摘 要

原始数据生成仿真试验是导航系统仿真试验的重要组成部分。原始数据生成的模型体系涉及诸如天文、大气、通信等多学科领域，需要多部门的协同工作才能实现其仿真功能。由于模型和算法繁多，接口和数据流关系复杂，传统的建模仿真方法难以适应原始数据生成仿真的要求。仿真模型可移植性(Simulation Model Portability, SMP)具有模型接口标准化、简单化、模型的运行机制与模型分离等特点，特别适合包括原始数据生成在内的航天器仿真。本文根据 SMP 的第二版本 SMP2 的各种集成方法的特点，建立了基于 SMP2 的原始数据生成模型框。最后针对该模型框架，讨论了试验调度方法和仿真试验分析。

论文以导航系统的原始数据生成仿真为研究背景，围绕原始数据生成的模型框架展开研究。具体研究内容和技术创新包括：

(1)讨论了 SMP2 的建模仿真机制，重点研究了基于 SMP2 的仿真模型集成方法；

(2)参考 GSSF 以及有关文献，提出了一个初步的原始数据生成模型体系，对体系内的模型和算法进行实体模型分析、数据流分析和面向对象分析；

(3)依据 SMP2 的建模机制，建立了基于 SMP2 的原始数据生成模型框架，并对该框架的扩展性进行了研究；

(4)针对该模型框架，依据 SMP2 的仿真运行机制，提出了面向原始数据生成的仿真调度框架，并进行了基于精密星历生成轨道的原始数据生成仿真试验。

主题词：导航系统 原始数据生成 模型驱动架构 模型集成 仿真模型可移植性规范 模型框架

ABSTRACT

The simulation of raw data generation (RDG) is an important part of the simulation of navigation system. Raw data generation involves multidisciplinary, such as astronomy, atmosphere and communication, so it is necessaryfor for the people from different domain to work together. The largeness of models and algorithms, and the complexity of interface and data flow characterize the simulation of raw data generation.Conventional technology of modelling and simulation can hardly encounter the difficulty. Simulation Model Portability (SMP) which aims to improve the portability and reuse of simulation model, standardize and simplify the interface of simulation model, and separate the implement from the simulation models. It is specially suit for the simulation of spacecraft. This paper discuse the mechanism of modelling and simulation, and take an emphasis on the study on the integrating mechanism, then propose the method of establishing the model framework based on SMP2.

The thesis takes the simulation of raw data generation of navigation system as the research background, and focuses on the research the model framework of raw data generation. The contents and innovations are summarized into the following parts:

(1)Discuss the mechanism of modelling and simulation based on SMP2, and study its integrating mechanism.

(2)Propose an original model system of raw data generation, study the modes system through entity analysis, data flow analysis and object oriented analysis;

(3)Establish a model framework of raw data generation based on SMP2, and discuss the assembly and schedule of its simulation models.

(4)Discuss the schedule framework and partly validate the framework through simulation experimentation.

Key Words: Navigation System, RDG, MDA, Model Integration, SMP, Model Framework

表 目 录

图 目 录

独 创 性 声 明

本人声明所呈交的学位论文是我本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，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和撰写过的研究成果，也不包含为获得国防科学技术大学或其它教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。

学位论文题目：基于 SMP2 的导航系统原始数据生成模型框架研究

学位论文作者签名：阳振坤　　日期：2007 年 12 月 10 日

学位论文版权使用授权书

本人完全了解国防科学技术大学有关保留、使用学位论文的规定。本人授权国防科学技术大学可以保留并向国家有关部门或机构送交论文的复印件和电子文档，允许论文被查阅和借阅；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。

（保密学位论文在解密后适用本授权书。）

学位论文题目：基于 SMP2 的导航系统原始数据生成模型框架研究

学位论文作者签名：阳振坤　　日期：2007 年 12 月 10 日

作者指导教师签名：李群　　日期：2007 年 12 月 10 日

第一章 绪论

1.1 研究背景

卫星导航定位系统广泛用于国民经济、科学研究、国防建设以及军事斗争。当今世界，卫星导航定位系统已是一个国家综合实力以及科学技术发展的重要标志，也是国家安全必不可少的重要保障。卫星导航定位系统是一种以卫星为基础的无线电导航系统，系统可以发送高精度、全日时、全天候的导航定位和授时信息，是一种可供海陆空领域的军民用户共享的信息资源。世界各主要大国纷纷投入巨资，实施其导航卫星系统计划。目前全球已投入运行的卫星导航定位系统主要有美国的 GPS 系统、俄罗斯的 GLONASS 系统以及我国的北斗双星系统，备受瞩目的欧洲 Galileo 系统也进入了实施阶段[1]。

在这些卫星导航系统的设计和实施过程中，人们越来越深刻的认识到系统仿真技术对于卫星导航系统建设的重要性。GPS 从概念提出到系统完全建成，前后历时二十余年，期间对定位的基本概念、基本原理、基本模式、系统算法、系统控制等都作了大量的工程计算与试验。但由于 GPS 概念提出是在上世纪七十年代，各种条件尤其是计算机技术极大地限制了系统分析工作的开展，主要分析和验证工作均采用卫星在轨演示验证完成，造成系统建设耗资巨大，周期长达二十多年。近十年来，由于计算机技术、网络技术、图形图像技术、多媒体技术、软件工程、信息处理、自动控制等多项高新技术的发展，加快了系统仿真技术研究的步伐，仿真技术也越来越深入到航天领域的研究。卫星导航系统主要包括空间部分、运行控制部分和用户终端部分，系统各个组成部分之间及其内部接口关系异常复杂，涉及多学科专业领域知识，是一项规模庞大结构复杂的系统工程。同时，从 GPS 系统、GLONASS 系统和目前的 Galileo 系统的建设经验来看，研制仿真系统对导航系统的设计与建设具有重要的校核和验证作用。因此，在我国的卫星导航系统建设过程中，有必要利用现代计算机仿真技术和试验技术，建立能够较真实地反映卫星导航系统工作原理及运行机制的系统模型，集成现有的各种卫星导航系统分析支持工具与设备，构建仿真验证与测试评定系统，以支持卫星导航系统方案主要技术指标的深化论证、系统关键技术分析与验证、系统间对接试验测试与评定、系统性能指标测试与评估、卫星导航系统长期发展论证等技术研究工作的，为我国卫星导航建设提供有效的技术支持。

在 Galileo 系统仿真工具(Galileo System Simulation Facility,GSSF[2-4])报告中明确提出了原始数据生成(Raw Data Generation, RDG)的概念。原始数据生成仿真，即使用高精度模型，包括信号传播过程中可能出现的关键事件，来生成导航系统地面监测站对可见卫星的观测数据(包括伪距、载波相位、多普勒频移和信噪比)仿真值，这些观测数据的仿真值用于轨道与同步处理设施(Orbitography and

Synchronization Processing Facility,OSPF)和完好性处理设施(Integrity Processing Facility,IPF)算法的确认和调整，也用于链路预算分析和误差预算分析，其功能组成如图 1.1 所示[5]。

在这里，链路预算分析是指在每一个历元的所有地面站(地面传感器站，以下同)与其可见卫星之间对下列参数进行分析：导航信号的发射功率、接收功率、发射增益、接收增益、自由空间损耗、环境段损耗、指向损耗、信噪比、干扰损耗和仰角。这些预算指标主要用于分析信号传播过程中的功率电平损耗来源及其大小。

误差预算分析是指在每一个历元的所有地面站与其可见卫星之间对下列参数进行分析：伪距、电离层延迟、对流层延迟、自由空间延迟、多径延迟、总的延迟、仰角和星地几何距离。这些预算指标主要用于分析导航定位的误差来源及其大小。

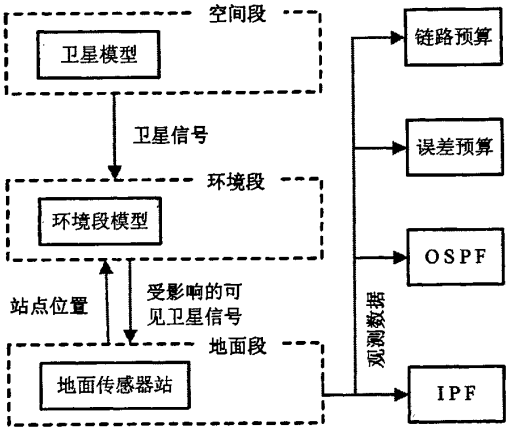

图 1.1 原始数据生成的功能

轨道与同步处理设施[6,7]，作为 Galileo 控制中心的重要组成部分，主要负责测定卫星导航数据产品，通过空间信号(Signal-In-Space, SIS)发布给导航系统用户。导航数据产品包括：数据获取及预处理、定轨及时间预处理、星历计算、星钟预报、长期空间信号精度测定等。

完好性处理设施[7,8]，也是控制中心的组成部分之一，主要负责测定卫星完好性信息，实时将其发布给用户。这些完好性信息主要包括完好性标志和空间信号监测精度(Signal In Space Monitoring Accuracy, SISMA)。它需要完成以下步骤：获取数据及预处理；估计实时信号误差；将计算的信号误差(Signal In Space Accuracy, SISE)与来自 OSPF 的信号精度(SISA)进行对比；生成恰当完好性状态表以及当 SISE 与 SISA 不符时向用户报警。

以上所述的链路预算、误差预算、轨道与同步处理设施、完好性处理设施，在导航系统的方案论证中具有重要的价值[9]。然而在系统的设计、论证和验证过程中往往缺乏真实的观测数据。因此，有必要通过仿真试验生成这些观测数据，这

也就是我们所要研究的原始数据生成。在卫星导航系统建设过程中，研究原始数据生成，对于系统方案和技术指标的仿真验证、系统性能评估、系统优化设计以及系统操作人员的培训等方面都有很重要的使用价值。

