计模式是否可行或有价值。

（3）参照模式结构图定义类和对象。

（4）添加其它属性和方法。

（5）实现执行模式中体现责任和协作的操作。这些协作和责任的操作可以通过 UML 的其它图形来实现，如状态图，顺序图，协作图等。

3.6 本章小结

本章首先详细介绍了设计模式，包括设计模式的基本概念、描述及分类；紧接着比较了设计模式与框架的区别，进一步明晰了设计模式的本质；再对设计模式的应用进行了研究，包括设计模式的优点和研究的意义；然后研究了如何在飞控系统建模中应用设计模式，一共应用到六种设计模式，包括五种 GOF 设计模式和 MVC 设计模式；最后在设计模式的实际应用的基础上探讨了应用设计模式指导 UML 建模，包括设计模式在 UML 建模中的应用分析和设计模式指导 UML 建模的步骤。本章的研究工作，为第四章进行飞行控制系统的建模打下了坚实的基础。

第四章　某型无人机飞行控制系统的虚拟样机设计

在已经搭建好的虚拟样机平台的核心模块 Rhapsody 软件上，采用模型驱动式柔性开发方法设计某型无人机飞行控制系统的虚拟样机，即建立该系统的 UML 模型，并进行数字化仿真和基于人机交互界面的交互式仿真，验证系统的行为和功能特性。系统的虚拟样机主要由需求模型、静态模型和动态模型构成。

4.1 模型驱动式柔性开发方法

本文所要设计的某型无人机飞行控制系统是一个以高性能嵌入式计算机系统为核心的数字式飞行控制系统，是一种复杂的嵌入式实时系统。针对此特点，本文采用模型驱动式柔性开发方法建立某型无人机飞行控制系统的模型。

4.1.1 模型驱动方法[21]

模型驱动方法(MDA)的核心思想是先抽象出与实现技术无关、完整描述系统的平台独立模型（Platform-Independent Model，PIM），再通过变换工具将 PIM 转换成与具体实现技术相关的平台模型（Platform-Specific Model，PSM），然后通过变换工具将 PSM 自动转换成代码。当需求有变化时，只需修改模型，然后重新自动生成代码。这种方法最大化了模型的价值[35]。

模型驱动方法的整个开发过程是由对系统的建模行为驱动的，其生命周期各个阶段产生的工件是能够被计算机理解的形式化模型，模型在整个生命周期中处于核心地位。

4.1.2 柔性开发模式

柔性开发模式(如图 4.1 所示)是指在需求工程的牵引下，首先建立系统的模型，再对模型进行模拟、分析和调整，然后进行从需求到建模的“自顶向下建模，由底向上修改”[36]。其开发流程如下：

（1）明确用户要求，确定需求优先级。

（2）为系统建立可执行的模型，通过对模型的模拟运行，分析模型是否满足用户需求和满足的程度。整个建模过程是自顶向下逐层细化的，而模拟修改则自底向上进行。

（3）在保证模型正确的基础上，进行代码的生成。同时考虑到对需求修改的灵活性和快速响应能力，可以采用“闭环开发”模式，即不仅支持从模型到代码的自动生成，还能支持从代码到模型的逆向变换，达到了模型和代码的双向工程关联，从而保证了模型和代码间的一致性，使系统的扩充、增删和维护工作顺利进行[15]。

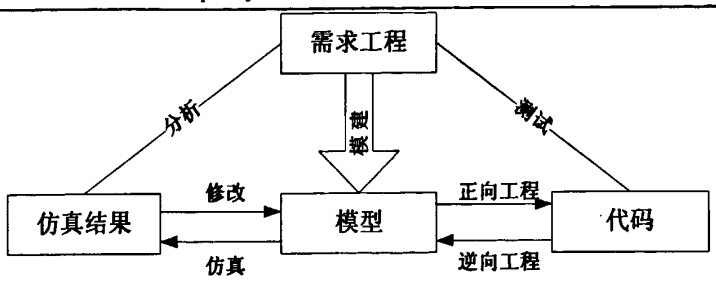


图 4.1 柔性开发模式

4.1.3 模型驱动式柔性开发方法

为了更好的运用 UML 语言和 Rhapsody 软件设计飞行控制系统的数字化模型，本文提出了一种模型驱动式柔性开发方法。这种开发方法由模型驱动开发方法与柔性开发模式相结合而成。

模型驱动式柔性开发方法是在整个系统虚拟样机模型的设计中以模型驱动的思想为指导，即整个系统虚拟样机的设计是由建模行为驱动的，模型贯穿系统虚拟样机设计的各个阶段，而在各个阶段的各个模型的设计过程中采用柔性的开发模式。模型驱动式柔性开发方法的流程图如下：

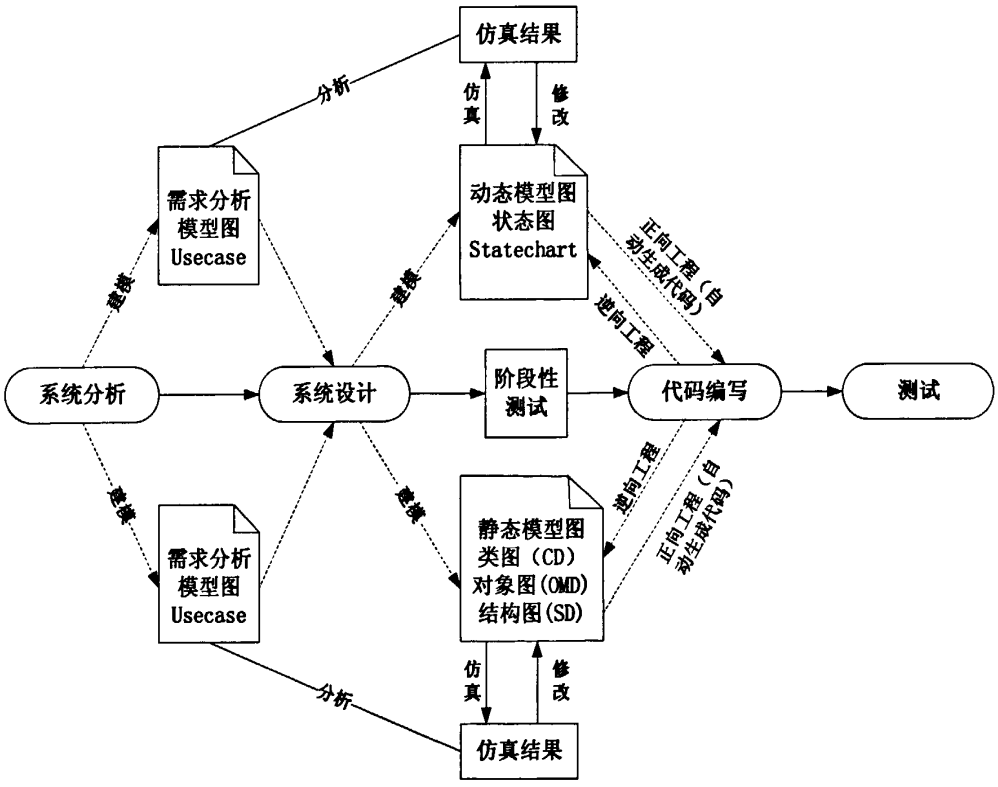


图 4.2 模型驱动式柔性开发流程

设计好的系统虚拟样机主要是由需求模型，静态模型和动态模型组成，在 Rhapsody 软件模

块上可以运用 UML 语言完成这些模型的设计，充分体现了模型驱动的思想。同时，Rhapsody 软件具有边设计边仿真验证的功能，给柔性开发模式提供了强大的支持。因此，在本文搭建好的虚拟样机平台上，采用模型驱动式柔性开发方法设计系统的虚拟样机是非常适用的。

4.2 系统分析

在确定系统设计方案、提出系统各种要求以后，下一步工作就是完成系统需求分析。系统需求分析的目的是深入描述系统的功能和行为。在本文搭建的虚拟样机平台上，需求分析模型是通过用例建模来完成的。

4.2.1 用例建模的主要目标[14]

用例建模是 UML 建模的一部分，也是最基础的部分。其主要功能是通过建立的用例模型来表达系统的功能性需求或行为，用例模型是表达外部事物(执行者)与系统之间交互的工具。使用 UML 语言进行用例建模的主要目标包含以下几点：

（1）将需求规约变为可视化模型，即将以前的文字描述需求变为图形描述需求，更为形象直观，也充分体现了 UML 这种图形化建模语言的优势所在。

（2）给出清晰、一致的关于系统做什么的描述，确定系统的功能要求。根据第二章提出的系统设计要求，对系统需求的描述按以下三级给出：

●FCS 级：主要功能是控制飞行模态，管理任务等。

●FCC 级：主要功能是数据交互，数据采集，发送指令等。

●UAV 级：主要功能是执行任务（包括飞行过程中的左/右转弯，上/下爬升等功能），起飞，着陆等。

（3）提供从功能需求到系统分析、设计、实现等各阶段的度量标准，即系统开发的后续各个阶段以需求分析为参照。

（4）为最终的系统测试提供基准，据此验证系统是否达到功能要求。

（5）为系统进度管理和风险管理提供依据。依据用例图，监看系统已经实现的功能。

4.2.2 用例建模的步骤[14]

用例建模最终是要建立起系统的用例图（Use Case Diagram，UCD），用例图是对系统需求分析的描述，包括执行者，用例，系统三要素。按照以下用例建模步骤确定用例图中的各个要素。

（1）确定系统的范围和边界

定义系统的边界是一项重要的活动，目的是为了明确什么在系统之内，什么在系统之外。系统的需求直接影响到系统边界的确定，典型的系统边界包括：

●整个系统组织。如本文确定的第一级 FCS 级，表示无人机飞行控制系统。

●一个系统的子系统。如本文确定的第二级 FCC 级，表示飞控计算机系统。

●系统的硬/软件边界。

（2）确定系统的执行者

执行者（Actor）是指在系统外部与系统交互的人或其它系统，它以某种方式参与系统内用例的执行。定义执行者应该注意以下几个问题：

●执行者之间可以有继承关系。

●执行者代表的是一种角色，而不是具体的人。

●执行者可以分为主动执行者和被动执行者。

在查找参与者的过程中，可以询问以下问题以帮助确定执行者[37]：

●谁负责提供，使用或删除信息？

●谁将使用此功能？

●谁对某个特定功能感兴趣？

●在组织中的什么地方使用系统？

●谁支持和维护系统？

●系统有哪些外部资源？其他还有哪些系统将需要与该系统进行交互？

按照上述问题分级确定各级的执行者，具体分析如下：

●FCS 级：谁发送/接收(提供，使用)系统信息？地面测控终端(执行者)。

在组织中的什么地方使用系统？无人机(执行者)。

谁需要管理、维护和维持系统的日常运行？飞控计算机(执行者)。

●FCC 级：谁发送/接收(提供，使用)系统信息？传感器(执行者)、舵机(执行者)。

●UAV 级：谁将使用此功能(即谁操纵无人机)？地面站操纵人员(执行者)、外控手(执行者)。

（3）确定用例

用例(Use Case)是系统所提供的一个功能（或者系统提供的某一特定用法）的描述。用例捕获某些用户可见的需求，实现一个具体的用户目标。用例由执行者激活。用例可大可小，但它必须是对一个具体的用户目标的完整描述。通常可以按以下方式确定用例：

●系统能做什么？

例如：UAV 能起飞（用例），着陆（用例）等。

●系统在工作中发生了哪些事件及结果？

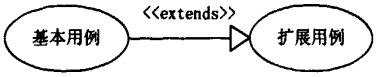
例如：FCC 级工作中发生了数据采集（用例），数据交互（用例）等。

●执行者可以做什么？

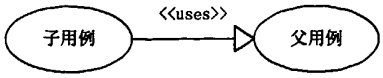
例如：UAV 级的外控手可以操纵无人机（用例）等。

（4）确定用例之间的关系

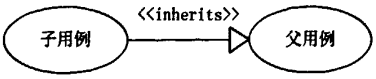
用例之间有以下四种关系



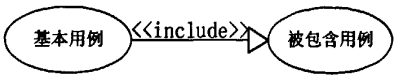
扩展关系



使用关系



继承关系

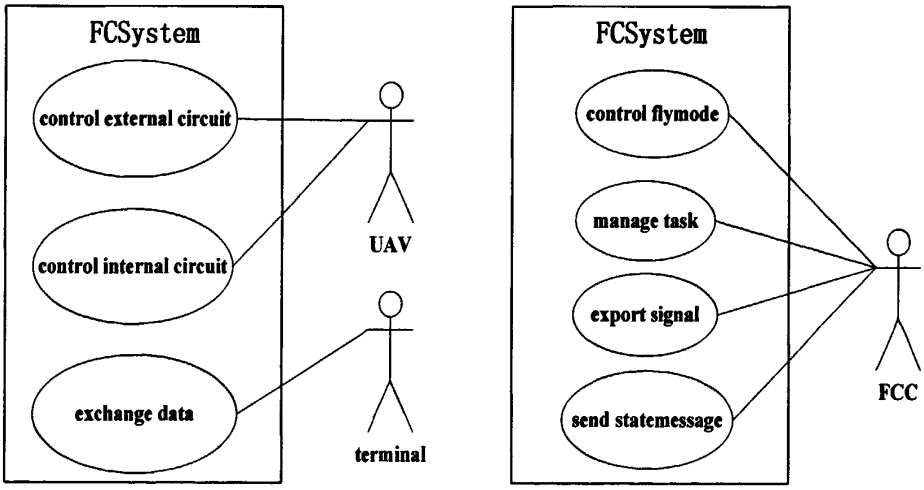


包含关系

4.2.3 建立系统的需求模型

在确定系统用例图中的三要素后，在 Rhapsody 软件上建立系统的需求模型，即用例图。

（1）建立系统 FCS 级的需求模型，如图 4.3 所示.



(a)　　(b)

图 4.3 飞行控制系统的需求模型

图中的执行者说明：UAV：无人机；terminal：测控终端；FCC：飞控计算机。

图中的用例说明： control external circuit：控制外回路；exchange data：交换数据；

control internal circuit：控制内回路；manage task：任务管理；

control flymode：控制飞行模态；export signal：输出信号；

send statemessage：发送状态信息。

（2）建立系统 FCC 级的需求模型，如图 4.4 所示。

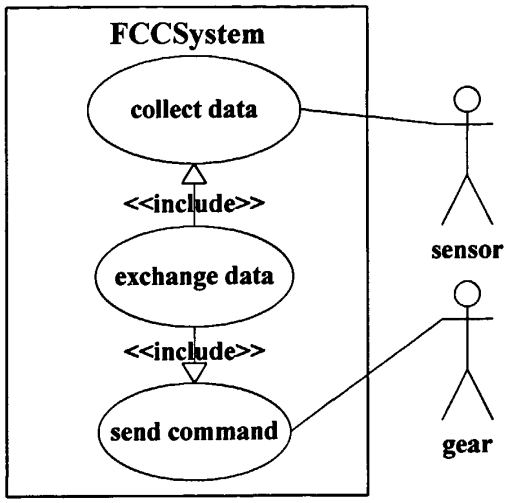


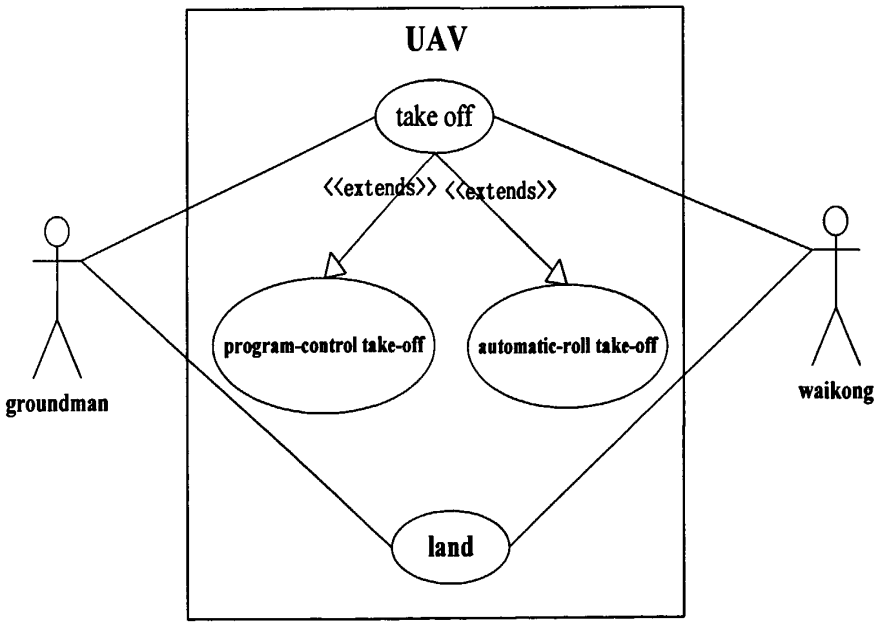
图 4.4 飞控计算机系统的需求模型

图中的执行者说明：sensor：传感器；gear：舵机。

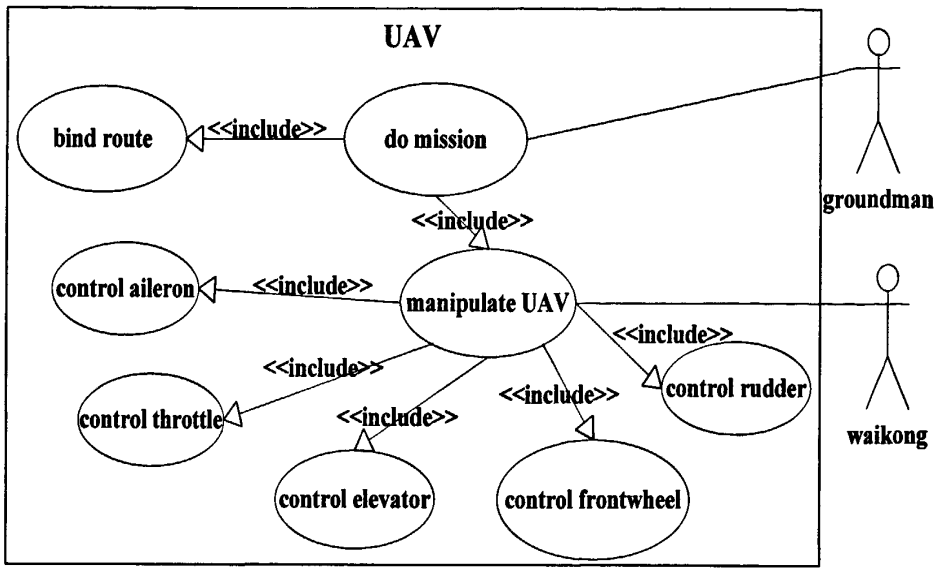
图中的用例说明：collect data：采集数据；exchange data：交换数据；

send command：发送命令。

（3）建立 UAV 级的需求模型，如图 4.5 所示。



（a）



(b)