原始数据生成的仿真试验，可以简单的表述为将现代建模仿真技术与原始数据生成的模型相结合，通过对其数学模型的仿真试验生成观测数据的仿真值。在进行导航系统的试验分析和方案论证中，可以用这些观测数据仿真值代替真实导航系统的观测数据进行仿真试验。原始数据生成仿真试验产生的数据可用于已有导航系统的性能分析、方案验证等方面，也可用于新一代导航系统的虚拟设计和性能研究。从上个世纪 90 年代初世界上第一个卫星导航系统 GPS 建立运行以来，从导航系统本身的理论和技术，到各个专业领域的研究都取得了巨大进步，理论和技术的更新发展迅速。原始数据生成的模型主要分成空间段模型、环境段模型和地面段模型，其仿真试验具有以下特点：

(1)原始数据生成的模型涉及天文、大气、地理和通信等多学科领域，仿真试验需要这些学科领域不同部门的协同研究。各专业领域的研究内容和分析方法往往局限于各自的专业，相互之间协作困难。

(2)各部门虽然理解本专业模型的理论基础和所使用的仿真算法，但可能对仿真建模及编写符合规范要求的程序不是很熟悉。

(3)不同学科领域不同部门定制的仿真算法以及实现这些算法的程序结构、编程语言等往往存在很大的差别。

(4)不同模型算法对计算资源需求和计算时间相差很大，给仿真运行的时间管理带来难度，必须合理分配计算资源，适应这种差别，并在建模和仿真运行过程中考虑到仿真时间的推进管理。

(5)模型和算法数量繁多，接口关系复杂，需要有良好的规范定义模型和算法的类型、属性以及接口关系。

(6)对同一个模型往往有多个可替代算法，这些不同的算法又分别适用于不同的仿真试验。

(7)模型间的数据交互复杂，仿真试验需要有良好的系统工程数据管理功能。

(8)导航系统以及相关各学科领域研究发展迅速，仿真试验应适应相关领域研究的可持续发展，可以不断的将最新理论和技术应用于仿真试验。

(9)导航系统的仿真研究日趋广泛，不同的仿真试验往往需要相同的仿真模型，为节约开发成本，缩短开发周期，仿真模型开发应当支持可重用。

(10)由于某些仿真试验的特殊需求，仿真模型可能面临着在不同仿真环境或者不同开发平台之间的可移植性问题。

综上所述，开发原始数据生成的仿真模型，并有效集成起来实现其仿真试验的功能将是一项复杂的系统工程。我们迫切需要一种有效的建模仿真方法，能够克服上述困难，这种建模仿真方法应该具备以下基本特点：支持建模人员建立合

理的模型框架，并且该框架可以根据试验的需要进行扩展；框架下开发的仿真模型，应当接口定义清晰、数据管理方便、易于实现模型重用；仿真模型应当支持各专业领域模型的封装，各专业模型的开发人员只需按照仿真模型定义的接口规范开发程序模块，而无需理解整个模型框架以及与该程序模块相关的其它领域模型接口和数据流关系；支持仿真试验人员对仿真时间推进的管理；支持仿真模型和仿真系统在不同仿真环境和不同平台之间的可移植性；支持现代软件工程技术，以保证最大程度上使用能大大提高软件开发生产率的现代技术。

1.2 相关领域研究现状

1.2.1 国外研究现状

GPS 作为人类第一个全球卫星导航定位系统，在研制过程中受相关技术手段的限制和经验上不足，并没有认识到原始数据生成仿真对于导航系统研制的重要作用。从目前所能找到的文献看，原始数据生成仿真是欧洲在研制 Galileo 系统过程中最早提出的。一些公司和研究部门也推出了相关软件，具有原始数据生成仿真的部分功能，比较典型的有美国 Analytical Graphic 公司的卫星工具软件 STK[10] 和瑞士伯尔尼大学天文研究所的 Bernese[11]。

(1)STK 是应用于航天通信技术领域的一个仿真和数据分析工具。STK 包括基础模块、分析模块、综合数据模块以及扩展、集成和接口模块，提供了丰富的基础库，具有轨道机动、姿态分析、链路分析、精密定轨等仿真分析功能。然而作为一个通用的商业软件，STK 目前还没有提供高精度的模型用于生成观测数据的仿真值，也暂时没有对 GPS 观测数据进行处理的功能，因而还不能用于进行原始数据生成方面的研究。

(2)Bernese 是一种应用广泛的导航数据处理软件，于 1999 年研制成功，到现在已经发展到 version5。Version5 提供对 GPS、GLONASS 等观测数据的处理功能，并将处理后的数据用于精密轨道生成，估计对流层天顶延迟和电离层延迟等功能。然而，Bernese 的这些功能是建立在已有观测数据的基础上的，没有提供高精度的仿真模型生成仿真数据，并不能对尚未建成的导航系统进行仿真试验。

(3)欧洲的 Galileo 卫星导航系统，目前处于系统开发和验证阶段。在此过程中，欧洲航天局(European Space Agency, ESA)组织开发了 Galileo 系统仿真工具作为系统开发建设的重要仿真验证手段。采用高精度的模型和观测数据生成地面站的观测量的仿真值，并将观测量用于 Galileo 系统的仿真试验，有力的支持了 Galileo 系统的系统技术指标的论证、系统关键技术分析和验证。系统具有以下主要特点：

- 仿真系统的使用将贯穿于伽利略计划整个过程，包括系统设计、系统验证、系统建设、系统管理；
- 仿真系统服务的对象是系统设计单位、工业制造部门以及最终的用户单

位；

- 仿真系统在伽利略计划中的定位是：具有支持大系统建设、操作准备、系统认证、系统维持、操作升级、用户开发等方面的能力，提供具有实际指导意义的定量分析支持结果；
- 系统模型和系统算法均为实际系统原型，具有高度的真实性；
- 系统具有良好的开放性能，支持模型和算法的即插即用，并提供通用的软硬件接口与实际系统实现互连互通；
- 具有良好的可扩展性，支持算法和模型的升级和扩展；
- 支持不同层次的仿真，从具体算法到系统控制；
- 支持多用户并行仿真；
- 系统具有良好方便的用户控制能力；
- 能够完成实时和超实时仿真；
- 对结果具有较强的综合分析能力。

目前，GSSF 仍在继续对原始数据生成进行研究，以适应伽利略系统研制的需要，预计不久将推出功能更强大、应用更广泛的版本。

(4)伽利略系统试验床(Galileo System Test Bed, GSTB)是伽利略计划设计开发和验证阶段的重要组成部分[22,23]。目前已经进行了试验床的第一阶段 GSTB-V1，其主要目标是评估未来伽利略系统在真实环境中的一些重要假设和运行目标，减少伽利略计划的风险，主要由以下几个部分组成：

- 试验性定轨与同步处理设施(Experimental Orbitography and Synchronisation Processing Facility,E-OSPF)，实现了伽利略定轨和同步的高精度原型算法，用于生成精确可靠的轨道和时钟预报以及空间信号精度(Signal In Space Accuracy, SISA)产品和导航电文；
- 试验性精确授时站(Experimental Precise Timing Station,E-PTS)，用于提供精确而且稳定的参考时间(试验性的伽利略系统时间)；
- 数据服务设施(Data Server Facility,DSF)，用于搜集、存档、检索和发布系统所使用和产生的科学数据；
- 试验性完好性处理设施(Experimental Integrity Processing Facility,E-IPF)，主要用于地面段、用户和系统的完好性分析。

由于已有条件充分和试验用途的不同，GSTB-V1 并没有使用高精度模型生成观测数据仿真值进行验证，而是使用真实的观测数据。GSTB-V1 所采用的数据是经过 DSF 处理的卫星在轨观测数据，已经实现了利用这些观测数据进行定轨与同步和完好性原型算法的验证和确认。

1.2.2 国内研究现状

观测数据生成与处理系统是整个仿真系统的核心部分，需要研究和建立大量复杂的、高精度的数学模型与算法，生成各类仿真数据，并提出相应的数据处理方案，以实现仿真系统的闭环测试以及用户导航定位性能验证。国内对于导航系统的仿真试验目前尚处于起步阶段，这些仿真试验主要集中在单个模型算法或者几个模型的简单组合，例如对星座优化、精密定轨、电离层延迟等领域的仿真。对单个模型和算法的仿真试验，其作用是局部的，不能从全局支持导航系统的研究工作。这方面，以国防科技大学航天与材料工程学院研制的航天分析与仿真基础程序库 AstroLib[12]为代表，该软件包含了大量航天系统分析所需的航天动力学函数和数值算法，采用标准 C++语言实现，可满足导弹及卫星的动力学仿真、航天器控制和深空探测等航天领域的应用需要。然而作为一个基础程序库，没有从系统工程角度针对导航系统进行集成，也不能提供原始数据生成仿真试验的功能。

一些单位和部门也试图对各个专业领域的模型和算法进行集成仿真，生成观测数据的仿真值，以支持导航系统的设计论证工作。然而，到目前为止都没有很好的解决方案，主要问题有：

(1)采用传统集成仿真方法，难以对来自各个领域的模型和算法进行有效集成，随着模型算法的增加，软件复杂程度加大，集成的困难也变得十分突出；

(2)软件结构固定，缺乏柔性，调整和升级都困难，在新的模型和算法出现时，替换部分算法或增加模块都需要较大幅度的改动软件结构；

(3)不能有效支持试验想定，用户难以根据试验需要更改试验的想定配置。

可见，缺少有效的建模仿真方法是造成上述问题的主要原因。必须从系统工程的角度出发，选择适当的建模仿真方法，分析原始数据生成的模型体系，建立易于集成和实现的模型框架，使得各领域模型能够按照该框架提供的接口、数据等规范编写代码，将其集成进来实现仿真功能。