图 4.5 无人机的需求模型

图中的执行者说明：groundman：地面操纵人员；waikong：外控手。

图中的用例说明：take off：起飞； program-control take-off：程控弹射起飞；

automatic-roll take-off：自动滑跑起飞；land：着陆；

do mission：执行任务；bind route：航迹绑定；

manipulate UAV：操纵无人机；control throttle：控制油门舵机；

control aileron：控制副翼；control frontwheel：控制前轮舵机；

control rudder：控制方向舵；control elevator：控制升降舵机。

4.3 系统设计

建立好无人机飞行控制系统的需求模型后，我们完成了模型驱动式柔性开发流程的系统分析阶段的工作，在此基础上我们进入模型驱动式柔性开发流程的系统设计阶段。系统设计阶段主要是在前面工作的基础上，设计无人机飞行控制系统的静态模型和动态模型。在设计系统的静态模型和动态模型之前，需要先确定系统的类与对象。

4.3.1 系统的类与对象的确定[14]

类是一组拥有相同的属性、操作、方法、关系和行为的对象的描述符(descriptor)。一个类代表了被建系统的一个概念。根据模型种类的不同，此概念可能是现实世界中的(对于分析模型)，或者也可能包含算法和计算机实现的概念(对于设计模型)[38]。

由于静态模型是以类和对象为基本的组成要素，而动态模型是对类或对象之间的行为功能

的描述，因此在设计系统的静态模型和动态模型之前，我们必须确定系统的类和对象。由于类是一组拥有相同的属性、操作、方法、关系和行为的对象的描述符，因此确定了类就等同于确定了对象。我们按以下的步骤确定系统的类。

（1）发现潜在的类。按照以下几个标准去发掘可能存在的类。

①与系统交互的角色：与飞行控制系统交互的角色，如：测控终端(TestTerminal)等。

②系统工作的环境场所。

③概念实体，发生的事件。如：电源转换模块(Power)等。

④所包含的设备或其它相关设备。例如：与飞行控制系统相关的设备有舵机(Gear)，传感器（Sensor），飞控计算机(FCC)等。

⑤与系统有关的外部实体。如：任务设备(DIS)等。

（2）标识类和对象名的原则。

①使用单个词来标识类或对象。如：舵机(Gear)等。

②类名或对象名必须有意义，简洁明了，含义明确，易于理解。

③尽量使用用户熟悉的行业标准术语。如：GPS传感器(GPS)，无人机(UAV)等。

（3）确定系统的类和对象。按照以下标准确定系统的类。

①关键性。缺少这个类或对象的信息，系统就不能工作。如升降舵机(itsEleavator)，左副翼(leftAilerons)，右副翼(rightAilerons)，左方向舵机(leftRudder)，右方向舵机(rightRud-der)，油门舵机(itsThrottle)，前轮舵机(itsFrontWheel)等。

②信息含量。选择信息量较大的类或对象作为最终的类或对象。如飞控计算机(FCC)，整个飞控软件就是安装在飞控计算机上，飞控软件的行为功能就可以看成是飞控计算机的行为功能，由飞控计算机的动态模型来体现。

③关键外部信息。问题空间中的外部实体和系统必须产生或消费的信息。例如：从测控终端(TestTerminal)发送过来的信息。

4.3.2 系统静态模型的设计[13] [39]

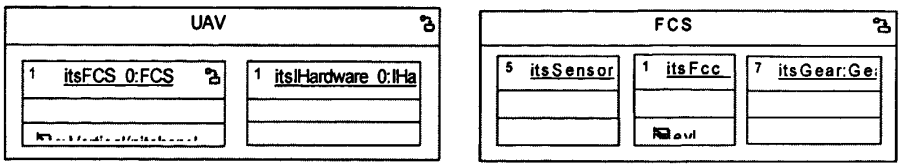
任何建模语言都是以静态建模机制为基础，UML 建模语言也不例外。通过静态建模，建立系统的静态模型。静态模型是相对固定不变的模型。在静态建模阶段，将从系统的内部结构和静态角度，分析和描述系统中的各类实体(对象和类)以及它们内部和彼此间的关系，确定实体功能范围的约束和限定，建立系统的粗略框架，再逐步细化其内部功能需求，最终建立系统的静态模型。静态建模主要是对飞行控制系统的体系结构建模，其结果是建立逻辑视图，主要包括类图(Class Diagram)和对象图(Object Model Diagram)等。

设计系统的静态模型，将充分体现柔性开发模式的“自顶向下建模，由底向上修改”和 UML

语言支持模型驱动、以体系结构为中心、以增量和迭代的方式进行系统设计，即在设计系统的静态模型时，采用自顶向下逐层递进的纵向模式设计系统体系结构的静态模型，每层的静态模型设计又以横向的模式拓宽。

（1）系统顶层结构图设计

依据第二章中给出的确定类与对象的方法，给出顶层结构图的类与对象，如图 4.6 所示。无人机系统包括飞行控制系统(itsFCS_0)和硬件部分(itsIHardware_0)等。飞行控制系统包括飞控计算机(Fcc)，5 个传感器(itsSensor)，7 个舵机(itsGear)等。



（a） 无人机类图　　（b） 飞行控制系统类图

图 4.6 无人机静态结构图

依据第二章设计的系统详细结构和确定的类与对象，在系统分析阶段的功能需求模型的基础上，确定各个类之间的联系，建立飞行控制系统顶层设计结构图，如图 4.7 所示。

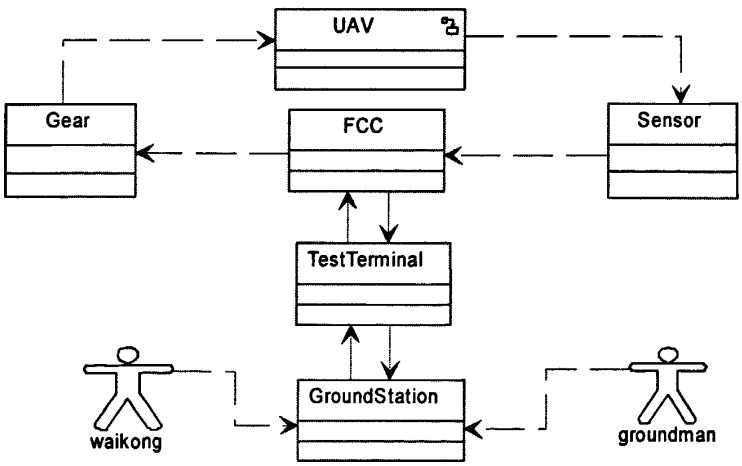


图 4.7 飞行控制系统顶层设计结构图

图 4.7 中，飞行控制系统主要是其飞控计算机(FCC)接收从地面测控站(GroundStation)上行信道经机载无线电测控终端(TestTerminal)送来的控制命令及数据，机载传感器测量各个有关的无人机运动状态参数，经机载飞行控制计算机(itsFCC)处理，输出控制指令到各个执行机构及有关设备，实现对无人机各种飞行模态的控制和任务设备的管理。同时，飞行控制系统亦把无人机的飞行状态数据及发动机、机载电源系统、任务设备工作状态参数实时传给无线电终端(TestTerminal)，经无线电下行信道发送回地面站测控站(GroundStation)，为地面控制人员

（groundman）和外控手（waikong）提供无人机及其任务设备的有关状态信息，以实时控制引导无人机完成飞行计划任务。

（2）飞控计算机及其相关部分的结构图设计

在完成飞行控制系统的顶层结构设计后，按照柔性开发模式的“自顶向下”的设计思路，设计系统的第二层结构图。

①依据第二章所设计的飞控计算机模块和系统分析阶段的需求模型，在 Rhapsody 软件上，运用外观模式设计飞控计算机的结构图，使用对象模型图来描述，如图 4.8 所示。

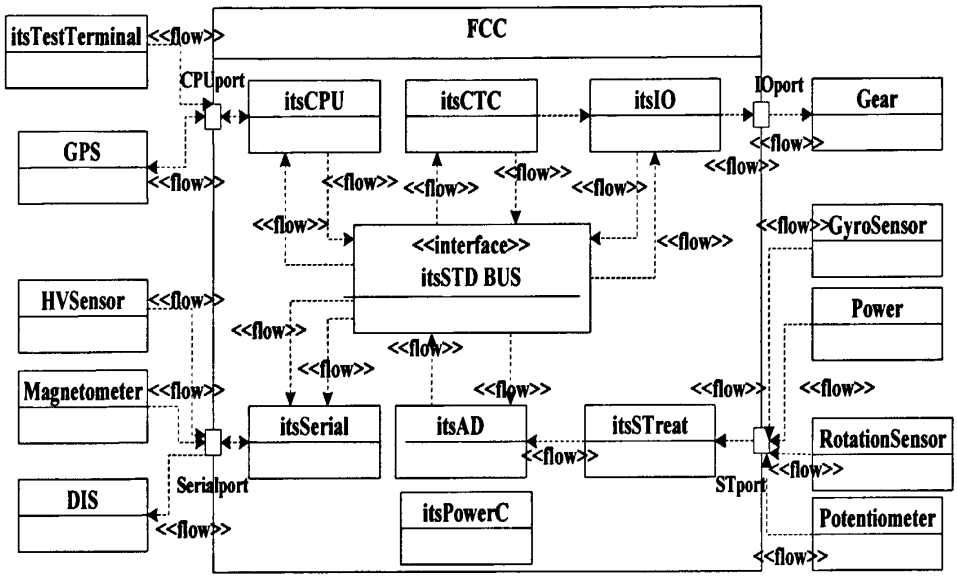


图 4.8 飞控计算机的结构图

图 4.8 中，飞控计算机包括 CPU 模块(itsCPU)，A/D 转换模块(itsAD)，串行通信模块(itsSerial)，定时计数模块(itsCTC)，离散 I/O 模块(itsIO)，模拟信号前置处理模块(itsSTreat)，电源转换模块(itsPowerC)等。各模块之间通过标准总线(itsSTD_BUS)进行联系。

其中由无线电测控终端(itsTestTerminal)送来的控制命令及数据、GPS 数据等信号直接传入 CPU 模块；气压高度/空速传感器(HVSensor)、三轴磁力计(Magnetometer)、任务设备(DIS)等设备的信号直接通过串行通信模块传入；垂直陀螺(GyroSensor)、角速率陀螺（RotationSensor）、电位计（Potentiometer）等设备的模拟信号经前置处理模块(itsSTreat)处理后，再由 A/D 转换模块(itsAD)转换为计算机所能接收的数字信号；计算机输出的执行命令经离散 I/O 模块(itsIO)的 IO 通道传送到执行机构——舵机(Gear)。

②图 4.8 中的电位计（Potentiometer）是其对应的舵机(Gear)的组成部分，用于测量对应舵机的位置。电位计测得的信号实时传入到飞控计算机，飞控计算机根据飞行要求和接收到的数据，输出给定数据到各舵机，各舵机根据给定数据进行适当调整，从而控制无人机按照给定的

要求飞行。运用第三章的简单工厂模式和工厂方法模式来设计电位机、飞控计算机和舵机之间的这种关系结构图。在 Rhapsody 软件上使用类图进行描述，如图 4.9 所示。

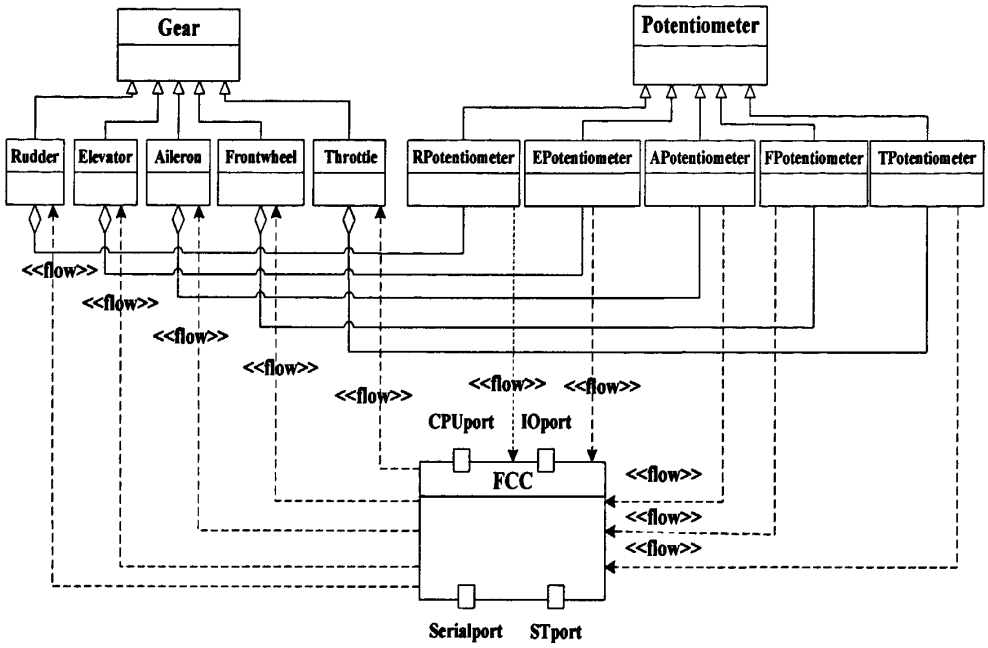


图 4.9 电位机、飞控计算机和舵机之间的类图

③垂直陀螺（GyroSensor），GPS 传感器(GPS)，三轴角速率陀螺(RotationSensor)，高度/空速计(HVSensor)，磁力计(Magnetometer)等传感器的数据会实时传入飞控计算机。运用简单工厂模式，设计的传感器与飞控计算机之间的结构图如下：

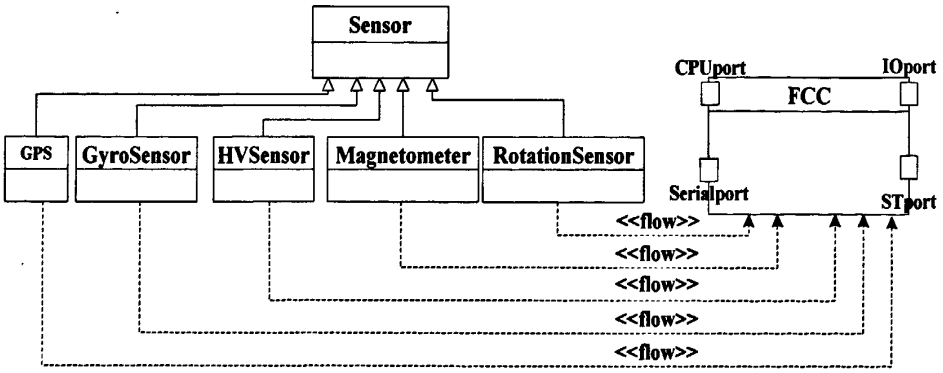


图 4.10 飞控计算机与传感器之间的类图

④根据市场调查，很难直接实现在所有设备功能、性能都满足要求的条件下，它们的供电电压也相同。如各传感器的供电电压就可能不一致。在进行系统设计时，不可能根据各设备需要的供电电压不同，提供不同的电源设备。因为使用多个电源设备不只是成本提高，还可能带

来一些不可预料的问题。因此，在第三章设计时，提出了电源转换模块。采用适配器模式设计电源转换模块的类图结构。使用类图描述其结构，如图 4.11 所示。在图 4.11 中，假设舵机(Gear)、飞控计算机（itsFCC）、A/D 转换模块的电压不同。Power 为提供的机载电源电压。

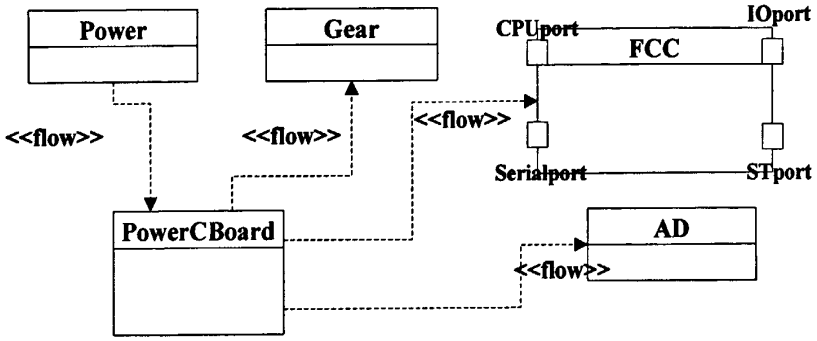


图 4.11 电源转换的类图

⑤在完成整个系统的建模和仿真后，还需要生成产品级代码，并下载到硬件平台进行半物理仿真。因此，需要在设计时预留出与硬件平台(IHardware)的接口，如图 4.12 所示.

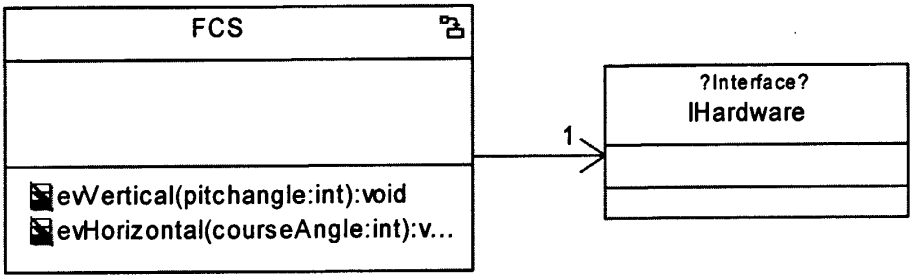


图 4.12 飞控系统与硬件平台的接口图

⑥设计出系统的数字化模型后，需要在 VC 软件模块上设计出无人机飞行控制系统的的界面，进行基于 VC 界面的交互式仿真。因此，还需要预留与 VC 软件的接口(IControlPanel)，如图 4.13 所示.

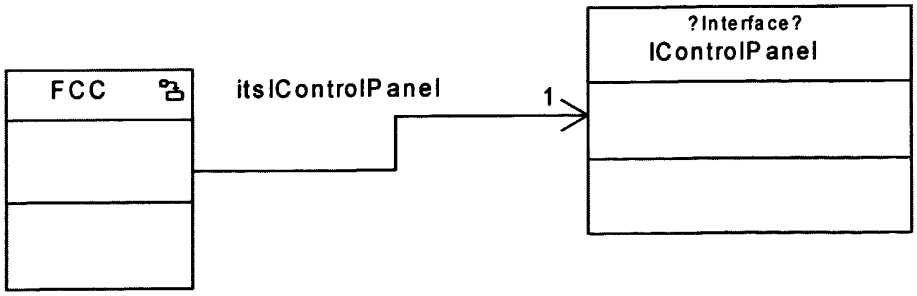
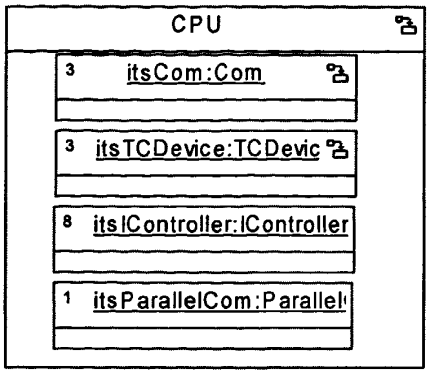


图 4.13 VC 软件的接口图

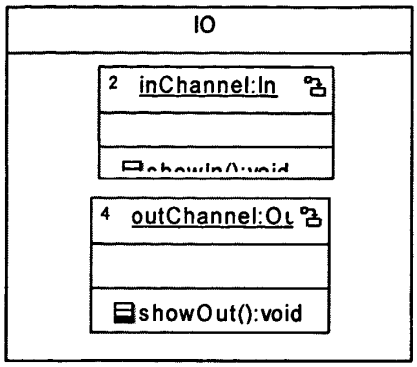
（3）飞控计算机内部各模块结构图的设计

在完成飞行控制系统的第二层结构设计后，按照柔性开发模式的“自顶向下”的设计思路，

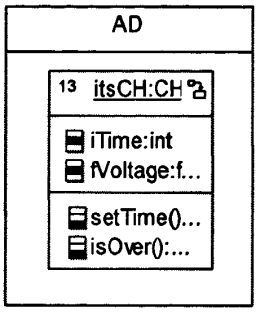
设计系统的第三层结构图。在第三层，主要是对飞控计算机的各模块进行详细的结构设计，在类图中设计它们的模型，如图 4.14 所示。



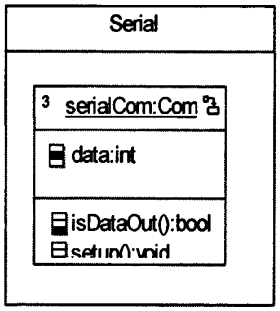
(a) CPU 模块的类图



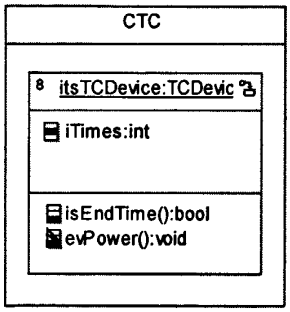
(b) IO 模块的类图



(c) AD 模块的类图



(d) 串行通信模块的类图



(e)CTC 模块的类图

图 4. 14 飞控计算机各模块的类图

图中 CPU 模块包含串行通道(itsCom)，定时计数器(itsTCDevice)，中断控制器(itsIController)，并行通道(itsParallelCom)；IO 模块包含离散输入通道(inChannel)，离散输出通道(outChannel)；AD 模块包含模拟输入通道(itsCH)；串行通信模块(Serial)包含串行通道(serialCom)；定时计数模块(CTC)包含定时计数器(itsTCDevice)。

至此，我们完成了无人机飞行控制系统全部静态模型图的设计。以上所有静态模型，如果出现错误，可以采取柔性开发模式的“自底向上”的方式修改。

4.3.3 系统动态模型的设计[13] [39] [40]

动态模型主要是描述系统的动态行为和控制结构。动态行为包括系统中对象生存期内可能的状态以及事件发生时状态的迁移，还包括对象之间动态交互关系，显示对象的交互过程以及交互顺序，同时描述了为满足用例要求所进行的活动及活动间的约束关系。UML动态模型主要用于描述系统的动态行为和功能，主要包括状态图(Statechart)、活动图(Activitychart) 、顺序图(Sequence Diagram)、合作图(Collaboration Diagram)等四种图。

Statechart是一种对基于离散事件的系统进行行为功能建模的图形化语言，具有层次性和并发性，可以清晰地描述基于状态的复杂系统。无人机飞行过程是多模态飞行控制律的切换过程，根据地面站控制指令，飞行控制系统在不同的控制律状态间进行切换，适用Statechart描述。采用柔性开发模式的“自顶向下”的设计方法，建立无人机飞行控制系统的动态模型。在设计每层的Statechart时，可以采用增量和迭代的方法进行反复的设计[22]。

（1）飞行控制系统的第一层动态模型设计

飞行控制系统的顶层结构设计包括飞行控制软件和无人机的动态模型。本文完成了无人机的动态模型设计。采用“自顶向下”的设计方式，完成了无人机从起飞到着陆全过程的静态模型设计。使用状态图来描述无人机从起飞到着陆的全过程。

从起飞到着陆的全过程的状态图如图 4.15 所示。无人机经过的状态依次是自检(selfTest)，静态联调(teststatic)，动态联调(testDynamic)，滑跑测试(testRoll)，起飞(takeoff)，飞行(flying)，返回(back)，着陆(landing)，检修(maintenance)等。其中，如果 selfTest、teststatic、testDynamic、testRoll 四环节中任意一环节出现问题都将回到检修，检修完毕重新进入各个环节测试，所有测试通过才可进入 takeoff 状态。

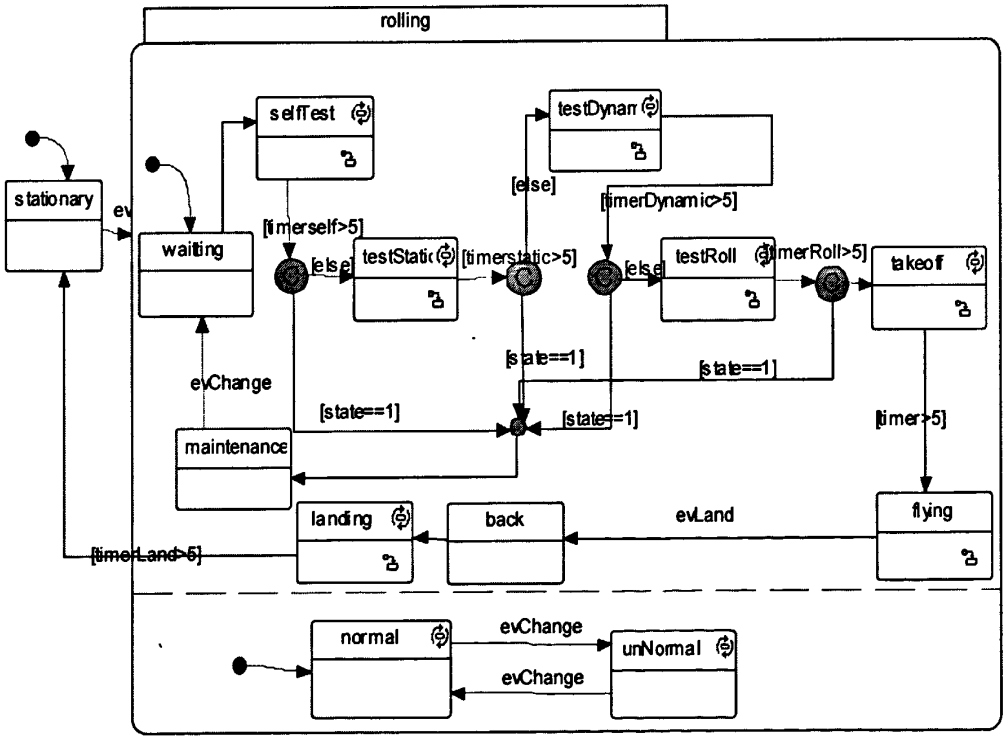


图 4. 15 无人机的行为功能图

根据第三章的状态模式(如图 3.14 所示)，建立无人机飞行状态(flying)的子状态图。如图 4.16 所示。

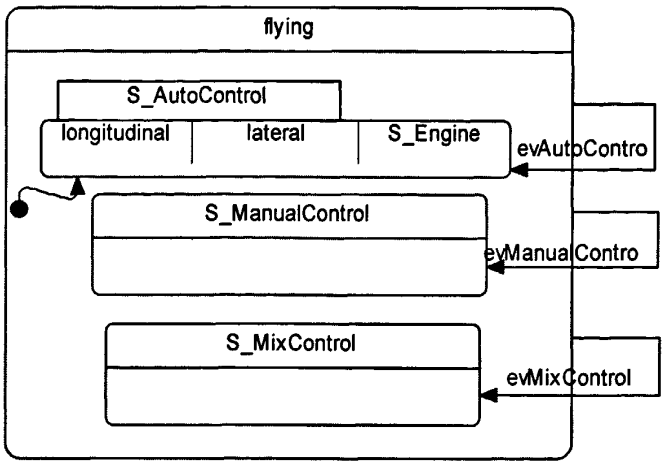
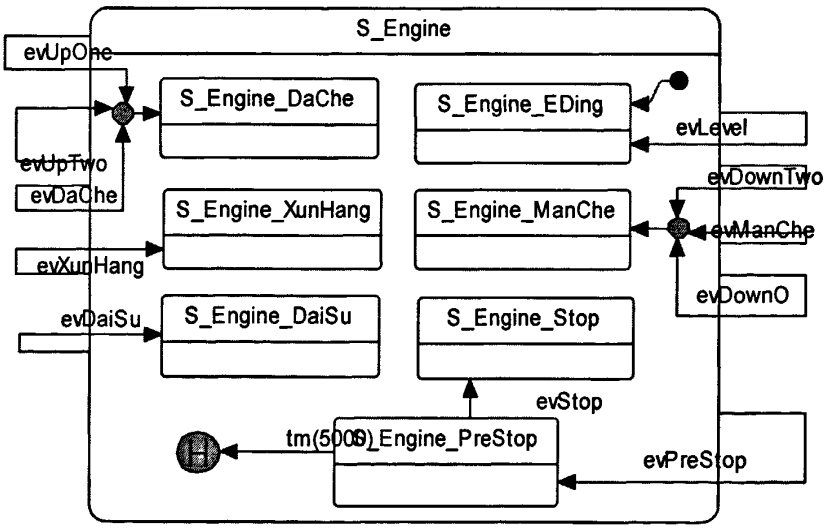


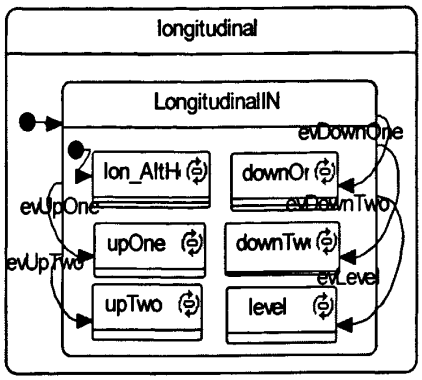
图 4.16 飞行状态的子状态图

图 4.16 中飞行(flying)的控制模态(S_ControlMode)包括以下各子状态：内控状态(S_AutoControl)，外控状态(S_ManualControl)，混控状态(S_MixControl)。其中内控状态又包括发动机子状态(S_Engine)，纵向模态子状态(longitudinal)和横向模态子状态(lateral)。通过发送evAutoControl、evManualControl、 evMixControl 等触发消息，可以控制无人机处于内控、外控、混控等三种不同的控制模态。当无人机处于其中一种模态时，只能响应该模态的行为，其它模态的行为不能响应。

分别建立内控状态的各子状态的状态图，以无人机纵向运行子状态以及发动机子状态为例，建立它们的状态图如下。



(a) 发动机的状态图



(b) 无人机纵向运行的状态图

图 4. 17 无人机飞行状态的状态图

处于内控状态时，无人机将处于一种自动模态飞行方式。在自动模态飞行方式下，只有地面操纵人员通过地面站发送的命令才有效，外控手是不能通过操纵杆来控制无人机的。

发动机有五种状态，五种状态的功率由大到小依次是：大车(S_Engine_DaChe)、额定(S_Engine_EDing)、巡航(S_Engine_XunHang)、慢车(S_Engine_ManChe)、怠速(S_Engine_DaiSu)、停车(S_Engine_PreStop)。通过控制发动机的风门开度，可以控制发动机处于不同的状态以及各状态的相互切换。其中预停是为了增加无人机的安全而设定的。在地面站发送停车命令时，必须先发预停命令后，在规定时间发停车命令才有效，从而保证了无人机的安全。

发动机的工作状态与纵向飞行模态密切相关。飞行控制系统将自动控制发动机状态随纵向飞行模态的切换而联动。参照一般飞行控制系统的设计，得到其对应的规律为：

爬升(S_UpOne/S_UpTwo):对应发动机状态为“大车”状态。可以响应“额定”，不响应其它状态。

下滑(S_DownOne/S_DownTwo):对应发动机“慢车”状态。可以响应“怠速”、“巡航”，不响应其它状态。

平飞(S_Level_HeightHold): 对应发动机“额定”状态。可以响应“大车”、“巡航”，不响应其它状态。

（2）飞行控制系统的第二层动态模型设计

飞行控制系统的第二层动态模型主要是用于描述系统中飞控计算机的各个模块的行为和功能。同样使用状态图建立各个模块的动态模型。

①建立CPU模块的动态模型图

CPU是飞控计算机的核心，用于所有信息数据的存储、运算和处理。建立CPU模块的状态图，如图4.18所示。它接受从传感器传输过来的输入数据信号(inputGPSSignal)和地面站传输过来的控制命令(inputCommand)，经运算处理(solution)输出处理后的控制信号(inputControlS)到执

行机构，输出解算出的姿态位置信息(outputAttitude)到地面站，由地面站接收(groundReceive)。

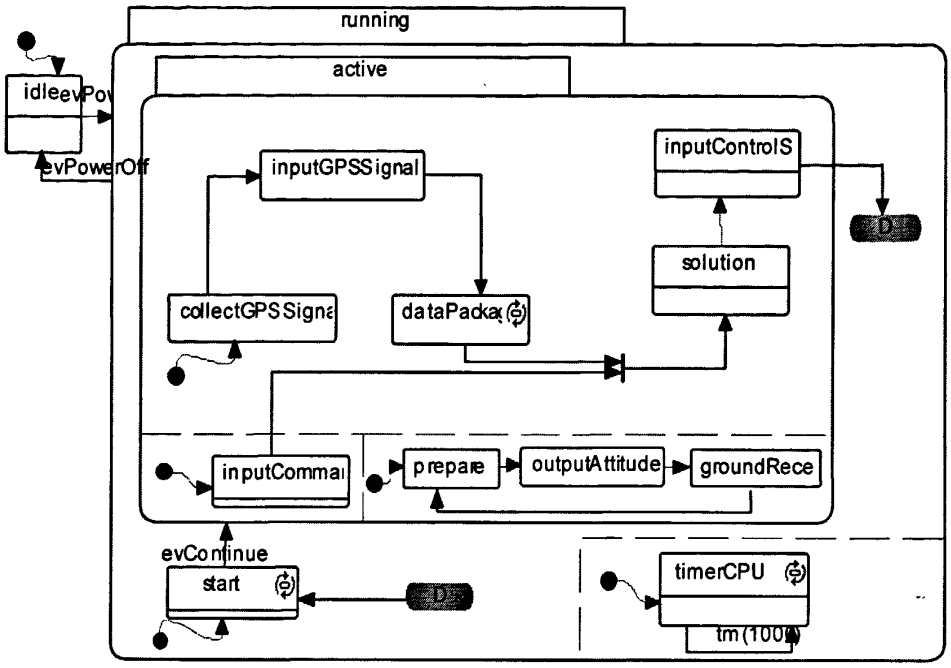


图 4. 18 CPU 模块的状态图

②建立AD模块的动态模型图

AD模块的行为功能主要由模拟输入通道(CH)的行为功能体现。建立模拟输入通道的状态图，如图4.19所示。

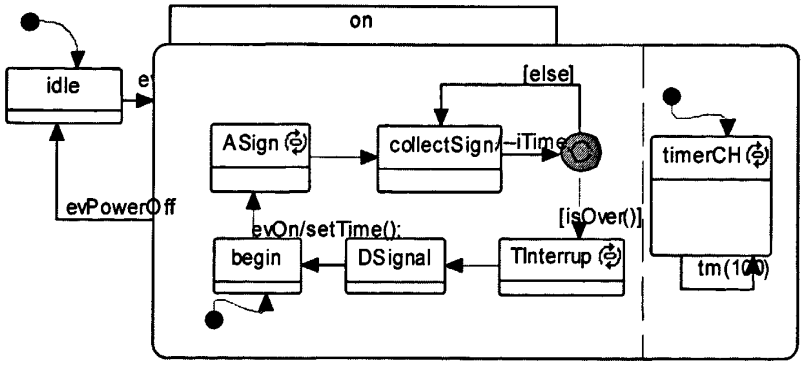


图 4. 19 模拟输入通道的状态图

模拟信号(ASignal)输入到模拟通道，信号采集(collectSignal)后，经中断处理(TInterrupt)后，转变为数字信号(DSignal)输出。

③建立串行通信模块的动态模型图

串行通信模块的行为功能主要由串行通道(Com)的行为功能体现。建立串行通道的状态图，

如图4.20所示。

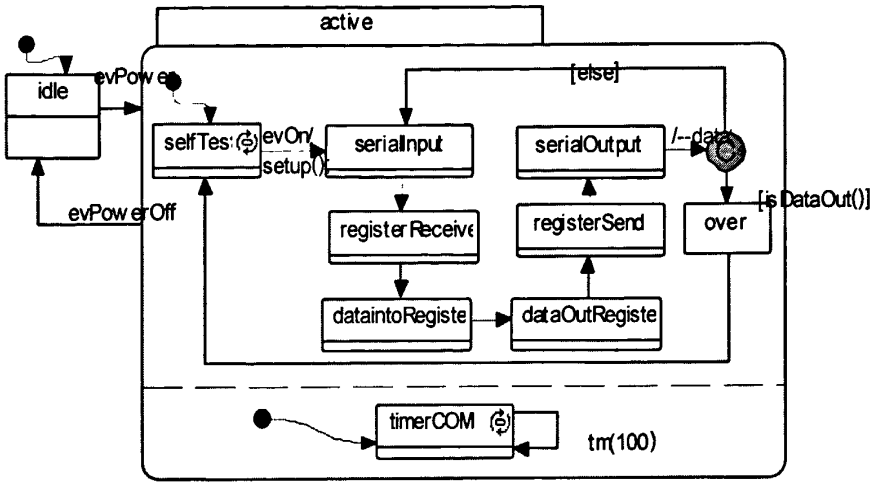


图 4. 20 串行通道的状态图

串行通道用于传感器、机载设备等的串行通信，这些设备的数据信号经串行输入(serialInput)，进入接收移位寄存器(registerReceive)，经接收移位寄存器进入数据输入寄存器(dataintoRegister)。数据输入寄存器中的数据是可供 CPU 读取的。处理后的数据进入数据输出寄存器(dataOutRegister)，经数据输出寄存器进入发送移位寄存器(registerSend)，由发送移位寄存器串行输出(registerSend)。