对系统进行仿真研究首先要建立系统的仿真模型。建模是仿真的前提，是计算机仿真的关键技术内容之一，仿真模型不正确仿真的结果将失去意义。原始数据生成仿真是典型的离散事件系统仿真。传统建模方法[13]主要有实体流图法、活动周期图法、Petri 网方法：实体流图法与计算机程序流程图的画法类似，可以描述临时实体产生、流动、消亡及其被永久实体加工、处理的过程和逻辑关系，应用比较广泛，所建立的实体流图模型易于被转换为面向事件的仿真模型；活动周期图法针对实体的行为模式进行建模，可以直观地表示出某类实体生命周期中的活动和状况，具有规范化的特点，对不同实体之间协同关系的描述十分清晰、明确；而且，应用活动周期法建立的系统模型转换为面向活动的仿真模型比较方便。Petri 网方法适于建立加工系统等多种离散事件系统的模型，并可对网系统的特性进行比较严密的数学分析，看到对并发、冲突、死锁等现象的深刻认识，应用也比较广泛。

这些传统建模方法虽然有较好的特性和广泛的应用，但是对于航天工程系统的仿真建模，传统的建模方法显然无法有效的适应原始数据生成仿真的特点并有效克服前面所述的困难，必须采用更先进的建模仿真方法。

近年来出现了一些较先进的建模仿真技术，如基于设计模式的基本对象模型(Base Object Model, BOM)[14,15]，基于 XML 的仿真参考标记语言(Simulation Reference Markup Language，SRML)[16]等，这些方法都有其各自的特点，然而对于类似原始数据生成这种航天器的仿真工程优势不明显，从现有的参考文献看鲜有这方面的研究和应用。

1.3 仿真模型可移植性规范简介

鉴于航天工程总体仿真在模型集成和试验分析上的特殊需求，2000 年欧洲航天局主持定制了模型可移植性规范(Simulation Model Portability, SMP[17-20])，并发布了 SMP1.0 版本，支持包括卫星导航、火星探测、金星探测等一系列工程总体计算模型的集成工作，其主要特点表现在：

(1)尽量减少模型与计算机环境的直接交互；

(2)模型接口标准化；

(3)模型接口简单化；

(4)模型易于为其他系统开发人员所理解；

(5)保证模型的运行机制能够与模型分离。

同时 SMP 也提供了专业领域模型开发指南，包括开发建议、模型设计模式、模型文档模板、数据文件交换标准和模型兼容性测试等，确保领域模型开发和集成的有效性。

目前 SMP1 已经用于许多航天器仿真开发(见表 1.1)，ESA 的两个主流仿真平台 EuroSim 和 SIMSAT 已经集成了对 SMP 的支持。然而，使用中也发现 SMP1 的作用没有预想的那么乐观，其中很多原因是 SMP1 本身的局限性造成的，包括以下几个方面：

(1)对模型的重用支持不够；

(2)对模型的集成支持不够，尤其是对于复杂的仿真系统；

(3)模型间通信仅支持动态调用方式，效率不高，实际中许多应用需要静态调用方式；

(4)不支持面向对象及基于组件的技术；

(5)缺乏对模型元数据描述的支持；

(6)不支持仿真的动态配置；发布调用需要手工编写，繁琐而且易错；

(7)调度机制比较原始；

(8)缺乏对模型初始值的描述；

(9)不支持对仿真状态的获取或改变；

表 1.1 SMP 在航天器仿真中的应用情况[21-24]

项目	单位
Generic Project Test Bed(GPTB)	ESTEC
Rosetta Operational Spacecraft Simulator	ESOC
Mars Express Operational Spacecraft Simulator	
Cryosat Operational Spacecraft Simulator	
Radarsat2 Operational Spacecraft Simulator	
AQUA Payload Simulator	
Galileo System Simulator Facility	ESA
Portable Satallite Simulator Mk III	
MCS Test and Validation Tool	
NDIUlite	

针对以上问题，ESA 联合航天器研制部门提出 SMP2 标准。SMP2 的主要目标包括：

(1)改善模型的可重用性：在仿真器开发中，模型缺乏重用性是提高开发效率和可靠性的主要障碍。重用性要求模型在设计时即考虑重用。目前实现模型重用还比较困难，因为还鲜有标准辅助模型开发人员实现重用。SMP2 的一个主要目标就是辅助建模人员开发可重用的模型，以便这些模型可以部署在不同的仿真系统、可以跨不同的仿真环境的使用、可以在不同组织或不同的使命开发阶段交换。

(2)提高模型的可移植性：同 SMP1 相比，SMP2 强调模型在不同计算平台之间、不同仿真运行环境之间的移植。

(3)增强模型的可维护性：通过定义标准的模型元数据定义，并合理的封装模型实现。

(4)支持模型的集成：SMP2 的另一个主要目标是提供开放的标准定义模型的集成方式。改善模型集成非常重要，因为模型集成可以提高模型开发的速度。

(5)便利系统工程的数据的交换：通过使用类似 XML 的开放的可访问格式定义 SMP2 模型，方便系统工程数据的交换。仿真可以容易的吸收系统工程数据（甚至完全由这些数据配置）这一点非常重要。

(6)支持并采用现代软件工程技术，包括面向对象和基于组件的设计；

(7)支持对模型元素的描述；

(8)采用开放的标准，以减少对专用技术 License 的依赖。

通过对 SMP2 的特点分析，可以看出，SMP2 提供了一套非常适合原始数据生成的建模仿真方法。因此，本文试图采用 SMP2，从系统工程的角度研究和建立了一个适合原始数据生成仿真试验的模型框架。

1.4 论文研究的主要内容及写作框架

当前国外卫星导航系统的分析试验大部分已基于计算机仿真技术完成，我国已经建成了北斗一号双星定位系统，在北斗的开发建设中已经完成了部分试验分析工作，如在系统演示验证阶段通过通信卫星完成系统原理的验证。从目前的研究状态来看，我国与其他先进国家在工程大系统模型集成和试验分析等方面还处于起步阶段，没有形成类似于 SMP 的模型开发标准，在应用上也没有形成可用的工程大系统模型开发工具。有鉴于此，本文主要研究了基于 SMP2 的原始数据生成模型框架及其仿真分析。本文的主要工作如下：

(1)研究了 SMP 的建模仿真机制，重点研究了基于 SMP2 的仿真模型集成方法；

(2)参考 GSSF 以及有关文献，提出了一个初步的原始数据生成模型体系，对体系内的模型和算法进行实体模型分析、数据流分析和面向对象分析；

(3)在模型体系分析的基础上，依据 SMP2 的建模机制，分别建立了基于接口集成的模型框架、基于事件集成的模型框架、基于数据流集成的模型框架和基于组件集成的环境段模型框架，并讨论了这些集成方式的仿真模型装配和调度机制。在此基础上建立了基于 SMP2 的原始数据生成模型框架，并对该框架的扩展性进行了讨论；

(4)针对该模型框架，依据 SMP 的仿真运行机制，提出了仿真调度框架，并进行了基于精密星历生成轨道的原始数据生成仿真试验。

为使阐述以上内容富有条理性，本文对论文章节进行如图 1.2 的安排。各章的主要内容如下：

第一章：绪论，主要介绍了论文的研究背景，目前相关领域的研究现状，提出了本论文需要解决的问题，进而给出了论文研究的目标、研究意义以及主要内容。

第二章：基于 SMP2 的原始数据生成仿真方法，本章讨论了 SMP2 的基本概念和建模仿真机制，基于 SMP2 的建模开发过程，重点研究了基于 SMP2 的集成方法；

第三章：原始数据生成模型体系分析，参考 GSSF，本章提出了一个初步的原始数据生成的模型体系，并对体系内的模型进行了分析，由体系的模型和算法决定的数据流，分析了模型间的关系，在此基础上确定了基于 SMP2 的仿真模型体系；

第四章：基于 SMP2 的原始数据生成模型框架，本章根据对体系内的模型分析和 SMP2 的建模仿真机制，建立了基于 SMP2 的原始数据生成模型框架；

第五章：试验调度框架及仿真试验，根据基于 SMP2 的模型框架，使用相关工具设计和开发了仿真模型，建立了试验调度框架，并在 SMP2 仿真器的支持下进行了初步的仿真试验分析；