④建立定时计数模块的动态模型图

定时计数模块主要是提供定时计数功能，它的功能由定时计数器(TCDevice)实现，建立定时计数器的状态图，如图 4.21 所示。

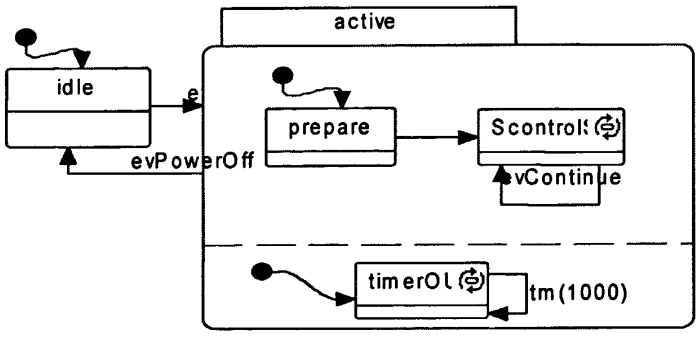
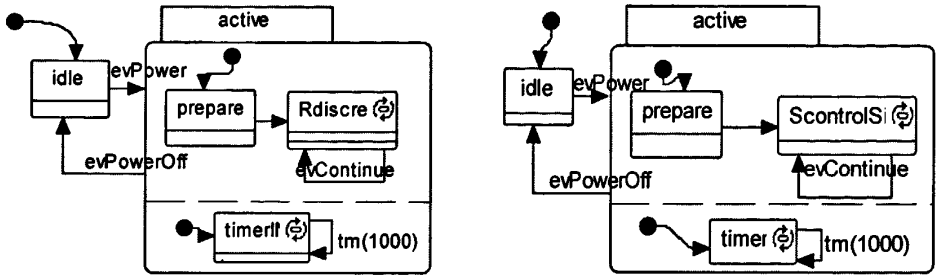


图 4. 21 定时计数器的状态图

⑤建立离散I/O模块的动态模型图

离散I/O模块主要起到输入离散状态信号，输出开关控制信号的作用，它的这部分功能可由离散输入通道(inChannel)和离散输出通道(outChannel)实现。建立离散输入通道和离散输出通道的状态图，如图4.22所示。离散输入通道主要功能是接收离散信号(RdiscreteSignal)；离散输出

通道主要功能是输出离散控制信号(ScontrolSignal)。



(a) 离散输入通道的状态图　　(b) 离散输出通道的状态图

图 4.22 离散通道的状态图

⑥建立前置信号处理模块的动态模型图

建立前置信号处理模块的状态图，如图4.23所示。

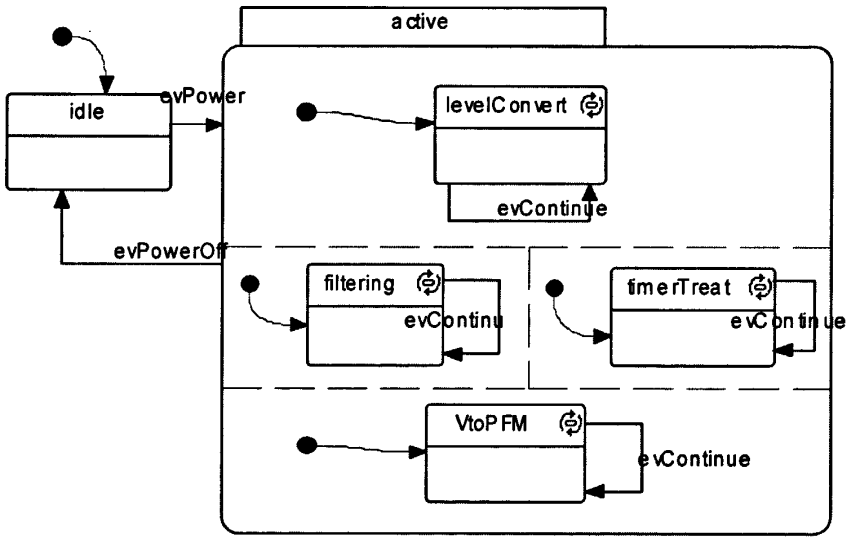


图 4.23 前置信号处理模块的状态图

模拟信号前置处理模块对模拟输入信号进行滤波(filtering)和电平转换(levelConvert)，将不同的模拟信号电压转换为A/D模块所要求的电压范围，以及将“电压信号转变为PFM信号”(VtoPFM)。

⑦建立电压转换模块的动态模型图

电压转换模块主要功能是实现电源转换功能：把机载电压转换成计算机、传感器和舵机等要求的不同数值的电压。如图4.24所示。在此图中，我们假设把机载27V直流电压转换为5V(DC27_5)，12V(DC27_12)，3V(DC27_3)。根据需要，我们还可以转换成其它值的电压。

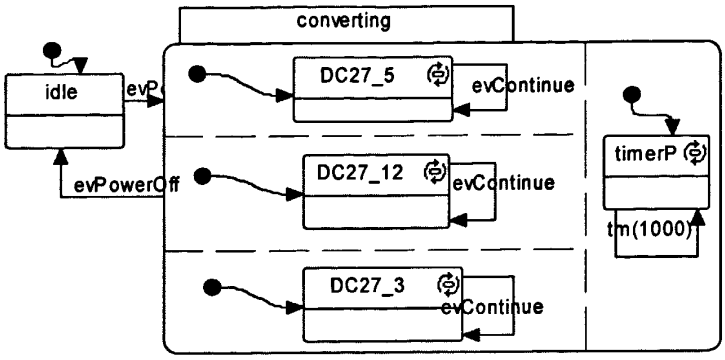


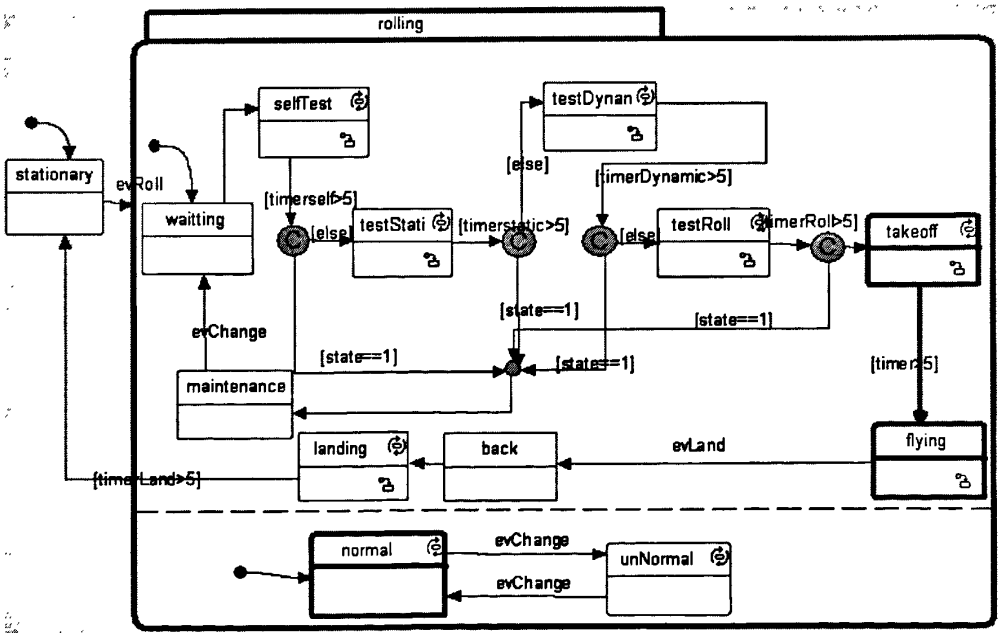
图 4.24 电源转换模块的状态图

4.3.4 模型的阶段性检测

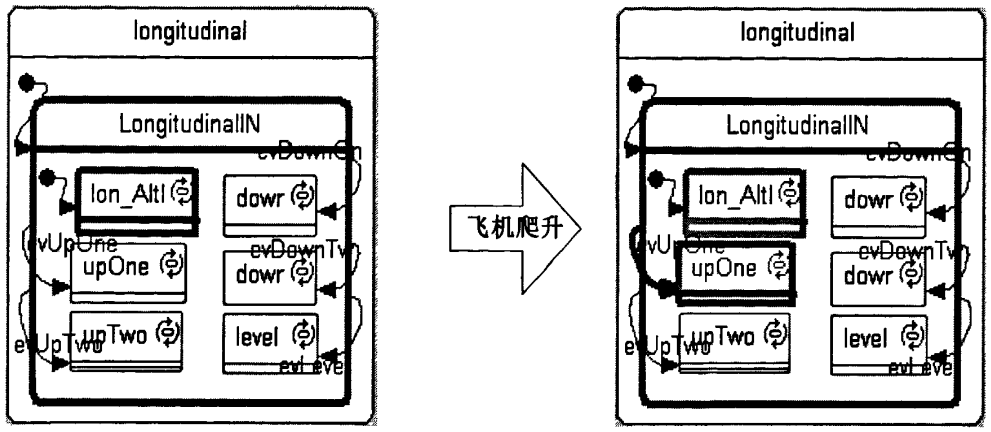
Rhapsody 软件一个最为显著的特点是可以边设计边仿真验证，而不用等到所有模型都建好。当设计完一个模型时，我们就可以在 Rhapsody 软件上设置只对该模型进行的仿真验证，一旦出现错误就可以及时的更正。采用“自底向上”的模式进行修改。

以图 4.15 所示的无人机的动态模型为例，对无人机的行为功能进行仿真验证。图 4.25 为其仿真验证图，图 4.26 为其 DOS 界面下的状态转变显示。

下图中，红色的粗边框表示当前无人机所处的状态。无人机进入飞行状态时，无人机的纵向和横侧向分别处于纵向保持(lon_AltHold)和横侧向保持(lat_AltHold)的状态，发送“向上1(upOne)”的纵向命令，改变无人机的运行姿态。



(a) 无人机数字仿真



(b) 无人机纵向的初始状态　　(c) 无人机向上爬升状态

图 4.25 无人机横侧向的初始状态

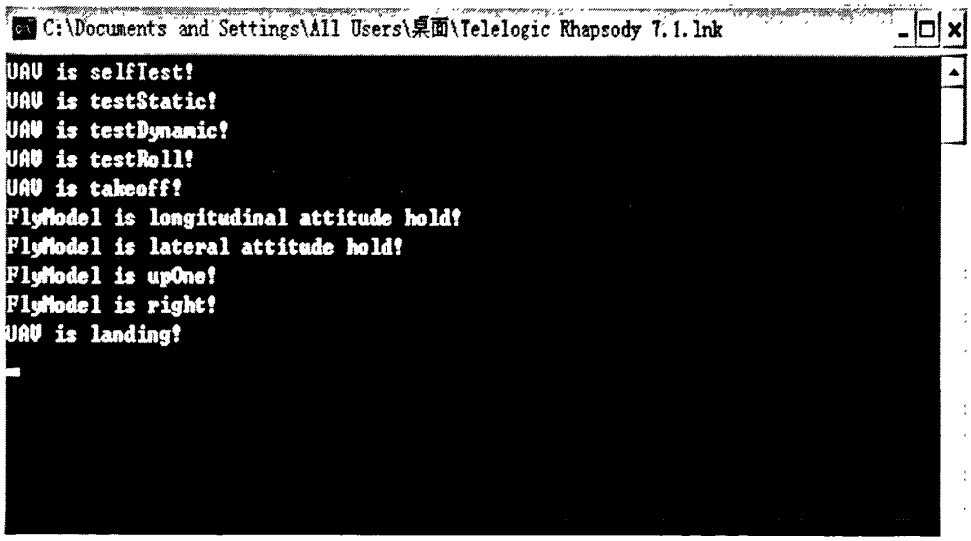


图 4.26 数字仿真的 DOS 界面显示

4.4 系统测试

在完成飞行控制系统数字化模型(即虚拟样机)的设计后，进行仿真验证，即进行系统的测试。系统数字化模型的一个共同点就是在虚拟样机平台的Rhapsody软件上完成， Rhapsody提供了三种不同的验证方式：1)利用GMR(G表示代码生成、M表示编译、R表示运行）方式检测模型的形式化语法和语义；2)基于人机交互界面的交互式仿真，验证系统行为和功能，完成对设计模型的一致性和完备性的检测；3)进行半物理仿真。在系统的阶段性测试阶段，分别采用前两种仿真验证方式对建立的飞行控制系统数字化模型进行仿真验证。

4.4.1 语法和语义的检测

Rhapsody软件本身不带有编译器，但它可以使用VC软件等其它软件的编译器对建立的模型的代码进行编译。由于代码和模型是对应的，对代码的编译检测，实际上就是对模型的编译检测。利用GMR方式对模型进行形式化语法和语义的检测，编译的结果会在Rhapsody软件上显示。如图4.27所示，没有错误，没有警告，语法和语义正确，编译成功。

```
Code generated to directory: E:\毕业课题\试验品\FCSTest第七版\guilib
Generating make file guilib.mak

Code Generation Done

0  Error(s), 0  Warning(s), 0  Message(s)
```

图 4. 27 系统虚拟样机的编译结果

4.4.2 基于人机交互界面的仿真

本文在虚拟样机平台的VC软件模块上设计了人机交互界面，用于模拟地面站操作界面。通过VC软件与Rhapsody的接口(如图4.13所示)，把人机交互界面导入到Rhapsody工程中，达到仿真过程中的交互式响应。

(1) 人机交互界面的设计准则

本文设计的人机交互界面需满足以下准则：

①操作性：界面直观、清晰、易操作；

②一致性：界面与现实世界尽可能在概念上、语言上、功能上保持一致；

③可辨性：界面的功能便于理解，文字清楚，无二义性；

④反馈性：对于用户的操作，立即给出反馈信息；

⑤健壮性：当用户操作错误，能够自动保护，并给予信息提示。

(2) 人机交互界面的设计依据

人机交互界面通常是根据设计阶段给出的对象类的属性、操作以及系统的模型，把系统交互的细节加入到VC界面的设计中，包括有效的人机交互所必须的实际屏幕显示、数据输入和指令发送。主要是强调如何操作系统，以及系统如何向使用者提交信息。图4.28为界面的交互机制图，其中GEN(evEvevt)是OXF框架中的函数；show()为VC软件中用C++语言定义的显示函数。当操纵人机交互界面改变无人机运行状态时，通过GEN(evEvent)函数响应，并通过该函数来启动statechart，进入相应的状态。进入相应状态后，调用show()函数，在人机交互界面上显示无人机改变后的状态。

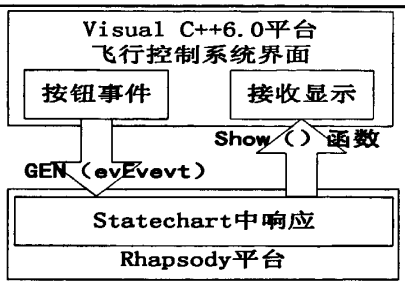


图 4.28 交互机制图

在 VC 软件模块上设计人机交互界面，设计的人机交互界面如图 4.29 所示。

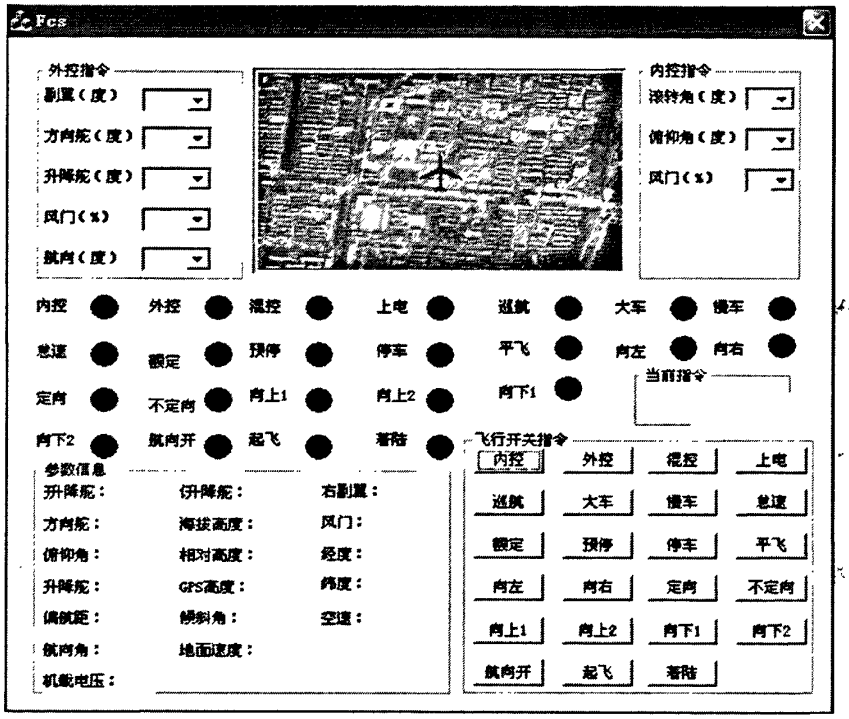


图 4.29 人机交互界面图

（3）基于人机交互界面的交互式仿真

运用基于人机交互界面的交互式仿真的方法对无人机飞行控制系统的数字化模型进行综合验证。 通过操纵VC界面，相应的statechart会响应，同时在VC界面上会显示当前发送的指令以及无人机的当前状态。如图4.30所示，当发送“内控”指令，让无人机处于内控飞行模态，在此模态下发送“向上2”指令。可以看到：在VC界面上会显示无人机的“内控”飞行模态、“向上2”的姿态(对应的指示灯由绿变红)。由于是爬升状态，对应的发动机会处于“大车”状态。对应的状态图如图4.31所示。

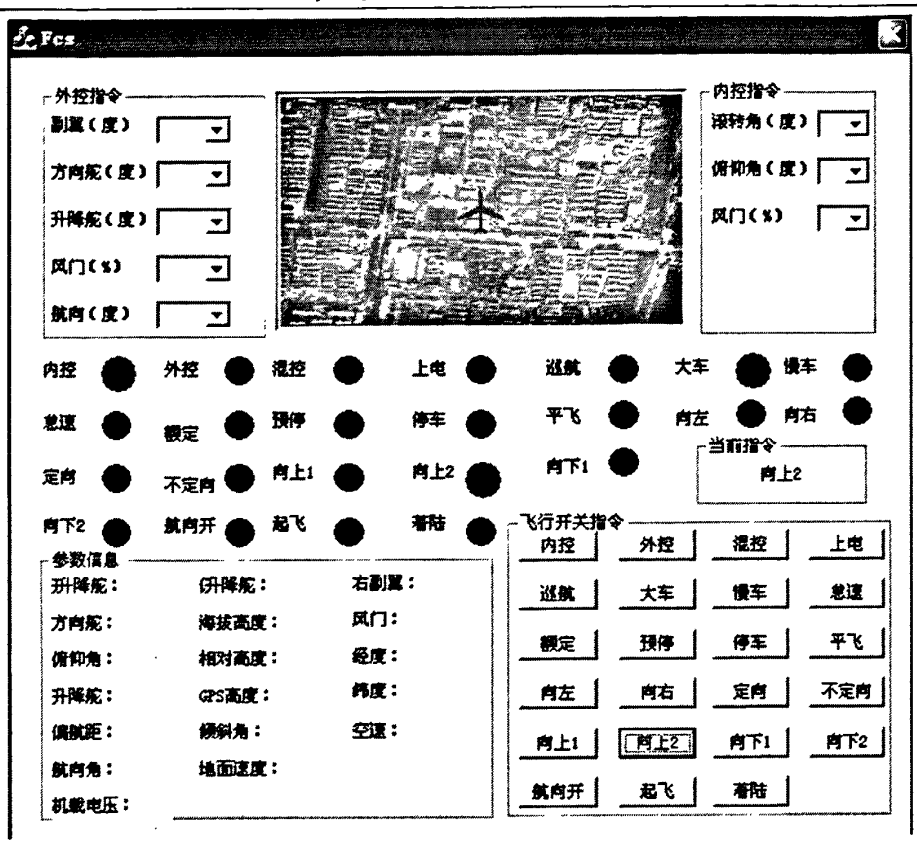
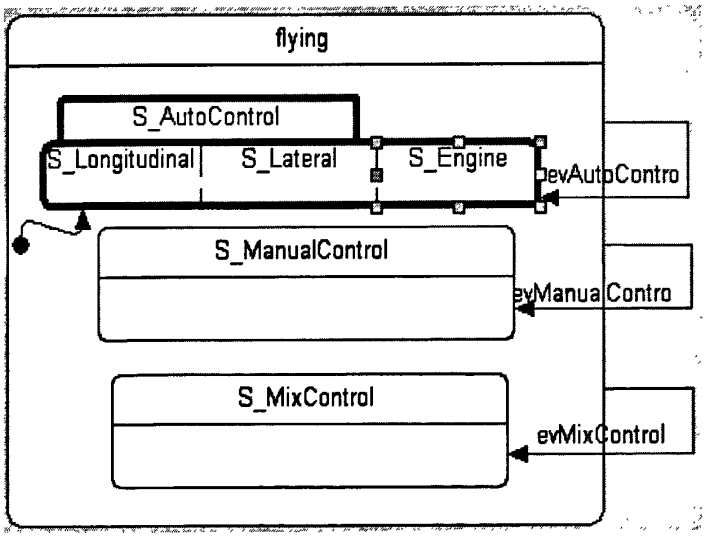
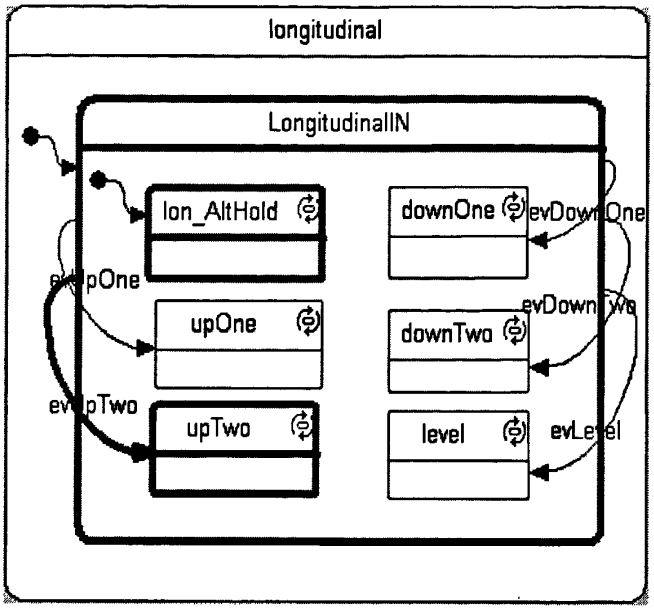


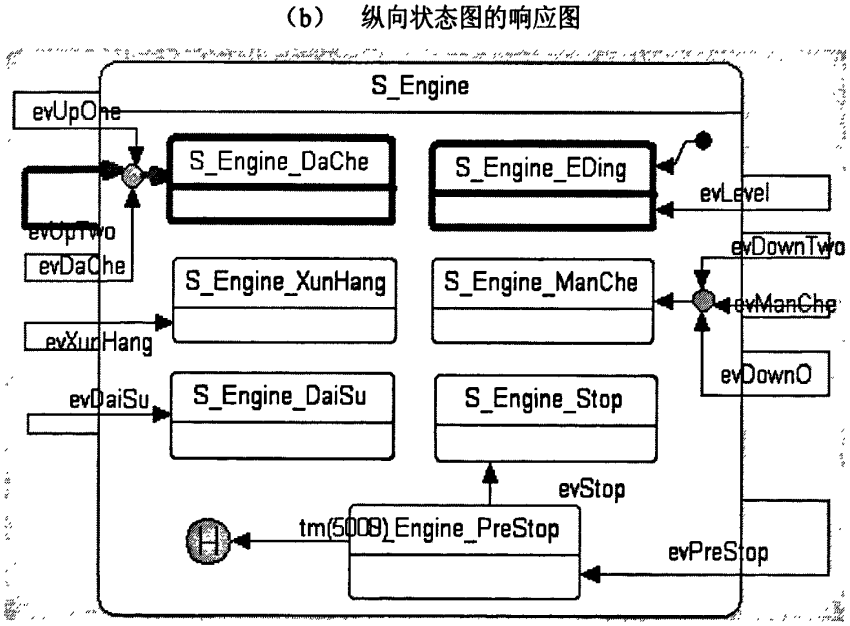
图 4.30 操纵 VC 交互界面



(a) 飞行模态的响应图



（b） 纵向状态图的响应图



（c） 发动机状态图的响应图

图 4. 31 statechart 响应图

4.5 代码编写

在完成系统的各个模型设计和测试后，下一步工作就是要编写系统产品级代码，即能下载到半物理仿真平台上运行的代码。

Rhapsody 的又一个最大优点就是能自动生成产品级代码。在完成系统各模型设计以后，我们在 Rhapsody 软件上自动生成模型的 C++产品级代码。所生成的代码位于系统的 Rhapsody 工程下，如图 4.32 所示，为软件自动生成的 C++代码的头文件和源文件。部分详细代码如图 4.33 所示。

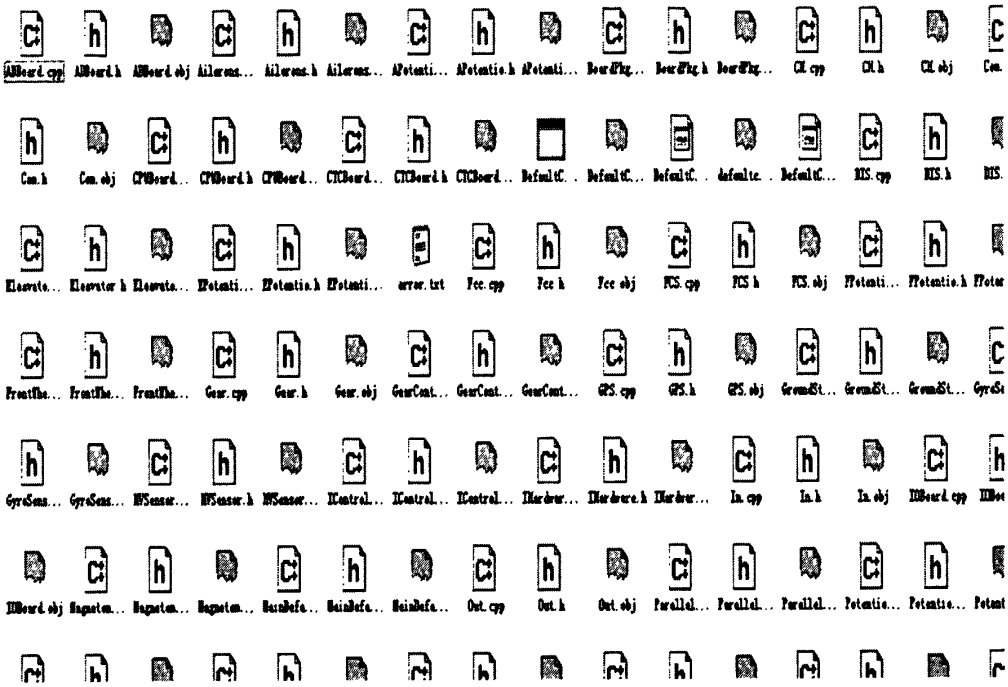


图 4.32 自动生成的 C++代码的头文件和源文件

```
/*********************************************************************
    Rhapsody     : 7.1
    Login        : Administrator
    Component    : DefaultComponent
    Configuration    : DefaultConfig
    Model Element    : UAV
//! Generated Date  : Mon, 3, Aug 2009
    File Path    : DefaultComponent\DefaultConfig\UAV.cpp
*********************************************************************
#define NAMESPACE_PREFIX
#define _OMSTATECHART_ANIMATED
#include "UAV.h"

#define UAVSystemPkg_UAV_UAV_SERIALIZE OM_NO_OP
#define UAVSystemPkg_UAV_rob_SERIALIZE OM_NO_OP
```

(a) 类 UAV 的源文件

```
/*********************************************************
    Rhapsody    : 7.1
    Login       : Administrator
    Component   : DefaultComponent
    Configuration   : DefaultConfig
    Model Element   : UAV
//! Generated Date  : Mon, 3, Aug 2009
    File Path   : DefaultComponent\DefaultConfig\UAV.h
*********************************************************/
#ifndef UAV_H
#define UAV_H

#include <oxf/oxf.h>
#include <aom/aom.h>
#include "UAVSystemPkg.h"
#include <oxf/omthread.h>
#include <oxf/omreactive.h>
#include <oxf/state.h>
#include <oxf/event.h>
// classInstance itsFCS
#include "FCS.h"
#include "IHardware.h"
//-----------------------------------------------------
// UAV.h
//-----------------------------------------------------
#ifdef _OMINSTRUMENT
class OMAnimatedUAV;
#endif // _OMINSTRUMENT
//## class UAV
class UAV : public OMReactive {
```

(b) 类 UAV 的头文件

图 4.33 自动生成的类 UAV 的代码

4.6 本章小结

通过把模型驱动方法与柔性开发模式相结合，本章得到用于系统设计的模型驱动式柔性开发方法。运用该方法，在第二章搭建的虚拟样机平台上完成了无人机飞行控制系统数字化模型的设计，其中系统的数字化模型包括需求模型、静态模型和动态模型。对所设计的模型，运用两种方式进行了阶段性测试。模型通过测试后，利用 Rhapsody 自动生成代码的功能生成产品级代码，为后续的半物理仿真奠定基础。

第五章 某型无人机动力学模型的导入

在完成了飞行控制系统的虚拟样机设计后，需要设计无人机的动力学模型。由于 UML 语言不适用于描述连续部分的模型，因此，无法使用 Rhapsody 软件建立无人机的动力学模型，需要在虚拟样机平台的 Simulink 软件模块上进行动力学模型的设计。参照文献[41]，完成某型无人机动力学模型的设计之后，通过 Rhapsody 软件与 Simulink 软件的接口，采用“黑盒子”方式把建立的动力学模型导入 Rhapsody 工程，并同时生成导入动力学模型后的飞行控制系统的代码。生成的系统的代码用于下载到系统的半物理仿真平台，进行系统的半物理仿真验证。

5.1 某型无人机动力学模型的设计

根据当前动力学模型的研究现状和本文的使用目的，本文所要设计的某型无人机的动力学模型为线性模型。在 Simulink 软件上设计飞行控制系统的动力学线性模型的方法已经非常成熟，而文献[41]便是按照这种经典的、成熟的方法设计了无人机的动力学线性模型。通过归纳总结，得到该方法的具体设计流程，如图 5.1 所示：

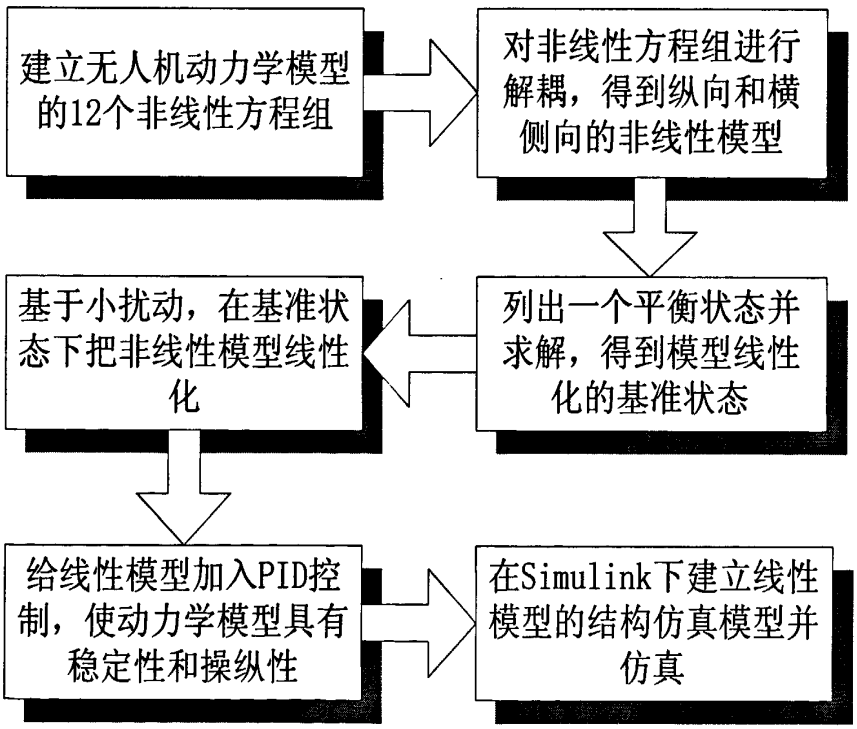


图 5.1 无人机动力学模型的详细设计流程图

由于本文是完全按照文献[41]提供的方法和数据建立无人机的动力学线性模型，只是简单

的重复，并无独特的创新之处，因此，在文中并没有详细给出在 Simulink 软件上建立无人机动力学模型的过程。

按照图 5.1 的详细步骤，建立无人机的动力学模型。通过对在机体坐标系中建立的六自由度非线性动力学模型的力矩方程组、运动学方程组和导航方程组等飞行控制系统动力学模型的 12 个方程[42]进行解耦，得到无人机动力学模型的纵向运动的非线性模型方程和横侧向运动的非线性模型方程。

无人机动力学模型的横侧向运动的非线性模型方程为：

$$\left.\begin{aligned}
&\dot{\beta}=Y/(mV_0)+\alpha_0 p-r\\
&\dot{p}-I_{xz}\dot{r}=L/I_x\\
&\dot{r}-I_{xz}\dot{p}=N/I_z\\
&\dot{\phi}=p+\theta_0 r\\
&\dot{\psi}=r
\end{aligned}\right\} \tag{5.1}$$

无人机动力学模型的纵向运动的非线性模型方程为：

$$\left.\begin{aligned}
&m\dot{V}=T\cos\alpha-D-G\sin\gamma\\
&mV\dot{\gamma}=T\sin\alpha+L-G\cos\gamma\\
&I_y\dot{q}=M+Tz_T\\
&\dot{\theta}=q\\
&\gamma=\theta-\alpha
\end{aligned}\right\} \tag{5.2}$$

通过求解一个平衡状态的解，得到一个基准状态。在基准状态的基础上，对非线性动力模型进行线性化，即对 5.1 式和 5.2 式进行线性化，得到无人机动力学模型的纵向线性化模型的方程和横侧向线性化模型的方程。由于运动方程组，力方程组和力矩方程组的计算并不依赖于偏航角ψ，因此在线性化处理时，并不需要对偏航角ψ进行处理，所以在对 5.1 式线性化时，直接忽略掉偏航角ψ。

无人机动力学模型的横侧向线性化模型的方程为

$$\left.\begin{aligned}
&mV_0\left(\frac{d\beta}{dt}+r\right)=Y_\beta\beta+Y_{\delta r}\delta_r+G\phi\\
&I_x\frac{dp}{dt}-I_{xz}\frac{dr}{dt}=L_\beta\beta+L_p p+L_r r+L_{\delta a}\delta_a\\
&I_z\frac{dr}{dt}-I_{xz}\frac{dp}{dt}=N_\beta\beta+N_p p+N_r r+N_{\delta a}\delta_a\\
&\frac{d\phi}{dt}=p
\end{aligned}\right\} \tag{5.3}$$

无人机动力学模型的纵向线性化模型的方程为

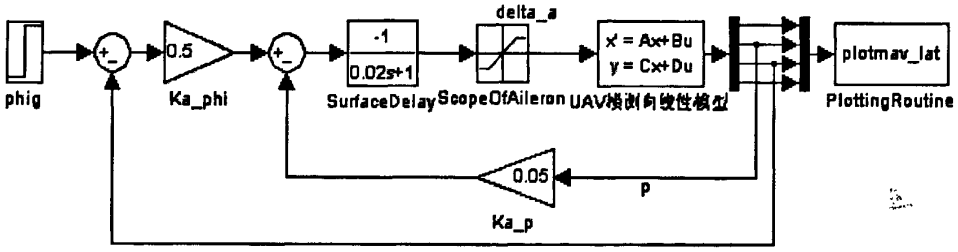
$$\begin{aligned}\dot{x} &= A\cdot x + B\cdot u \\ y &= C\cdot x + D\cdot u\end{aligned} \tag{5.4}$$

$$x = y = \left[\Delta\bar{V} \quad \Delta\alpha \quad \Delta q \quad \Delta\theta\right]^T, u = \left[\Delta\delta_e \quad \Delta\delta_T\right]^T$$