结束语：总结与展望，本章在对论文总结的基础上，提出了论文进一步可以研究的方向。

最后，在附录 A 中给出了与文中调度文档相关的仿真模型设计、装配文档的开发界面，在附录 B 中给出了解释仿真模型设计、装配和调度文档的有关代码。

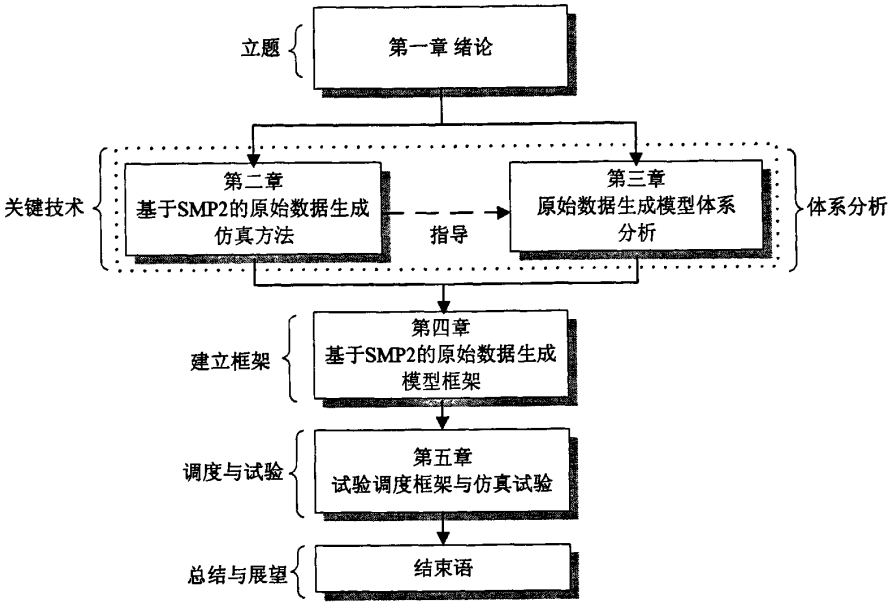

图 1.2 论文组织结构

第二章 基于 SMP2 的原始数据生成仿真方法

在大型复杂仿真系统的开发中，基于组件的技术是解决软件重用和组合的主流方法。基于模型驱动架构(Model Driven Architecture, MDA)的仿真开发主要采用 MDA 的思想，将仿真模型分为设计模型组件和仿真运行组件，通过建立面向仿真设计模型组件的仿真元模型规范、仿真组件对象规范、仿真服务规范和仿真模型组合规范确保仿真模型的设计、开发、集成和运行的一致性和仿真系统的可持续性[25]。SMP2 借鉴了基于组件设计(Component-based Design，CBD)和模型驱动架构的思想，采纳了 UML、XML 等开放标准，具有强大的生命力。SMP2 标准由 SMP2 组件模型、元模型、C++映射、使用手册四份文档共同组成。组件模型明确地定义什么是仿真组件，组件的接口规范如何描述，仿真由哪些组件构成。SMP2 定义了三种类型的组件：仿真器组件、仿真服务组件和模型组件。SMP2 采用仿真模型定义语言(Simulation Model Definition Language, SMDL)描述平台无关模型。平台无关模型存储为 XML 文档。作为一种模型描述语言，SMDL 自身的构造和规则需要用元模型精确定义。SMDL 元模型规定了 SMDL 的组成元素及元素之间的关系，主要由 Catalogue、Assembly 和 Schedule 三部分组成：Catalogue 保存模型的类型信息；Assembly 描述模型实例如何基于 Catalogue 中的类型信息在一个具体的想定中进行装配；Schedule 描述了 Assembly 中的附加调度信息。C++映射文档明确地定义了 SMP2 组件模型和 SMP2 元模型到 ANSI C++程序设计语言的映射。SMP2 手册则介绍了 SMP2 标准的基本概念和采用 SMP2 开发仿真模型的方法。同时 SMP 也提供了专业领域模型开发指南，包括开发建议、模型设计模式、模型文档模板、数据文件交换标准和模型兼容性测试等，确保领域模型开发和集成的有效性。

2.1 SMP 的概念框架

2.1.1 SMP2 的仿真模型共性

SMP2 标准的目的是提供一个模型框架，以实现仿真模型的(计算)平台独立、跨仿真平台互用和重用。为了实现这一目标，SMP2 仿真模型必须满足下面两大基本需求：

(1)公共概念(Common Concepts)：所有的 SMP2 模型必须采用公共的高层概念建立，这些概念涵盖了基本的建模问题。这使得模型能在一个抽象的水平上开发，这对模型的平台无关和重用非常重要。

(2)公共类型(Common Type System)：所有的 SMP2 仿真模型必须基于公共的类型。这一点确保不同的模型对于基本的类型有着相同的语法和语义理解，这对模型的互操作/互用非常重要。

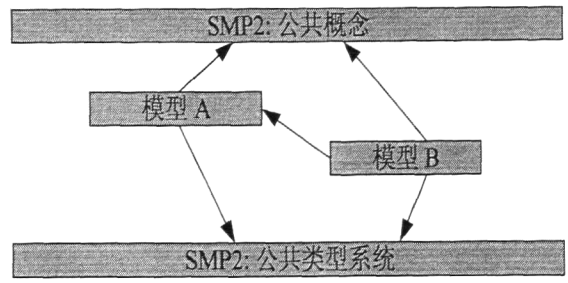

图 2.1 嵌入于公共概念和公共类型系统之间的 SMP2 模型

其中，前一个需求要求所有的 SMP2 模型派生于相同的基本概念，第二个需求要求所有 SMP2 模型基于相同的类型系统建立，一个 SMP2 仿真模型将位于 SMP2 的公共概念和公共类型系统这两层之间，如图 2.1 所示。

2.1.2 SMP2 的顶层结构视图

SMP2 由三层顶层视图构成，对应系统的不同抽象层次，其顶层结构视图如图 2.2 所示。

处于最顶层的是对应现实世界或者被建模的真实系统。在这个层次，SMP2 并没有提高建模的概念，而是借助于已有的建模工具，如 UML 与系统工程建模语言(SysML)等工具。

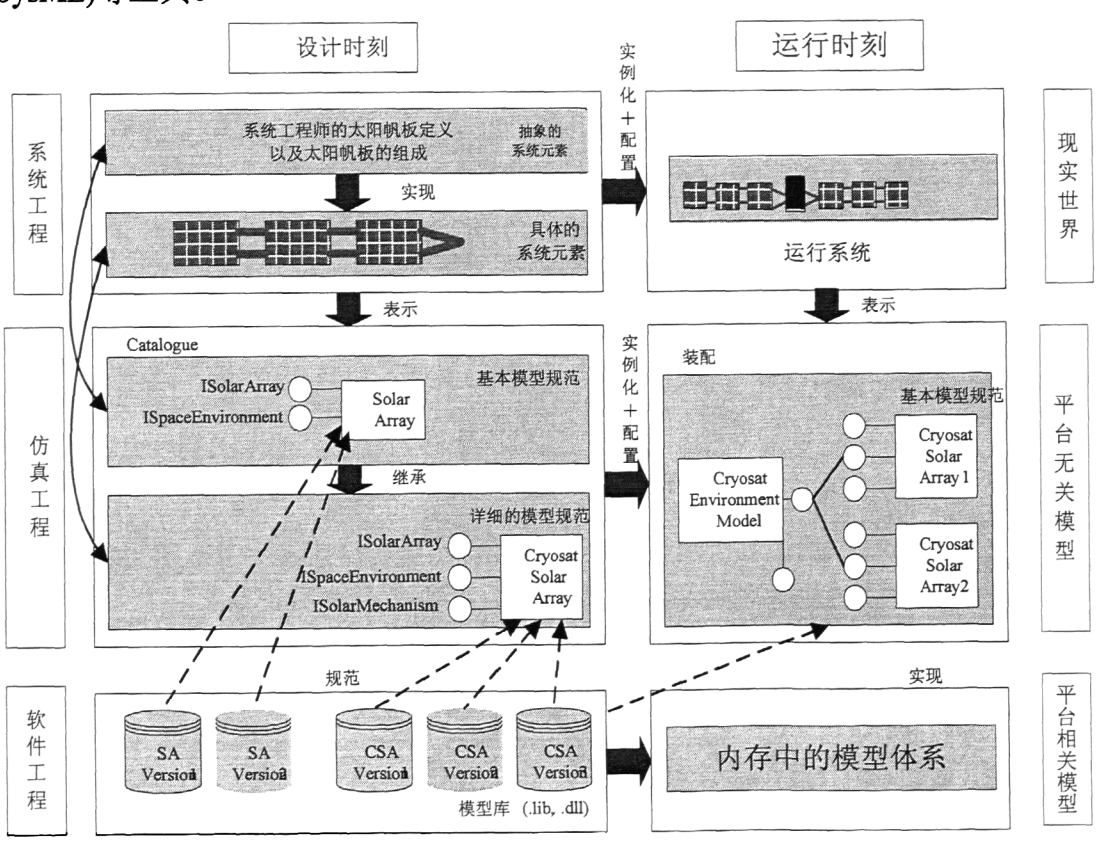

图 2.2 SMP2 的模型体系结构

第二层描述仿真系统的平台无关模型(Platform Independent Model, PIM)。形成了不同仿真模型的规范层次，主要包括定义模型的字段、属性和接口规范以及模型使用的数据类型(保存在设计文档 catalogue 文件中)、定义模型实例如何集成与配置(保存在装配文档 Assembly 文件中)、定义模型实例如何进行调度(保存在调度文档 Schedule 文件中)。这些文件都是 XML 格式的，其格式由 SMDL 规定。本层的 Assembly 文件和 Schedule 文件是可选的，主要是为动态和柔性仿真提供支持。在提供动态和柔性仿真支持时，采用独立于模型实现的外部 XML 文件进行配置和调度。设计文档 Catalogue 文件用于说明模型的准确接口，可以认为是机器可读的模型文档，因而对每一个 SMP2 模型具有特殊的作用。

第三层是仿真系统的平台相关模型(Platform Specific Model, PSM),即 SMP2 仿真模型运行组件，当前 SMP2 模型的实现层次为 ANSI/ISO C++。SMP2 模型可以手工实现，然而模型实现的很大一部分工作是与集成相关的，可以根据设计文档 Catalogue 中的信息自动生成。为了支持信息的自动生成，SMP2 通过组件模型描述了 SMP2 组件(包括模型和服务)的标准接口以及一组标准仿真服务(时间、事件调度、进程交互、日志管理等)，并且建立了这些仿真元模型到特定目标平台的映射，确保仿真元模型和组件模型映射到 ANSI/ISO C++代码。

从顶层结构视图中可以看到，该图纵向上分为两部分，左边部分对应于模型的定义(对应于面向对象术语中的类)，右边部分对应模型的世界中的实体(对应于面向对象中的实例)。在该例中，建模人员首先用抽象形式描述了太阳帆板(Solar Array)是什么以及它由什么组成。根据仿真分析的需求，该信息可以在 SMP2 Catalogue 中使用一个名称为 Solar Array 的模型来表示。此模型可以用于派生出更加详细的模型如例子中的 Cryosat Solar Array，它定义了附加的新的接口 ISolarMechanism。在装配中，可以确定和配置这些模型的实例，决定相关接口的引用，如例子中的 ISpaceEnviroment，并且通过 UUID(Universally Unique Identifier)选择适当的模型实现。在实现层，相应的模型组件工厂通过 UUID 指向 Catalogue 中相关的模型规范，对于一个给定的模型规范，允许采用工具提供智能用户接口支持模型实现的选择工作。