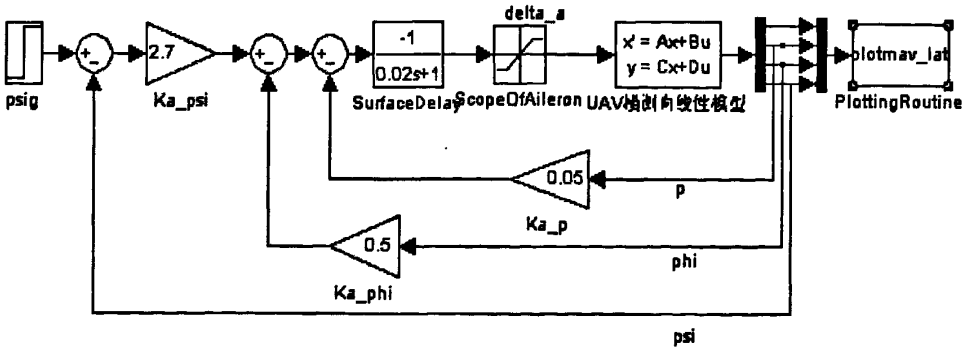
$$A = \begin{bmatrix} -X_V & -X_\alpha & 0 & -X_\theta \\ -Z_V & -Z_\alpha & 1 & 0 \\ M_{\dot{\alpha}}Z_V - M_V & M_{\dot{\alpha}}Z_\alpha - M_\alpha & -M_{\dot{\alpha}} - M_q & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & -X_{\delta T} \\ -Z_{\delta e} & 0 \\ M_{\dot{\alpha}}Z_{\delta e} - M_{\delta e} & -M_{\delta T} \\ 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

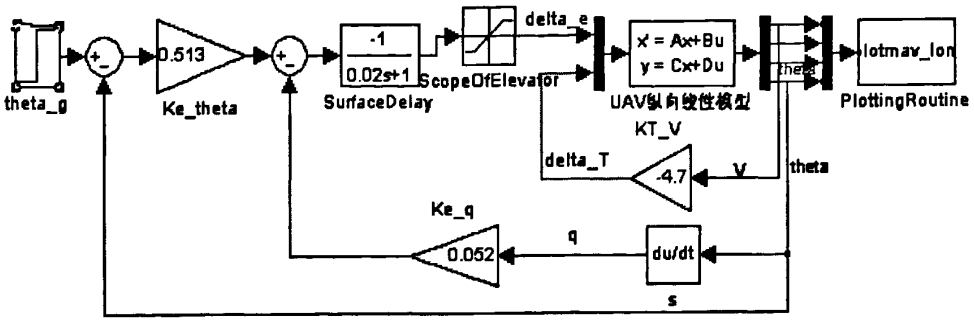
根据无人机动力学线性化模型的方程和文献[41]提供的数据，进行适当的 PID 控制律设计，得到了需要导入到 Rhapsody 工程的动力学模型的 Model 图，如图 5.2 所示。



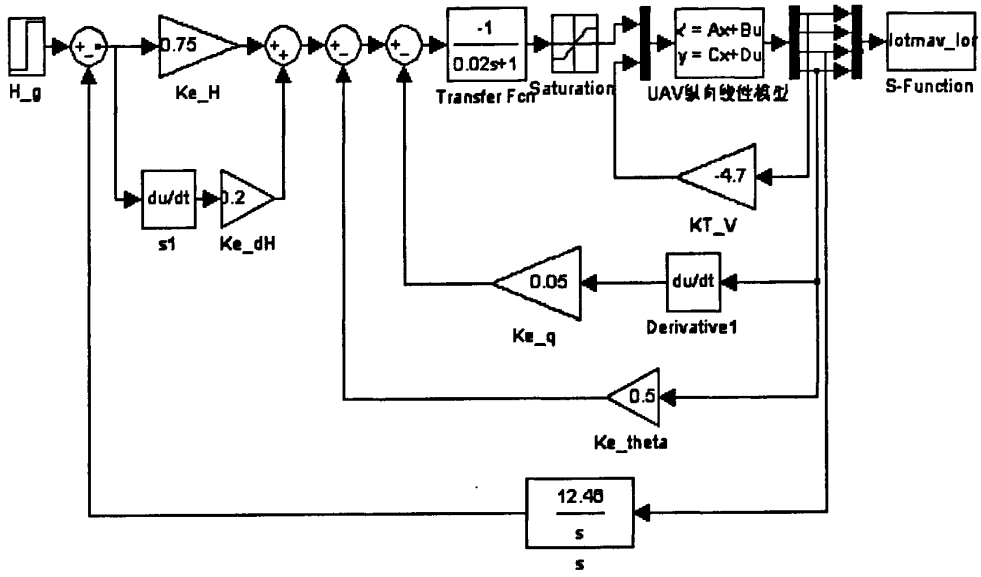
（a） 滚转姿态控制仿真模块



（b） 航向保持控制仿真模块



（c）速度控制时，俯仰姿态控制仿真模块



（d） 高度保持控制仿真模块

图 5.2 无人机动力学模型的模块图

在图 5.2 中，图 5.2（a）和图 5.2（b）所示的模块为无人机动力学模型的横侧向线性模型，图 5.2（c）和图 5.2（d）所示的模块为无人机动力学模型的纵向线性模型。图 5.2 中的 theta_g 为给定的俯仰角，H_g 为给定的高度， phig 为给定的滚转角，psig 为给定的偏航角。

图 5.2（a）和图 5.2（b）的线性模型表达式具体参数如下：

$$x=y=\left[\beta \quad p \quad r \quad \phi\right]^T, u=\delta_a$$

$$\begin{bmatrix}\dot{\beta}\\ \dot{p}\\ \dot{r}\\ \dot{\phi}\end{bmatrix}=\begin{bmatrix}-1.310 & 0 & -1 & 0.787\\ 121.553 & -0.527 & 0.057 & 0\\ 525.176 & -0.031 & -0.245 & 0\\ 0 & 1 & 0 & 0\end{bmatrix}\begin{bmatrix}\beta\\ p\\ r\\ \phi\end{bmatrix}+\begin{bmatrix}0\\ -530.694\\ -19.655\\ 0\end{bmatrix}\delta_a$$

$$\begin{bmatrix}\beta\\ p\\ r\\ \phi\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}\beta\\ p\\ r\\ \phi\end{bmatrix} \tag{5.5}$$

图 5.2（c）和图 5.2（d）的线性模型表达式具体参数如下：

$$x=y=\left[\Delta\bar{V}\quad \Delta\alpha\quad \Delta q\quad \Delta\theta\right]^T,u=\left[\Delta\delta_e\quad \Delta\delta_T\right]^T$$

$$\begin{bmatrix}\Delta\dot{\bar{V}}\\ \Delta\dot{\alpha}\\ \Delta\dot{q}\\ \Delta\dot{\theta}\end{bmatrix}=\begin{bmatrix}-0.388&-0.483&0&-0.787\\-1.608&-5.429&1&0\\0.261&-337.684&-0.005&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}\Delta\bar{V}\\ \Delta\alpha\\ \Delta q\\ \Delta\theta\end{bmatrix}+\begin{bmatrix}0&0.5\\-1.7&0\\-1062.7&0\\0&0\end{bmatrix}\begin{bmatrix}\Delta\delta_e\\ \Delta\delta_T\end{bmatrix}$$

$$\begin{bmatrix}\Delta\bar{V}\\ \Delta\alpha\\ \Delta q\\ \Delta\theta\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}\Delta\bar{V}\\ \Delta\alpha\\ \Delta q\\ \Delta\theta\end{bmatrix} \tag{5.6}$$

5.2 将动力学模型导入 Rhapsody 工程[43]

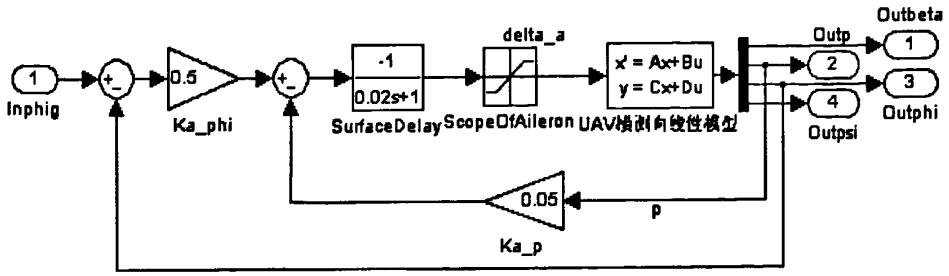
在 Simulink 软件模块上完成无人机动力学模型的设计之后，通过 Simulink 软件和 Rhapsody 软件的接口，将动力学模型导入 Rhapsody 工程中。

5.2.1 建立导入到 Rhapsody 工程的 Simulink 动力学模型

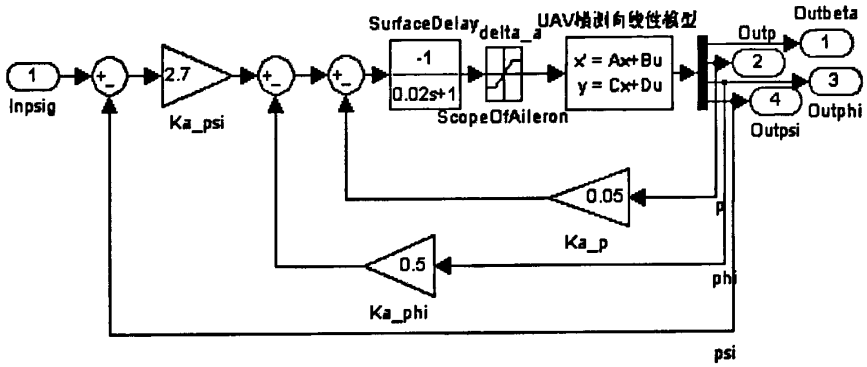
（1）对已经建好的动力学模型进行适当的修改，以便导入到 Rhapsody 工程。

需要被导入的四个模型分别为横侧向的两个模型和纵向的两个模型。横侧向的两个模型分别为：滚转姿态控制仿真模块，即图 5.2(a)所示的模块；航向保持控制仿真模块，即图 5.2(b)所示的模块。纵向的两个模型分别为：速度控制时，俯仰姿态控制仿真模块，即图 5.2(c)所示模块；高度保持控制仿真模块，即图 5.2(d)所示的模块。

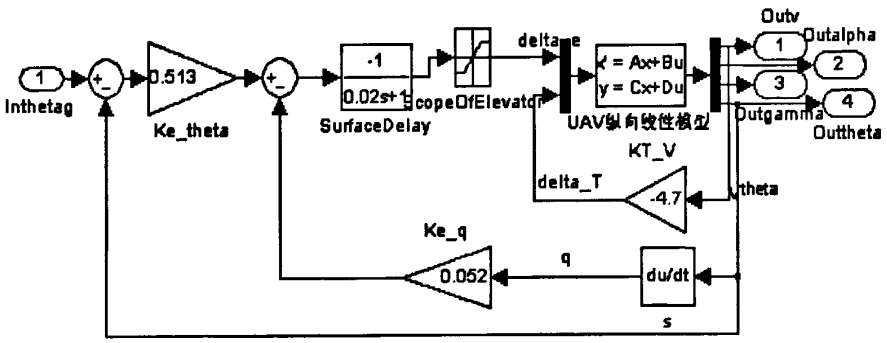
以上四个模块在 Simulink 软件上通过仿真验证，性能满足需求。因此采用上述四个模块作为无人机的动力学线性化模型，导入到 Rhapsody 工程。导入之前，需要对模型进行适当的修改，以便导入。修改后的四个模块如下。



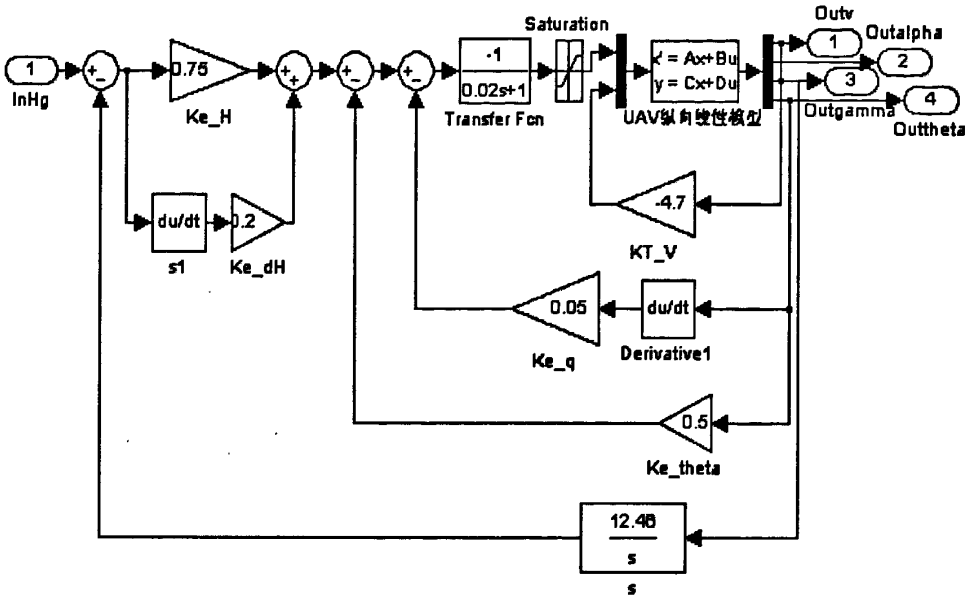
（a）滚转姿态控制的导入模块



（b）航向保持控制的导入模块



（c）速度控制时，俯仰姿态控制的导入模块



（d）高度保持控制的导入模块

图 5.3 导入 Rhapsody 工程的无人机动力学模型的模块图

图 5.3 中的输入、输出变量即为图 5.14 中的输入、输出变量。

（2）生成动力学模型的 C++代码

修改完上述四个模型后，分别在 Simulink 软件模块上生成 C++代码，生成 C++代码之前需要进行适当的设置。选择到 Tools>Real-Time Workshop>option 项，作如下设置：

① 把 Hard Implementation>Device Type Unspecified 设为 assume 32-bit Generic。

② 把 Real-Time Workshop>system target file 中的 grt.tle 改为 ert.tlc。

③ 把 Real-Time Workshop>Template makefile 中的 grt.tle 改为 ert.tlc。

④ 在 Real-Time Workshop>Language 中选择 C++选项(表示生成 C++代码)。

⑤ 把 Interface>support 的 Continuous time 选上。

设置完后生成对应模型的 C++代码。

5.2.2 将修改后的动力学模型导入 Rhapsody 工程

在 Simulink 软件上，完成系统的动力学模型设置后，生成 C++代码。把模块和代码文件一起拷贝到已经建好的 Rhapsody 工程中，通过 Rhapsody 和 Simulink 的接口，采用“黑盒子”模式把动力学模型的四个模块及其代码分别导入到 Rhapsody 工程，具体步骤如下：

（1）在 Rhapsody 软件上新建一个 FCS 工程。新建工程时，弹出一个对话框，在对话框的 Type 栏目选择 Simulink 项，表示可以向新建的工程中导入 Simulink 模型。注意：Simulink 选项与 Rhapsody 软件自动生成 C++代码相对应，SimulinkInc 选项与 Rhapsody 软件自动生成 C 代码

相对应。对话框图如下：

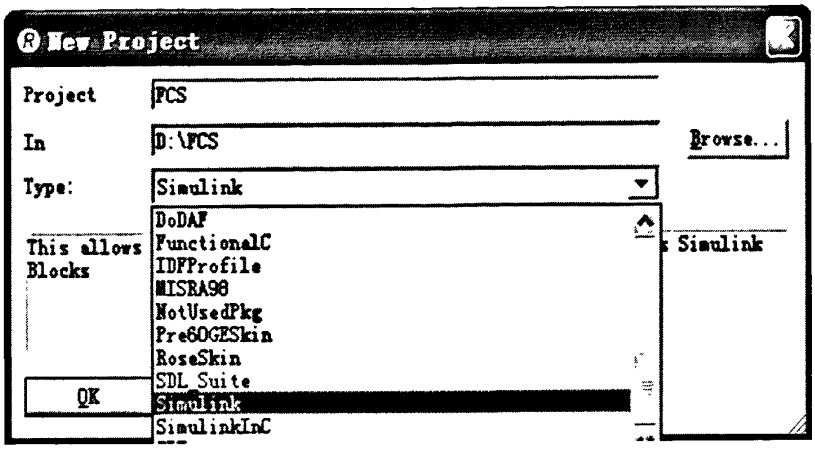


图 5.4 新建工程对话框

（2）在 FCS 工程中，按照第四章的方法，重新建立某型无人机飞行控制系统的数字化模型。

（3）把在 Simulink 软件中生成的 C++代码和模型拷贝到 FCS 工程的目录下。

（4）在 Rhapsody 工程中，新建一个对象模型图，在对象模型图中放入已经建好的 FCC 类和创建 Motor 类。如图 5.5 所示。Simulink 动力学模型通过 Motor 类导入。

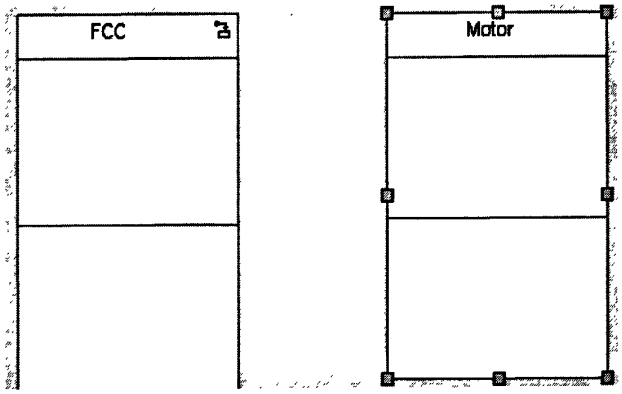


图 5.5 对象模型图

（5）通过 Motor 类，导入 Simulink 动力学模型到 Rhapsody 工程中，步骤如下：

①右击 Motor 类，选择 Features 项，弹出一个对话框，在对话框的 stereotype 栏目选择 SimulinkBlock in Simulink 选项，这样就可以把 Motor 类设置为能够导入 Simulink 模型的类，设置的对话框如图 5.6 所示。