2.2 基于 SMP2 的仿真模型集成方法

SMP2 总结了常见的模型集成方法，提供了基于类的集成、基于接口的集成、基于组件的集成、基于事件的集成以及基于数据流的集成五种集成方法。同一个仿真模型与其它模型之间往往同时采用上述的几种集成方法，因而对于 SMP2 集成的仿真模型，其单个仿真模型的设计，是与其它仿真模型之间集成方式的综合。

2.2.1 基于类的集成方法

基于类的集成由具有保存内部状态的字段、实现所需行为的操作、响应时间或动态事件的入口点的模型组成。类提供了实现继承的机制，但是 SMP2 模型只提供了单实现继承。

基于类的集成所需的设计元素有：模型、字段、操作和入口点，如图 2.3 所示。

模型：在 SMDL 中每个模型设计的核心元素是 Model。基于最少的元素考虑，Model 应提供描述其内部状态的字段，描述模型行为的操作以及用于向调度器(Scheduler)或事件管理器服务注册的入口点。

模型被映射为 ISO/ANSI C++中实现了 IModel 接口的类，即具有返回名称、返回描述、返回父模型、可以进行字段发布以及设置服务提供对象的函数。一个模型可以继承一个单基类的实现。

字段：每个字段都是模型内部状态的一部分。每个字段引用一个类型，可能还需要包含一个缺省值。对于引用类型(即类或其他模型的引用)，隐含的缺省值为 null。一个字段具有可见属性，用于描述该字段是否仅能被模型本身访问(private)、能被本模型的子模型访问(protected)、能被在同一个包内的所有模型访问(package)或能被所有模型访问(public)。

一个字段被映射为 ISO/ANSI C++中 struct 或 class 的成员变量。如果字段是静态字段，它还可以成为 C++中 class 的一个类变量。

操作：一个模型的操作是该模型的一个成员函数也就是模型的方法，这些函数可以获取外部传入的参数并且可以具有返回值。在调用这些函数时，需要想函数传递一个模型实例的引用(在 C++中称为 this)。这些函数可以访问该模型实例的所有字段。

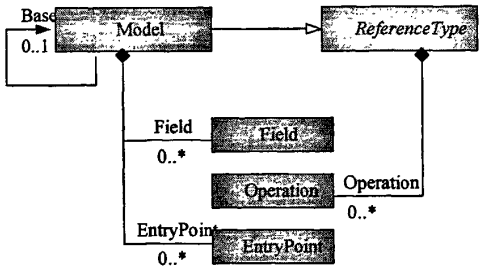

图 2.3 基于类集成的仿真模型示意图

操作被映射为 ISO/ANSI C++中 class 或 struct 的成员函数。如果若该操作为静态操作，它还可以映射成为 C++中 class 的一个类函数。

入口点：入口点是模型的一个简单操作，在模型实例中表示为一个没有参数和返回值的成员函数。一般模型配备两个入口点函数，一个为初始化入口点，用于模型实例的初始化；另一个入口点函数用于执行某种计算。入口点总是具有 public 属性，以便其它模型或组件调用。

一个入口点被映射为 ISO/ANSI C++中没有参数和返回值的成员函数。当提供该入口点时，即一个服务要使用该入口点时，首先要将该入口点包装为一个 IEntryPoint 的实现类，该类只有一个没有参数和返回值的执行入口点的函数。

2.2.2 基于接口的集成方法

基于接口的集成方法将接口作为标准的模型间通信机制。它将接口(约定)的定义同其实现相分离。基于接口的设计方法中，模型可以提供任意数量的接口。

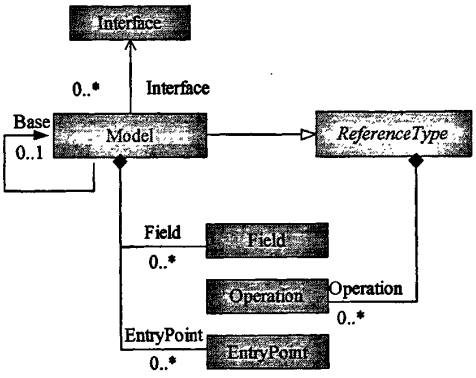

图 2.4 基于接口集成的仿真模型示意图

基于类的集成所需的设计元素有：接口、操作、属性，如图 2.4 所示。

接口：接口定义了模型间合同约定。每个实现该接口的模型必须提供所有接口规定的功能，这样每个使用该接口的模型可以基于一个完全的实现，即认为接口的实现肯定存在。接口是一种模型解藕机制，并不提供对字段的访问，而只提供对操作的访问。针对特殊的单值的读写操作，可以增加对字段的访问。

ISO/ANSI C++语言本身既不支持接口机制，也不将接口继承和实现继承分离。因此，一个接口被映射为 C++中的虚类，即所有的接口操作和属性对应的 C++成员函数都必须声明为纯虚函数。

操作：与基于类的集成类似，接口定义了操作。由于接口并不提供实现，每个模型必须为其提供的每个接口的每个操作提供实现，该实现可以从其基类继承得到。

接口的操作被映射为 ISO/ANSI C++的纯虚成员函数。从该接口继承的每个非抽象类必须提供每个操作的实现，否则将包含仅能由抽象类定义的抽象方法。

属性：接口不提供对字段的访问，通常是提供一对操作通过接口读写字段。为简化这一对操作的描述，引入了 Property 元素。Property 通常映射为一个字段的 getter 和 setter 操作。Property 可以通过仅提供一个方法赖限制对字段的只读何只写访问。

ISO/ANSI C++语言本身并不支持属性机制。由于属性可以转化为一到二个读/写值方法，所以属性的 getter 何 setter 可以被映射为 ISO/ANSI C++接口中的纯虚成员函数。

2.2.3 基于组件的集成方法

在基于接口的集成方法基础上，基于组件的集成提供了描述聚集和组合的机制。这些机制描述了模型间的依赖型：引用或包含。

在基于组合的集成中所需的设计元素是聚集和组合，如图 2.5 所示。

聚集：模型的聚集意味着含有对其它模型的引用，而不是拥有这些模型(即不是这些模型的父模型)。对另一个模型的引用是因为需要其它模型才能正常工作或是因为需要使用其它模型的接口功能。一个引用允许在其中增加新的组件。指向其它模型的引用规定了可以引用的其它模型数的上下限。典型情况如表 2.1 所示。

SMDL 中用引用来表示聚集。每个包含引用的模型必须实现一个名 IAggregate 接口以支持对引用的访问。针对每个引用，模型必须返回一个实现了 IReference 接口的引用，IReference 接口提供对模型所聚集的组件的访问。模型的每个单个引用都由 IReference 的一个实例表示。

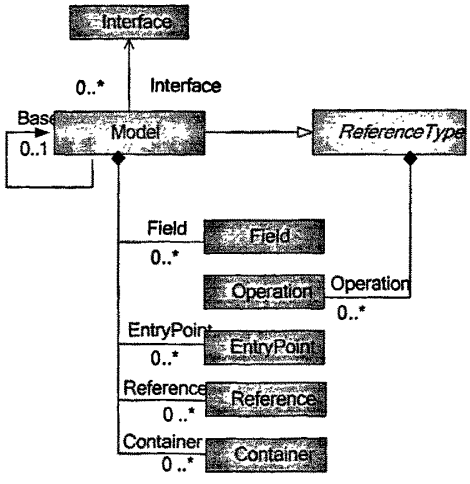

图 2.5 基于组件集成的仿真模型示意图

组合：模型的组合拥有被组合的模型(即作为这些模型的父模型)。模型通过模型容器所拥有的子模型，而子模型一般是系统的子系统。

表 2.1 引用的典型上下限

下限	上限	UML	典型应用
0	1	0..1	可选引用，模型可以使用另一个模型，但不能依赖该模型。
1	1	1	基本引用，模型依赖另一个模型提供的功能。
0	-	0..*	无限引用，模型能使用任意数量其他模型的功能。

容器允许向其中添加新组件，并且规定了可包含的其它模型数的上下限。典型情况如表 2.2 所示。

表 2.2 容器的典型上下限

下限	上限	UML	典型应用
0	1	0..1	可选的子模型，模型可以包含另一个模型，但不能依赖该模型。
1	1	1	基本子模型，模型依赖一个提供某些功能的子模型。
0	-	0..*	无限子模型，模型能拥有任意数量的特定类型的其他子模型。

在 SMDL 中使用容器来表示组合。每个包含容器的模型必须实现一个名为 IComposite 接口以支持对其容器的访问。对每一个容器，模型必须返回一个实现了 IContainer 接口的容器。IContainer 接口给出了模型中所组合组件的访问方法。模型的每个单个容器都由一个 IContainer 接口的具体实例表示。

2.2.4 基于事件的集成方法

基于事件的集成可以与基于类的集成一起使用，也可以在基于接口的设计或基于组件的设计基础上应用。基于接口的设计中，接口的使用者主动调用该接口的提供者，在此之前，使用者必须获取该提供者。如果它不知道提供者是否存在，将不能通过接口获取所需的功能。在基于事件的设计中，情况有所不同：事件提供者(Event Source)调用所有订购该事件的使用者(Event Sink)。不同于接口定义了一组操作，一个事件总是映射为一个单一的函数。基于事件的设计中的模型可以提供事件也可以使用事件。一个事件定义了一个具有类型参数(称为事件变量)的一个函数，所以需要一个事件类型实现提供方和使用方的匹配，仅仅事件类型匹配的事件源和事件槽可以建立连接。