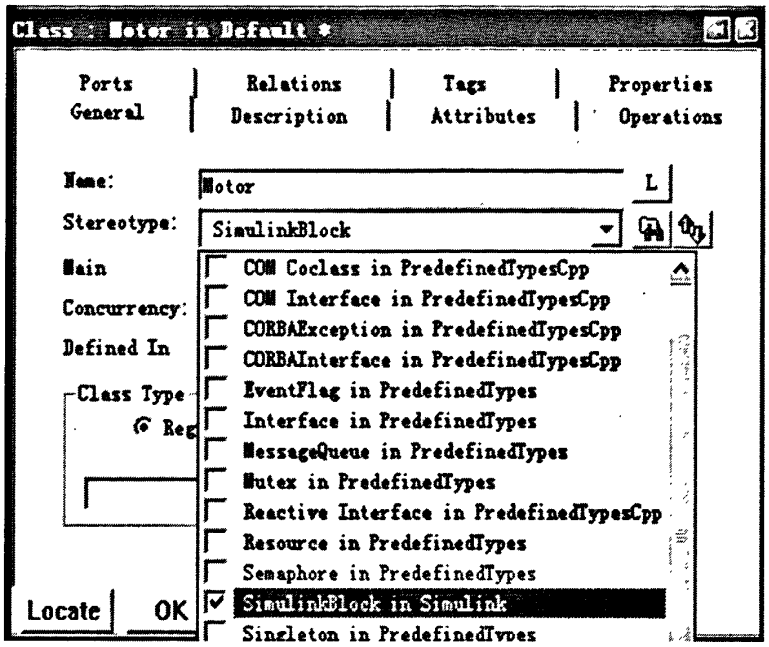


图 5.6 Features 对话框

选定之后，对应的 Motor 类变为图 5.7 所示的模型。

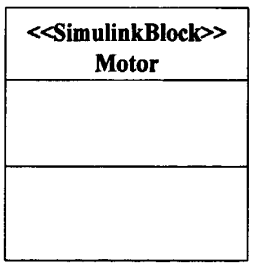


图 5.7 Motor 类的对象图

②右击 Motor 类，选择 Import/Sync Simulink Model 项(注意：如果不完成①步骤，右击时将不会有 Import/Sync Simulink Model 选项)，弹出以下对话框，用于导入模型和代码。

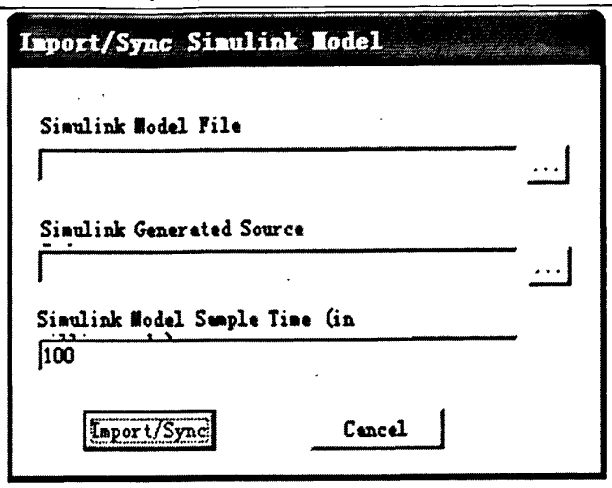


图 5.8 导入 Simulink 模型的对话框

③选择需要被导入的Simlink模型。在Simlink Model File栏目，选择需要被导入的模型(后缀名为. mdl 的文件)，即图 5.3 所示的四个模型。

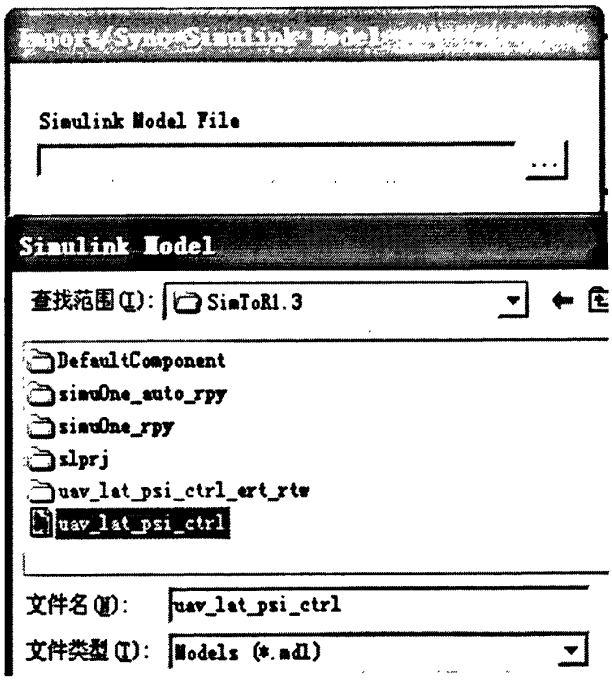


图 5.9 选择 Simulink 模块图导入的对话框

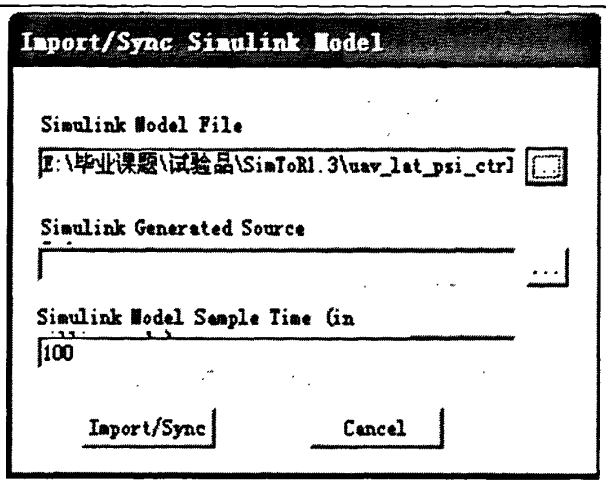


图 5.10 导入 Simulink 模块图后的对话框

④选择需要被导入的 C++代码。在 Simulink Generated Source 项，选择除 ert_main.cpp 外的所有源文件代码。显示的对话框如图 5.11 所示。图中的 Simulink Model Sample Time 为模型采样时间，单位为毫秒（ms），在默认情况下，设置的时间为 100ms，通常情况下，不需要更改时间。特殊情况下，根据实际需要进行修改。

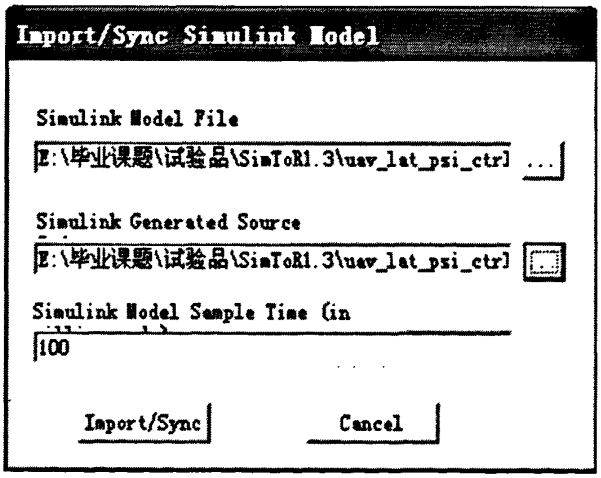


图 5.11 导入 Simulink 代码后的对话框

⑤导入模型和代码。模型和代码选择完毕后，点击对话框的 Import/Sync 按钮，就能把模型和代码导入 Rhapsody 工程，导入后的 Motor 类如图 5.12 所示。

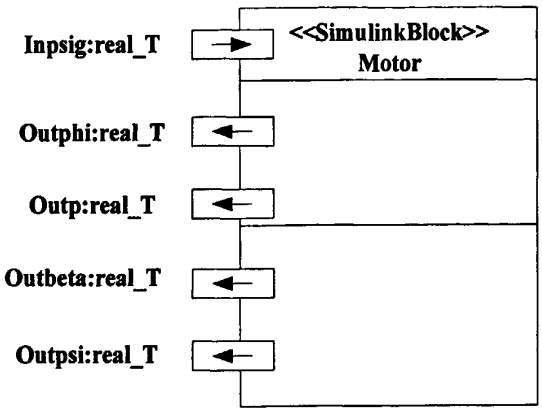


图 5. 12 导入 Simulink 模型后的对象模型图

⑥依据 Motor 类的接口，给 FCC 类添加相应的接口。

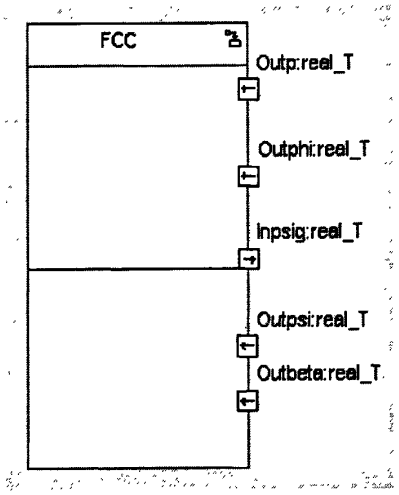
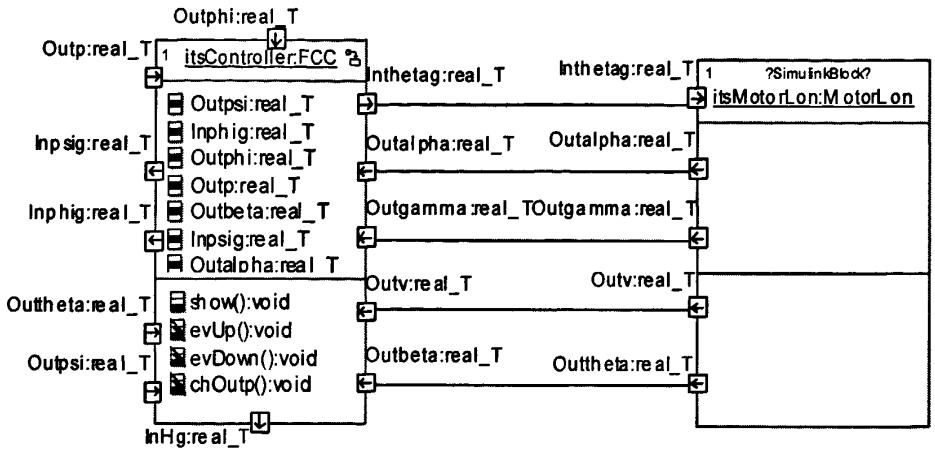


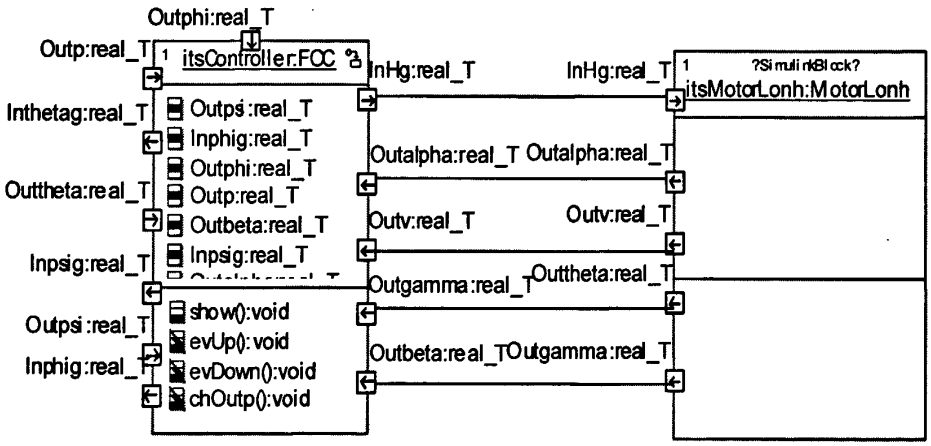
图 5. 13 更改后的 FCC 类的对象模型图

⑦把类实例化为对象。右击 FCC 类和 Motor 类，选择 Make an Object 项，把类转变为对象，再连接(类之间是不能连接的)，导入完毕。

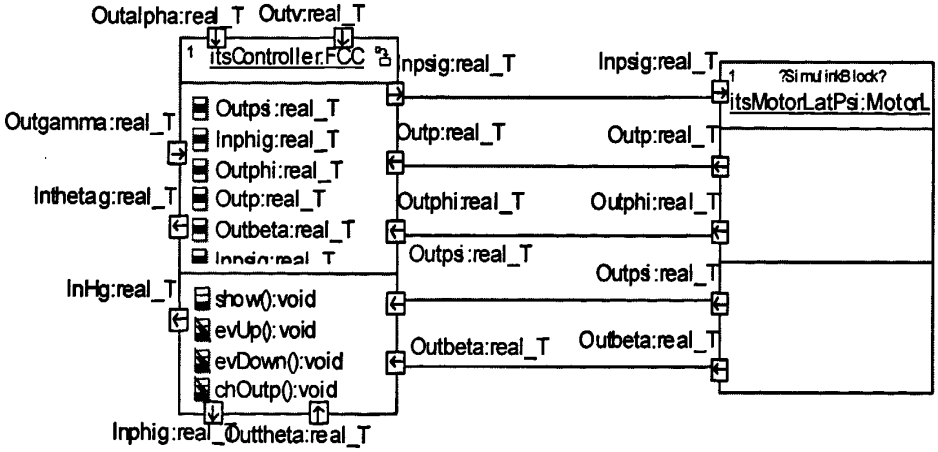
（6）依据此方法，依次导入的四个模型如下：



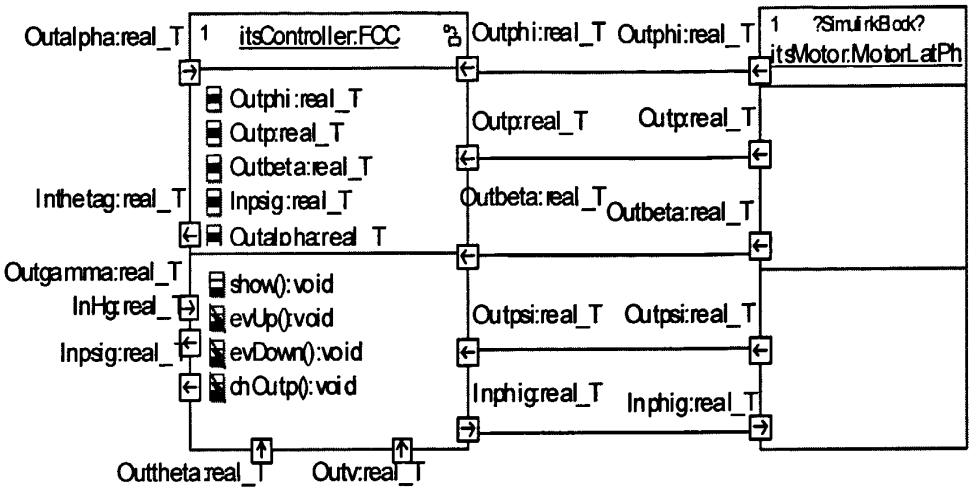
(a) 导入俯仰姿态控制仿真模块



(b) 导入高度保持控制仿真模块



(c) 导入航向保持控制仿真模块



(d) 导入滚转姿态控制仿真模块

图 5.14 导入 Simulink 动力学模型后的对象模型图

图中的 itsController:FCC 是与 Simulink 动力学模块交互的无人机飞行控制系统的飞控计算机对象，itsMotorLonh 和 itsMotorLontheta 是导入到 Rhapsody 工程中的 Simulink 动力学模型的纵向模型；itsMotorLatphi 和 itsMotorLatpsi 是导入到 Rhapsody 工程中的 Simulink 动力学模型的横侧向模型。其中 Inphig、Inthetag、Inpsig 和 InHg 等分别为滚转角、俯仰角、偏航角和高度等给定的控制输入向量。而 Outphi、Outpsi、Outbeta、Outalpha、Outgamma、Outp 和 Outtheta 等分别为滚转角、偏航角、侧滑角、迎角、偏航角速度、滚转角速度和俯仰角等输出状态向量。

5.3 系统虚拟样机的仿真验证

把无人机的动力学模型导入到Rhapsody工程之后，在Rhapsody软件上进行调试验证。通过以下两种方式进行验证。

5.3.1 语法和语义的检测

利用GMR方式对模型进行形式化语法和语义的检测，编译的结果如图5.15所示，没有错误，没有警告，语法和语义正确，编译成功。

```
Code generated to directory: E:\毕业课题\试验品\FCSTest第七版\guilib
Generating make file guilib.mak

Code Generation Done

0  Error(s), 0  Warning(s), 0  Message(s)
```

图 5.15 系统虚拟样机的编译结果

5.3.2 基于 DOS 界面的仿真

通过DOS界面可以观察数据是否正常发送。以航向保持控制仿真模块为例，输入变量Inphig给定为 1 度数值，对其进行调试验证，得到如图 5.15 所示的输出结果。输出状态向量按照设计的控制律正常的输出。

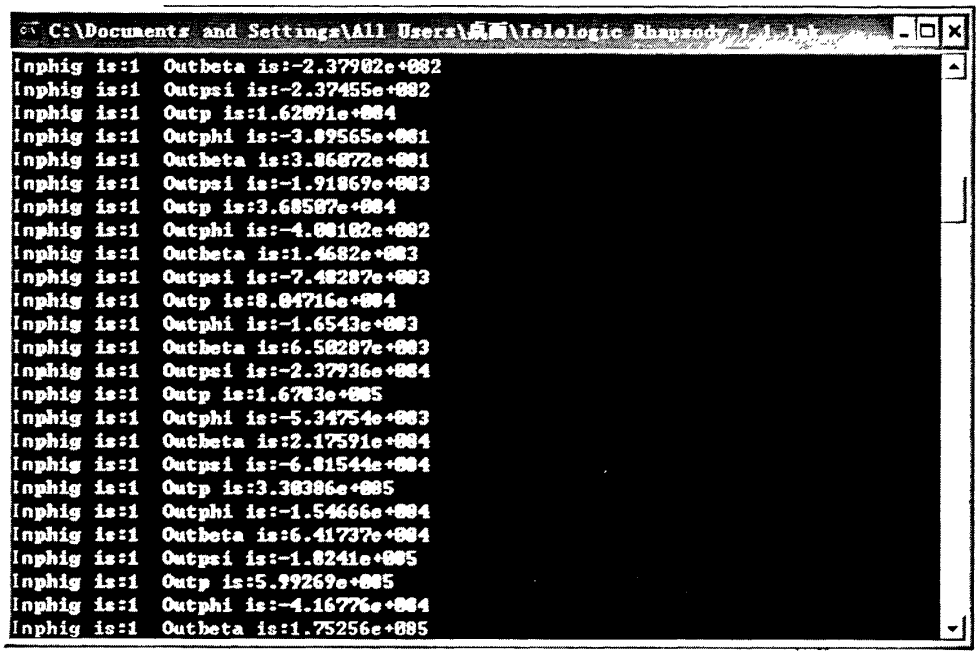


图 5. 16 航向保持控制仿真模块的输出

5.4 生成模型的代码

把动力学模型导入 Rhapsody 工程，调试通过后，利用 Rhapsody 软件自动生成代码的功能，生成带有动力学模型的飞行控制系统的 C++代码。该代码用于下载到硬件平台，进行半物理仿真试验。

5.5 本章小结

本章运用常用的动力学线性模型的设计方法和已有的数据，在 Simulink 软件上完成了某型无人机动力学线性模型的设计。通过 Rhapsody 软件与 Simulink 软件的接口，采用“黑盒子”方式把建立好的动力学模型导入 Rhapsody 工程，并同时生成导入动力学模型后的飞控系统的代码，生成的代码用于下载到系统的半物理仿真平台，进行系统的半物理仿真验证。

第六章 总结与展望

6.1 工作总结

本文以某型无人机飞行控制系统为重点研究对象，对其进行了 UML 建模和仿真，即设计了某型无人机飞行控制系统的虚拟样机。本文主要完成的工作如下：

（1）先在总体上提出了某型无人机飞行控制系统的设计方法：面向虚拟样机设计的数字化设计；再设计出某型无人机飞行控制系统的组成结构；然后提出了飞行控制系统所包含的功能模块；最后重点给出了各组成部分的具体功能和性能要求。

（2）设计出适用于面向虚拟样机的数字化设计方法的虚拟样机平台。

（3）对设计模式做了重点研究，包括基本设计模式的研究，将设计模式用于 UML 建模的优点以及如何将设计模式用于 UML 建模。在上述研究的基础上，选出合适的设计模式用于飞行控制系统 UML 建模。

（4）提出了用于系统虚拟样机设计的模型驱动式柔性开发方法，运用该方法和设计模式，在虚拟样机平台的 Rhapsody 软件模块上完成了无人机飞行控制系统虚拟样机的设计。

（5）在虚拟样机平台的 Simulink 软件模块上建立了无人机的动力学线性模型。动力学线性模型设计完成之后，通过 Rhapsody 和 Simulink 的接口，采用“黑盒子”模式，把建立的动力学线性模型导入 Rhapsody 工程，并同时生成导入动力学模型后的飞控系统的代码。

（6）仿真验证的研究。运用两种方式对飞行控制系统的虚拟样机和导入动力学模型的飞行控制系统模型进行了测试：①利用 GMR 方式进行语法和语义的检测；②基于人机交互界面的交互式仿真。该方法首先是在虚拟样机平台的 VC 软件模块上设计了人机交互界面，然后通过 VC 软件与 Rhapsody 的接口，把人机交互界面导入到 Rhapsody 工程，达到仿真过程中的交互式响应。除了上述两种仿真验证方式外，还可在设计过程中实现边设计边仿真验证。

（7）利用 Rhapsody 自动生成代码的功能生成系统的产品级代码。

无人机飞行控制系统的研究是一个长期的课题，本文虽然完成了某型无人机飞行控制系统虚拟样机的设计，但是，由于能力及时间有限，部分工作存在不足或有待进一步完善。不足或有待进一步完善的工作如下：

（1）在虚拟样机平台的 Rhapsody 软件模块上对无人机飞行控制系统进行建模，主要是依据无人机飞行控制系统的结构和功能要求，即模型是对系统结构和功能的映射，而没有准确反应出系统的性能要求。

（2）在虚拟样机平台的 Simulink 软件模块上建立的动力学模型是简单的动力学线性模型，

并不是实际动力学模型的完整写照。

（3）系统中的设计模式的应用还没有充分显现出设计模式的优越性能，系统的扩展或修改还需添加和修改部分代码，因此，对于代码的重用还没有达到一种非常理想的效果。

（4）在 Rhapsody 软件上生成的 C++代码是基于 OXF 框架，其稳定性还有待进一步的研究。

（5）搭建起系统的半物理仿真平台后，下载生成的代码到半物理仿真平台，进行半物理仿真实验。此部分已经完成了半物理仿真平台的总体设计和半物理仿真平台的总体构架。根据某型无人机飞行控制系统的要求，设计的半物理仿真平台的总体功能结构如下：

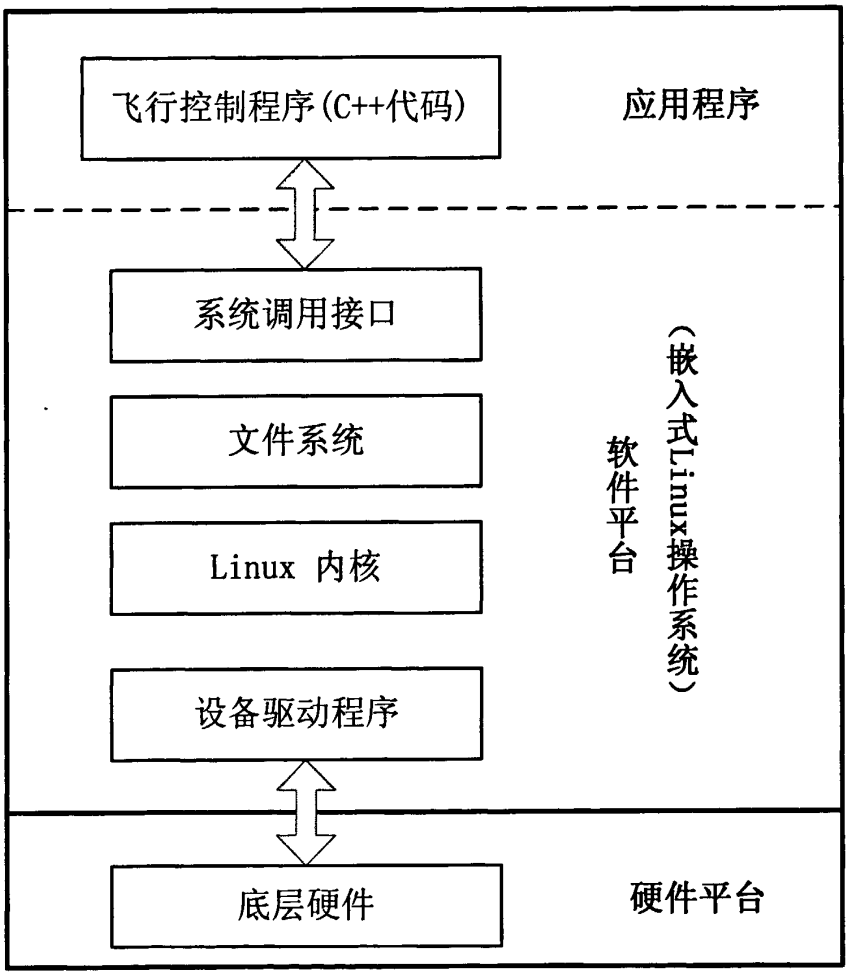


图 6.1 半物理仿真平台的总体功能结构图

飞行控制系统半物理仿真平台的总体构架如图 6.2 所示

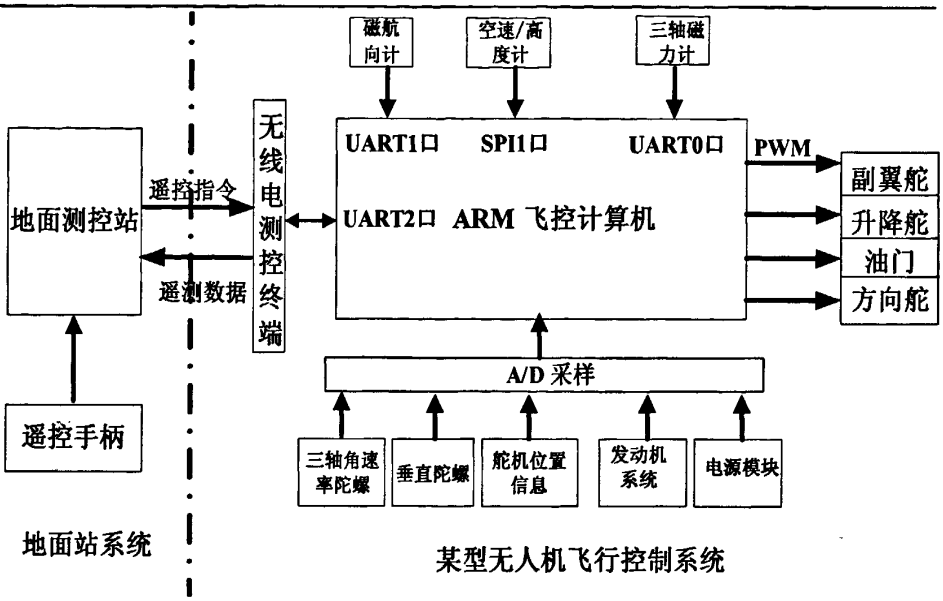


图 6.2 飞行控制系统硬件平台的总体结构

该部分的下一步工作就是依据飞行控制系统半物理仿真平台的总体构架，选择合适的部件，搭建起半物理仿真平台，把生成的代码下载到平台上进行半物理仿真实验。

6.2 前景展望

由于造价相对较低、用途广泛、适应性强和“零伤亡”的优势，无人机技术得到越来越广泛的重视，尤其是引起了各国政府的高度关注和参与。军用和民用的无人机存在着如下发展趋势[44-45]：（1）总体技术向小型化、智能化、隐身化的方向发展；（2）任务设备向全天候、高分辨率、远距离、实时化的方向发展；（3）测控、传输系统向安全保密、通用化、数字化、网络化的方向发展；（4）整个系统的使用向高生存率、低造价、低损耗的方向发展。

最后，希望本文的研究工作能够给后来的研究者一点启示，同时祝愿祖国的航空航天事业蒸蒸日上，蓬勃发展。

参考文献

[1] 曹云锋，苏丙未，沈春林．无人机飞行控制系统先进设计技术评述．飞机设计，2001，(4)：10～14.

[2] Karthik.K，Ward Donald．T.An intelligent flight director for autonomous aircraft．Texas A&M University，AIAA 200020168，2000.

[3] FCS—CAD 需求规格说明书[Z]，航空航天部618 所，1988.9.

[4] 陈宗基，黄浩东，秦旭东．飞行控制系统虚拟原型技术．航空学报，2002，23(5)：441～447.

[5] 刘兴华，曹云峰，沈春林．模型驱动的复杂反应式系统顶层设计与验证．系统仿真学报，2009，21(14)：4284～4287.

[6] 陈海东，沈　重，张　冶等．航天数字化应用技术的发展与趋势．导弹与航天运载技术，2008，(03)：23～27.

[7] 齐　欢，王小平．系统建模与仿真．北京：清华大学版社，2004：1.

[8] 刘心堂，吴晓燕．现代系统建模与仿真技术．西安：西北工业大学出版社，2001：8～10.

[9] 王红卫．建模与仿真．北京：科学出版社，2002：13～16.

[10] Hayward，David．Flight simulator．Ground Engineering，2003，36(11)：18～32.

[11] P.Carlo Cacciabue，Carlo Mauri，Douglas Owen．The development of a model and simulation of an aviation maintenance technician task performance．Cognition，Technology & Work，2003，5(04)：229～247.

[12] 童中翔，王晓东．飞行仿真技术的发展与展望．飞行力学，2002，20(03)：5～8.

[13] C++ Tutorial for Rhapsody [DB/OL]．[2007-4-15]http://www.telelogic.com/.

[14] 刁成嘉．UML系统建模与分析设计．北京：机械工业出版社，2007：31～34，63～64.

[15] 王　云，刘又诚，周伯生．UML 可视化建模系统的模型一致性检查机制．计算机研究与发展，2000，37(1)：1～7.

[16] 王　佩，杨秀梅．虚拟样机技术综述．新技术新工艺，2008，(03)：29～32.

[17] 熊光楞，李伯虎，柴旭东．虚拟样机技术．系统仿真学报，2001，13 (01)：114～117.

[18] 唐　硕，陈士橹，赵建卫．飞行器设计与试验的虚拟样机技术．宇航学报，2000，21 (S)：1～6.

[19] OXF Model [DB/OL]．[2005-9-14]http://www.telelogic.com/.

[20] 孙　强，张振华．使用 Rhapsody 软件框架和 UML 的实时系统开发．单片机与嵌入式系统应用，2003，(2)：22～25.

[21] Anneke Kleppe, Jos Warmer, Wim Bast. MDA Explained：The Practice and Promise of The Model Driven Architecture（鲍志云译）. 北京，人民邮电出版社，2004：17～31.

[22] Eran Gery, David Harel, Eldad Palachi. Rhapsody：A Complete Life-Cycle Model-Based Development System [A]. In : Integrated formal methods:IFM2002:proceeding of the 2002 International Conference, Turku,15-18 May，2002 [C].Berlin: Springer，c2002.

[23] 徐路宁，张和明，张永康. 基于虚拟样机环境下的复杂产品协同设计. 制造技术与机床，2005，(9)：69～72.

[24] 程翔宇，张红旗，王 璐等. 虚拟样机协同设计思想实施. 电子机械工程，2009，25（03）：61～64.

[25] 文传源. 现代飞行控制系统. 北京：北京航空航天大学出版社，1992：104～108.

[26] 张明廉. 飞行控制系统. 北京：航空工业出版社，1994：17～18.

[27] Eric Gamma、Richard Helm、Ralph Johnson，etal. Design patterns：elements of reusable object-oriented software（李英军译）. 北京：机械工业出版社，2007：389～623.

[28] 林 穗，李振坤，许建威. 基于构件的应用框架设计. 广东自动化与信息工程，2002，(4)：37～39.

[29] Alan Shalloway，James R.Trott. Design patterns explained：a new perspective on object-oriented design（徐言声译）. 北京：人民邮电出版社，2006：61，184～190.

[30] Bernd Bruegge,Allen H. Dutoit. Object-Oriented Software Engineering：Using UML、Patterns and Java（叶俊民，汪望珠译）. 北京:清华大学出版社,2006：243～261.

[31] Alexandre Sauve. Automating the Application of Design Patterns based on UML Models，[硕士学位论文]. Ottawa, Ontario, Canada：Carlcton Univcrsity，2005.

[32] 魏 功. 基于 UML 和设计模式的 Web 应用开发，[硕士学位论文]. 天津：天津大学，2005.

[33] 张洪涛，段发阶，王学影等. 基于 MVC 设计模式搭建嵌入式系统应用框架. 哈尔滨工业大学学报，2009，41（1）：166～168.

[34] 侯衍龙，基于 UML 面向对象建模技术及应用，[硕士学位论文]. 南京：南京航空航天大学， 2002.

[35] 王赟华，基于 UML 的模型驱动软件开发应用与研究，[硕士学位论文]. 北京：中国科学院研究生院，2005.

[36] 张 莉，葛 科等. UML 软件开发过程和支持环境研究. 北京航空航天大学学报，1998，24(4)：407～410.

[37] 谭云杰. 大象—Thinking in UML. 北京：中国水利水电出版社，2009：41.

[38] James Rumbaugh，Ivar Jacobson，Grady Booch. The Unified Modeling Language Reference

Manual(Second Edition). Beijing: Science Press, 2005: 2, 153.

[39] Rhapsody User Guide [DB/OL]. [2007-4-27]http://www.telelogic.com/.

[40] Banerjee A, Ray S, Dasgupta P, etal. A dynamic assertion-based verification platform for UML statecharts over rhapsody[C]//IEEE Region 10 Conference, Korea, 2008: 1～6.

[41] 曹美文. 基于 DSP 的 MAV 控制律设计与工程实现, [硕士学位论文]. 南京: 南京航空航天大学, 2006.

[42] 吴森堂, 费玉华. 飞行控制系统. 北京, 北京航空航天出版社, 2005.

[43] Using Third-party Tools [DB/OL]. [2007-4-15] http://www.telelogic.com/.

[44] 牛新元. LE110 无人直升机自主飞行控制研究, [硕士学位论文] . 南京: 南京航空航天大学, 2005.

[45] 梁秋憧, 程维明, 黄 欣. GPS 自主导航原理、算法及应用. 机床与液压, 2003, (04): 48～50.

致 谢

时光如梭，转眼间两年半的研究生学习生活即将落下帷幕。在这段宝贵的求学路途中，我受到了许多老师、学友和亲人的关心和帮助，没有他们的鼓励和支持，我的课题研究工作就不会顺利完成。在论文即将完成之际，我要向那些曾经给予我支持和帮助的人们表示深深的谢意。

首先要衷心的感谢我的指导老师曹云峰教授。感谢曹老师两年多来对我在学术上的悉心指导和生活上的关怀。曹老师深厚的学术修养、丰富的工程经验让我受益匪浅；同时，曹老师为我提供了良好的学习环境和其他有益于我学习的优越条件，使我可以顺利地完成相关的研究工作和论文的撰写，同时，本人的自身的能力也得到了很大的提高。

其次要衷心的感谢我的博士师兄刘兴华。自 2008 年夏季启动我的毕业课题研究工作以来，刘兴华师兄就给予我极大的帮助和关怀。正是在他的带领和帮助下，我的课题才得以稳步快速的向前推进。同时也要感谢庄丽葵老师、丁萌师兄的帮助和关怀。

再次是要感谢我的女朋友杨俊英。这些年来一直感谢她的陪伴。虽然我们身居两地，但她六年如一日的给予了我最珍贵的支持和鼓励，伴我走完了本科和研究生六年多的学习生活。

最后要感谢同门曲晓雷、谢娟、蔡晟、程尚、陈超、邹小志的帮助，大家在一起的相互交流和学习，使我有很大的进步。感谢赵滨，谭建鹏，徐慧，张寅生，胡建华，李航等实验室成员，与他们一起度过了令我难忘的研究生生活。

感谢舍友陈肖夏、丁健。谢谢他们在生活和学习上给予我的关心和帮助，与他们共处一室度过了最为珍贵的研究生学习生活。

再次向所有关心、支持和帮助我的老师、同学、亲人和朋友表示衷心的感谢！

最后，向在百忙中抽出时间对论文进行审阅和参加答辩会的各位评审老师致以诚挚的谢意。

何火军

2010 年 1 月

在学期间的研究成果及发表的学术论文

攻读硕士学位期间发表（录用）论文情况

1．HE Huo-Jun，ZHUANG Li-Kui，CAO Yun-Feng，LIU Xing-Hua．Digital Design for Embedded System On the Basis of Rhapsody．航天器工程．录用时间：2009.06.

2．何火军，曹云峰，刘兴华．无人机飞行控制系统虚拟样机平台．应用科学学报．2009，27（06）：637-643.

攻读硕士学位期间参加科研项目情况

1．在学期间参加的研究项目“综合系统顶层设计与数字仿真软件包”，2009 年 12 月结题