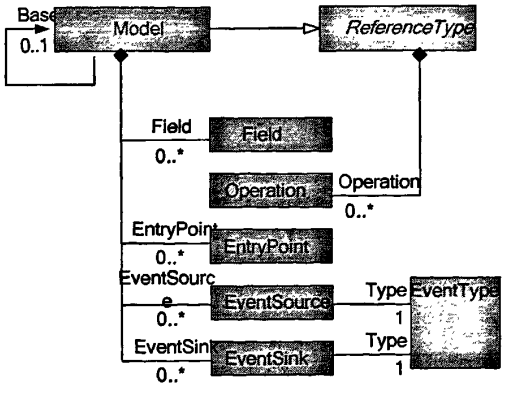

图 2.6 基于事件集成的仿真模型示意图

在基于事件的设计中，设计元素有：事件类型、事件源和事件槽，如图 2.6 所示。

事件类型：事件类型定义了事件的名称和参数格式。事件参数格式应该是一个值类型，即一个由其值完全定义的类型。仅仅这些类型可以被作为事件参数在事件源和事件槽之间传递，而不是传递内存地址。

一个事件类型被映射为 C++中事件槽方法中事件变量的类型。

事件源：事件源是模型的一个具名特征，由一个事件类型定义。除了这些类型外，可以规定该事件源能够连接多个事件槽或仅仅一个。事件源对应一个触发函数，按照事件类型定义传递一个事件参数。接下来事件源将调用所有目前相连的所有事件槽，并传递对发送者的引用及事件参数。

一个事件源映射为一个具有 Subscribe()和 Unsubscribe()方法的接口。

模型的每个单个事件源都由一个 IEventSourse 的命名实例表示。

事件槽：事件槽是模型的一个具名特征，由一个事件类型定义。可以连接到同样类型的任意数量的事件源。无论何时这些事件源被触发，将调用本事件槽，并传递对事件发送者的引用以及事件的参数。

一个事件槽映射为包含 Notify 方法的接口。模型的每个单个事件槽都由一个 IEventSink 接口的命名实例表示。

2.2.5 基于数据流的集成方法

基于数据流的设计可以在基于类的设计方法上实现。这种设计方法为模型的字段添加了两个属性，称之为输入标志和输出标志，这两个标志标识了模型的输入和输出字段。不同于上述其它设计方法，模型并不主动的执行数据的流动，而是通过一个外部协调器基于模型间的输入输出字段及字段链实现数据的传输。SMDL 提供一种机制用于将 catalogue 中模型的字段标记为输入或输出字段，并定义了 Assembly 中模型实例之间的数据流链。但是，该信息模型实现并不主动估计，因为模型需要输入数据。

除了上述不同组合方式中介绍的设计元素外，设计组件元模型还定义了一些应用于所有组合方式的基本概念：用于分组机制的命名空间以及 float、integers、enumerations、arrays、strings、structures 和 classes 的用户定义类型。

命名空间：命名空间是一种分类机制。它允许将一些结构放入模型库中。

命名空间映射为 ISO/ANSI C++中的 namespace。由于名字空间可能包含嵌套的命名空间，C++ namespace 可能包含下一级的 namespace。

用户自定义的类型也映射成 C++中的相应数据类型，例如 Integer 实型映射成具有更大范围信息的 32 位有符号实数机制。

2.3 基于 SMP2 的仿真模型开发集成框架

基于 SMP2 的仿真开发可以包括仿真模型设计、仿真模型开发、仿真模型集成和仿真模型运行四个阶段，其开发与集成过程如图 2.7 所示。

2.3.1 仿真模型设计阶段

仿真模型设计阶段的主要任务是基于 SMDL Catalogue 设计仿真模型，生成 XML 格式的模型设计文档(*.catalogue)。在此阶段，采用仿真模型设计组件定义工具生成仿真模型设计组件模型即模型设计文档；通过仿真模型文档生成工具形成仿真模型设计文档；通过仿真模型验证工具验证仿真模型设计组件的语法正确性。想定生成系统和实验管理系统也可以根据仿真模型设计组件进行基于模型组合的想定生成和实验设计。

仿真模型设计的主要依据是，根据物理模型和算法的需要设置仿真模型的字段和操作，根据模型之间的关系采用适当的集成方式。对于基于 SMP2 的仿真模型设计，模型之间的集成方法指的是 2.2 中提到的五种集成方法。在复杂的模型体系中，对同一个模型往往采用多种集成方法。

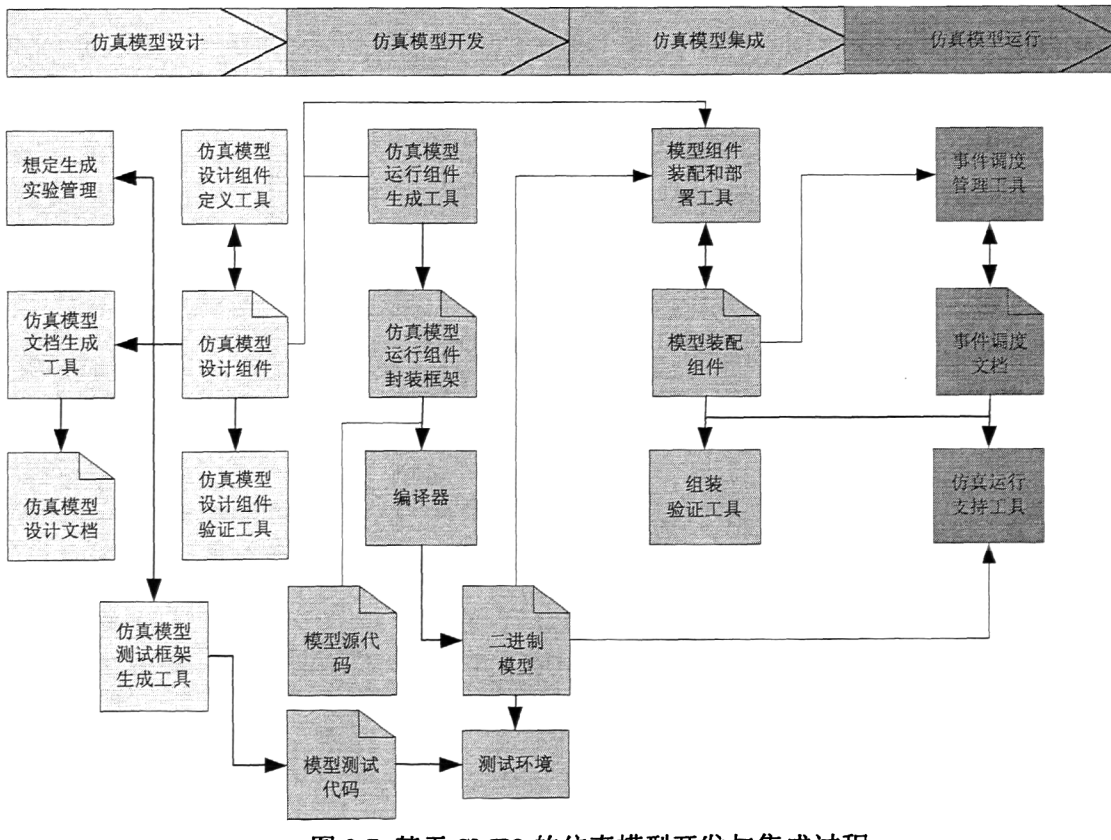

图 2.7 基于 SMP2 的仿真模型开发与集成过程

2.3.2 仿真模型开发阶段

根据仿真模型设计组件生成仿真模型运行组件代码框架，开发人员可以编写特定的仿真模型算法，以及在接口函数和入口点函数手工添加的实现代码，最终

形成仿真模型运行组件，即二进制模型库；采用仿真模型测试框架生成工具生成仿真模型测试框架，通过测试环境测试仿真模型运行组件。

代码框架由仿真模型设计文档决定，根据 SMP2 提供的 C++映射规范，采用代码生成器生成。代码框架下，开发人员只需关注具体算法的实现，这些算法，可以在代码框架下编写，也可以使用外部库文件(.dll，.lib)。

2.3.3 仿真模型集成阶段

仿真模型集成阶段的主要任务是基于 SMDL Assembly 装配仿真模型，生成 XML 格式的装配文件(*.assembly)。在此阶段，采用仿真模型装配模型定义工具定义仿真模型模型装配文档，然后由仿真器根据该文档和运行组件自动进行仿真模型实例装配，形成内存中的模型实例配置；通过模型部署工具确定组合模型中的模型实例配置；通过组合模型验证工具检验组合模型装配语法和语义的正确性。

模型的所采用的集成方法不同，则装配方法也不同。如果采用了基于类的模型集成方法，则在集成阶段需配置字段值；如果采用基于接口的集成方法，则需在接口的实现模型和接口的消费模型之间建立连接；如果采用基于组件的集成方法，则需指定模型所处的容器及所引用的模型；如果采用基于事件的集成方法，则需在发送事件模型和接收事件模型之间建立连接；如果采用基于数据流的集成方法，则需在数据的输入字段和输出字段之间建立连接。

对于同一个仿真模型，可以根据仿真任务的需要，在装配过程中可以指定产生多个内存实例，并对每个实例分别进行配置。

2.3.4 仿真模型运行阶段

仿真模型运行依赖于仿真模型的装配文档和调度文档，运行阶段的主要任务是基于 SMDL Schedule 生成 XML 格式的调度文档(*.schedule)。调度文档定义了仿真任务和事件信息，不同的调度文档对应不同的仿真模型运行调度顺序和方式。

在此阶段，通过事件调度管理工具确定仿真运行中可能发生的不同事件或事件序列，形成事件调度文档；仿真运行支持工具可以基于组合装配文档和事件调度文档支持组合模型或单个模型的仿真执行。

在 SMP 中，仿真模型的调度有两种：Trigger(触发模型)和 Transfer(传输数据)。Trigger 是指调度仿真模型，需要指定仿真模型及其入口点，对同一个模型指定不同的入口点函数，则其调度也不同；Transfer 是指启用在模型装配集成阶段建立的输入－输出字段链接，在基于数据流集成的两个模型字段之间传输数据。

SMP 以 Task (任务)组织 Trigger 和 Transfer，一个 Task 可以包含若干个 Trigger 和 Transfer，也可以包含其它 Task。一般而言，同一个 Task 所包含的 Trigger 和 Transfer 在执行时间上不能有冲突，即它们的执行结果不应依赖于执行顺序。

SMP 中以 Event(调度事件)组织 Task，一个 Event 可以包含若干个 Task。在这里 Event 是指执行一个 Task 集合的一个时间点，调度事件分为两种：可以仅被调用一次，称为简单事件；可以重复调用，称为周期事件。SMP 支持四种不同的时间格式定义事件的执行时间：

(1)仿真时间(Simulation Time)：是一个相对时间，从仿真启动的 0 时刻开始，在仿真环境处于执行模式下推进时间；

(2)纪元时间(Epoch Time)：是一个绝对时间，一般与仿真时间一起推进。纪元时间和仿真时间有一个固定的偏差，这个偏差可以在仿真过程中人为干预改变；

(3)使命时间(Mission Time)：是一个相对时间，其数值相对于使命开始日期。使命时间使用一个与纪元时间的固定偏差来维护，因此与仿真时间和纪元时间一起推进。

(4)Zulu 时间：为计算机时钟时间，也称墙上时钟时间，不管仿真处于 Execution(执行)还是 Standby(等待)模式，该时间均向前推进，不与仿真时间、纪元时间或使命时间发生关联。

2.4 基于 SMP2 的 RDG 模型框架开发过程

由前面的讨论可以看出，基于 SMP2 的建模仿真主要任务就是采用恰当的集成方法，对仿真模型进行集成仿真。集成方法的采用，要根据仿真模型在模型体系中的特点，这些特点主要指模型之间的数据流关系、接口关系和组合关系。对于原始数据生成仿真而言，由于模型和算法众多，对每一个模型和算法都建立仿真模型进行集成仿真，显然加大了建模仿真任务的复杂性，延长了开发周期。可见，仿真模型的设计并不是一个随意的过程，应当根据仿真需求需要确定仿真研究的问题和背景，收集权威资料、描述模型和算法、确定实体及其算法、确定仿真模型、建立模型框架和建立调度框架，图 2.8 显示了这一过程。

(1)收集权威资料

在此阶段，主要完成仿真需求的定义工作，根据仿真背景收集仿真将要解决的问题领域相关的“权威”资料，包含了仿真将要表示系统的物理过程、物理原理和工程原理。例如导航卫星的工作原理，电离层的物理特性，地面站的组成及运行机制等。仿真背景只是说明了仿真中需要表示的行为和过程的信息来源。如果中间需要考虑相关的物理原理和工程原理，还需要界定相关的公式、假设条件和影响因素，如果这些公式可能来源不同，必须确保不能出现相关的物理原理和工程原理假设条件冲突的问题。这些资料可以从有关领域的权威专家获得，也可以借鉴已有的系统或参考文献。

(2)描述模型和算法

模型是对相应的真实对象和真实关系中那些有用的和令人感兴趣的特性的抽象，是对真实系统的某些本质方面的描述，它以各种可用的形式提供被研究系统的信息。根据收集到的资料，进行仿真需求分析，将仿真任务中出现的行为、过程、实体、算法等抽象成数学描述或物理描述，也就是模型和算法。

(3)确定实体模型及其算法

在离散事件系统中，实体是被仿真系统中可单独辨识和刻划的构成要素。例如，导航卫星、地面接收站等。在建模人员看来，实际系统就是由相互间存在一定联系的实体集合组成的，实体间的相互联系和作用产生系统特定的行为。因此SMP2 的仿真模型一般以实体为基础，根据仿真任务的需要和模型算法之间的复杂性，可以对重要的模型和算法也可建立仿真模型。可以看出，建立仿真模型之前，有必要确定模型的实体，将其它的部分模型和算法归为实体中的算法。

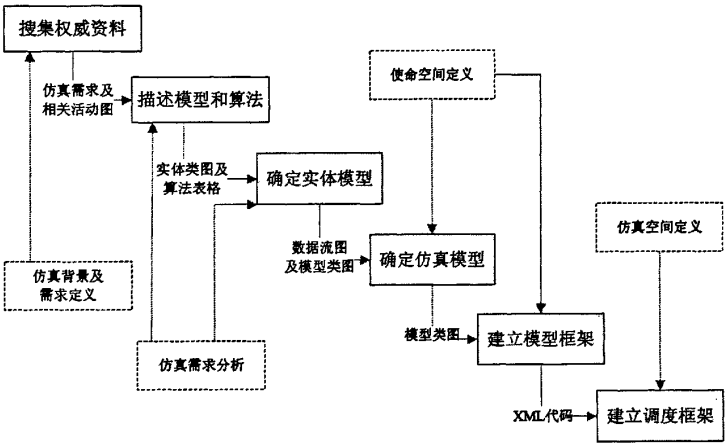

图 2.8 基于 SMP 的 RDG 模型框架开发过程

(4)确定仿真模型

原始数据生成模型和算法复杂繁多，仅从实体模型确定仿真模型不一定合适，比如有些实体模型比较简单，可以归结为其它模型的一个模块；而有些非模型与其它模型间具有复杂的数据、接口关系，对这些模型建立仿真模型有利于仿真的实现。然而仅从单个模型和算法根本无法确定模型和算法的复杂程度，需要从全局上分析模型和算法中的组合关系、接口关系、计算过程和数据传输关系。因此在实体模型分析的基础上，从多角度分析模型之间关系，这些关系包括数据流关系、组合关系、继承关系等。分析这些关系的重要方法是对模型和算法建立数据流图和面向对象分析图。根据模型和算法的关系分析，可以考虑对关系复杂的模型和算法确定为仿真中的仿真模型，这样加上实体模型，原始数据生成中的模型体系中的仿真模型就确立了，当然仿真模型并不是唯一确定的。

(5)建立模型框架

在此阶段，根据仿真模型之间的关系和 SMP2 仿真模型集成方法特点，确定仿真模型的集成关系。根据组合关系确定仿真模型之间基于组合集成；根据接口

关系确定仿真模型之间基于接口的集成；根据数据流关系确定仿真模型之间的基于数据流集成或者基于事件集成。

(6)建立调度框架

同的仿真任务有不同的仿真模型调度过程，因此，要根据仿真任务的需要，设计仿真模型的调度过程，及仿真任务的调度框架。

第三章 原始数据生成模型体系分析

根据 2.4 中提出的基于 SMP2 的原始数据生成模型框架开发过程，首先必须搜集有关原始数据生成的权威资料。由于原始数据生成需要高精度的模型和算法进行计算，本文对可能影响观测量的因素都进行了充分考虑，此外由于存在大量的模型和算法，我们在本文中只列举了其中的主要部分。对所有的模型和算法建立仿真模型必将导致建模困难，计算复杂，显然不是最好解决办法。为此我们进行了实体模型分析，并将部分算法和模型归结为实体模型的一个模块；为弄清这些模型和算法的数据流关系、接口关系和组合关系，我们根据模型和算法的特点进行数据流分析、面向对象分析。针对这些分析所得出的模型和算法关系特点，从而确定了仿真模型体系。

绪论中提到，原始数据生成仿真要生成高精度的伪距、载波相位、多普勒频移和信噪比这些观测数据仿真值，这需要全面考虑在真实导航系统中影响这些观测量的因素[26-28]。目前的全球卫星导航系统存在以下四种系统误差：由于星载原子钟误差和轨道的不确定性引起的卫星误差；大气引起的误差，即卫星信号在穿过大气层时的延迟(电离层、对流层)；接收机引起的误差，由于接收机本身技术问题造成的系统误差；地面误差，由于多径效应和高大物体遮蔽引起的误差。系统误差是影响导航信号观测量的主要因素。除了系统误差，要生成高精度的观测量仿真值，还要考虑星地几何距离、发射增益、大气衰减等因素。这些对观测量产生影响的因素分布在地球外空间、大气层以及地球表面，我们可以将这些因素大致划分为三个部分：包括太阳、卫星在内的空间段部分，包括自由空间、对流层、电离层在内的环境段部分以及包括地面站在内的地面段部分。下面的 3.1 我们依照空间段、环境段、地面段的顺序分析这些因素，建立相应的数学模型和算法体系，并分析这些模型和算法之间的关系。

3.1 模型和算法体系

3.1.1 空间段模型

计算伪距、载波相位和多普勒频移，需要知道卫星的轨道信息，即卫星的位置和速度。在空间段，星座的布局决定了卫星的轨道根数等几何分布信息；各种摄动力，包括太阳引起的摄动力、大气阻力以及地球潮汐等决定了卫星的运动不是简单的二体运动；而卫星模型本身又可以分解为推进子系统、轨道计算和星载原子钟等子模型；同时，卫星在运行过程中可能会发生星载原子钟异常、偶然的外部摄动等事件，我们称之为关键事件。因此，在空间段，我们主要考虑到各种对卫星轨道有关系的模型，这些模型主要有：天体模型(包括太阳、月球等行星模

型)、导航星座模型、卫星模型、星载原子钟模型、各种摄动力模型以及空间段的关键事件(Feared Event)等。

(1)天体模型

根据天体力学，任何天体的运动都是天体质量彼此相互吸引的结果。导航卫星载地球引力场内运动，除了地球引力外，还受许多天体的吸引，如太阳、月球等，这些行星不同程度的对卫星运动产生影响。因此，在原始数据生成的仿真试验中，我们需要考虑这些天体模型：太阳模型、地球模型和月球模型，在精度要求更高的的轨道计算中，还需要考虑包括火星、木星在内的更多的天体模型。这些天体对卫星产生摄动，摄动力的大小由天体与卫星之间的距离决定，因而在计算卫星所受摄动力时需要了解天体的位置信息。天体与卫星的这种关系，可以通过下面即将提到的摄动力模型间接发生。从而，天体模型只与天体引起的摄动力模型有直接关系，并向这些摄动力模型提供天体的位置信息。太阳、月球等行星的位置可以由美国喷气推进实验室的 JPL-405 行星历书[29]计算得出。

(2)摄动力模型

人造地球卫星在轨运动是地球引力和其它多种力共同作用的结果[30]，地球引力是最主要的引力场源。此外，太阳、月球、金星、木星等多个天体对卫星有摄动作用，其中最主要的是太阳和月球。地球内部质量分布不均匀，地球形状实际上并不规则，并具有一定的粘性，在日月等星体引力作用下，地球上会发生各种潮汐现象，因此，导航卫星还会受到地球固体潮、海潮、大气潮等摄动。由于广义相对论效应，使得卫星在以地心为原点的坐标系中的运动方程不同于仅考虑牛顿定律时的受力状况，这一差异导致卫星受相对论摄动。同时，卫星还会受到太阳光照，产生太阳辐射光压。地球反射太阳光压，对卫星也产生辐射压。由于大气具有一定的密度，大气与卫星相互作用，卫星就会受到大气阻力和浮力作用。为讨论方便而又不失一般性，本论文建立的模型框架所考虑的对卫星运动产生影响的摄动力有：地球引力、日月的第三体摄动、太阳辐射光压、相对论效应。在更为精确的卫星轨道计算中，其它摄动力模型可以参照上述摄动力模型加入到模型框架中。这里的日月摄动力模型、太阳辐射光压模型与太阳模型、地球模型、月球模型有直接联系：

表 3.1 天体模型与摄动力模型关系

天体模型	摄动力模型	关系
太阳模型	太阳摄动力模型、太阳光压模型	太阳模型计算太阳位置，用于太阳摄动模型、太阳光压模型的计算
月球模型	月球摄动力模型	月球模型计算月球位置，用于月球摄动模型计算

(3)星座模型

卫星星座决定了卫星的几何分布，包括卫星总数、轨道平面数、轨道倾角等[31]。导航星座常用的性能指标有：共视性要求、精度因子 DOP、星座值 CV 和可用性等。在原始数据生成仿真试验中，我们只关心星座对卫星轨道的决定作用。星座可以基于三种方法创建：

- 标准方法，添加预先定义导航星座，如 Galileo、GPS、GLONASS 等星座；
- 用户定义，分别设定每颗卫星的轨道根数，或者通过读取尤马(Yuma)历书文件数据进行设定；
- 基于 Walker 参数定义，该方法基于一个种子卫星创建，星座的其它卫星则在不同的轨道面展开，每个轨道面的卫星数相等，卫星间隔相同。种子卫星通过开普勒轨道根数设定，Walker 参数是指：轨道平面数、相邻轨道相位差、每个轨道平面卫星数和右旋升交点赤经分布。其它卫星可以通过种子卫星的开普勒轨道根数和 Walker 参数计算得出。

(4)卫星模型

面向导航领域，导航星座由多颗不同轨道的导航卫星组成，卫星模型是原始数据生成模型体系中最重要的实体模型。导航卫星的主要任务是发射各种精确定时的导航信号，这些信号必须以导航数据的形式嵌入各载频，其中包括精密卫星时钟时间和卫星位置[32]。在仿真试验中，我们将卫星建模为具有这些属性：开普勒轨道根数、PRN 号、卫星编号、视准轴、遮蔽角、历元时间、卫星的健康状态等。卫星模型用于为其它模型和算法提供卫星信息，如时间、位置等。卫星模型本身不执行，而把一些计算模块单独考虑，作为卫星模型的子模块。这些计算模块包括星载原子钟模型、轨道计算模块以及通信模块。

(5)轨道计算

卫星的位置与速度是一个整体，其计算需要考虑到作用于卫星上的各种摄动力。如前面所述，卫星在轨运动受到许多摄动力作用，其运动方程是一个二阶微分方程，通常运动方程的解法有分析法和数值法两种[33,34]。数值解法计算公式简单，精度高，在卫星精密轨道计算中得到普遍应用。在这里，对于轨道计算方法的使用根据要求达到的计算精度来决定。如果计算精度为 4 阶以下，可以采用 Runge-Kutta 积分；而对于更高的八阶以内的精度，则采用 Adams-Moulton 积分。

另外，也可以采用精密星历 SP3 文件，通过插值计算生成卫星轨道数据。

(6)星载原子钟模型

现代卫星导航的基础是将一部非常稳定的时间基准放置在能使用户得到最大限度利用的地方。以电磁信号在大气层中的传播速度，用户测量所产生的 1ns 的信号相位不确定性大致相当于 1/3m。能够将这种相位不确定性维持在导航系统要求范围内的唯一仪器是原子钟。导航卫星上携带的原子时钟，为导航信号提供精确、稳定和可靠的定时。在原始数据生成仿真试验中，星载原子钟与地面站原子时钟

类似，原子钟模型主要提供与系统参考时间的偏移量，即钟差。Allan 方差表征了原子钟的特性。原子钟的仿真时间算法是：

$$x_j = t_0 + \sum_{h=1}^{j} \Delta t + b + \left[\sum_{h=1}^{j} d \cdot \Delta t\right] + \left[\sum_{h=1}^{j}\left\{\sum_{g=1}^{h} a \cdot \Delta t\right\} \cdot \Delta t\right] + \sigma \cdot \left[\sum_{h=1}^{j} ran_h \cdot \Delta t\right] \tag{3-1}$$

其中，t_0 是初始真实时间，Δt 是推进步长，b 是初始钟差，d 是初始时钟漂移，a 是频率漂移，ran_h 是服从正态分布的随机数，数学期望是 μ_{ran}，标准方差是 σ_{ran}，σ 是比例因子，其算法是：

$$\sigma(\sigma_A, \tau, \Delta t, a, \sigma_{ran}, \mu_{ran}) \approx \left[\frac{\tau}{\Delta t}\left\{\frac{\sigma_A^2 - \frac{a^2 \cdot \tau^2}{2}}{\sigma_{ran}^2}\right\}\right]^{1/2} \tag{3-2}$$

这里的 τ 是 Allan 方差取样周期，σ_A 是 Allan 标准方差。

另外，钟差数据可以采用钟差数据文件，通过插值的方法计算得出得到，因此星载原子钟模型应该提供读取钟差数据文件的方法。

(6)关键事件

空间段的关键事件用于模拟在空间段可能发生的系统非正常运行事件。这些关键事件包括：伪轨道漂移事件(Pseudo Orbit Drift Feared Event, POrbitDriftFE)、外部摄动(External Disturbance Feared Event, ExDisturbanceFE)、非正常时钟行为(Abnormal Satellite Clock Behavior, AbnClockFE)。

外部摄动也称为卫星轨道错误，是指卫星由于非计划的轨道机动离开轨道平面的行为。该摄动由作用于卫星的摄动力触发，用户可以配置力的开始时间，持续时间，力的大小和力的扭矩。

伪轨道漂移事件，指在某段时间内用于计算伪距的卫星位置与仿真的卫星位置存在连续的偏差，从而对伪距的计算结果产生影响，但该故障事件不会对仿真的轨道预报和卫星位置本身产生影响。

非正常时钟行为，指在可预知的卫星钟差的基础上，添加卫星时钟非正常行为的仿真，这些非正常时钟行为通过配置非正常偏差、时钟漂移和时钟漂移率实现。

3.1.2 环境段模型

信号从卫星发射到地面站接收，由于环境对信号传播的影响，会造成信号传播的延迟。根据延迟来源不同，我们把这些环境因素划分为自由空间、电离层和对流层，这些延迟使得信号传播的速度和相位发生变化。信号在环境段传播总的延迟是这些延迟的求和。

(1)自由空间延迟模型

自由空间延迟模型是对信号由卫星到接收机之间传播经过自由空间引起的延迟进行仿真，包括 Sagnac 效应和轨道偏心效应的相对论改正，而这些相对论改正可以根据计算的需要可选。自由空间延迟是理想自由空间延迟、轨道偏心效应和 Sagnac 效应的求和：

$$\Delta t_{freeSpace} = \Delta t_{freeSpace,ideal} + \Delta t_{orbitEcc} + \Delta t_{sagnac} \tag{3-3}$$

为确保测量发生在相同的时间间隔上，需把自由空间延迟转换为共同的接收时刻，一般采用改正的自由空间延迟算法：

$$\begin{aligned}\Delta t_{freeSpace,corr} &= \Delta t_{freeSpace} + (\Delta t_{freeSpace} + \Delta t_{clockdrift,rx}) \cdot \frac{\vec{v}_r \cdot \vec{a}}{c} \\ &= (\Delta t_{freeSpace,ideal} + \Delta t_{orbitEcc} + \Delta t_{sagnac}) + ((\Delta t_{freeSpace,ideal} + \Delta t_{orbitEcc} + \\ &\Delta t_{sagnac}) + \Delta t_{clockdrift,rx}) \cdot \frac{\vec{v}_r \cdot \vec{a}}{c}\end{aligned} \tag{3-4}$$

这里，$\Delta t_{clockdrift,rx}$为接收机钟差，$\vec{v}_r$为卫星和接收机的速度差，$\vec{a}$是接收机到卫星的视线单位向量，c 为光速。

$\Delta t_{freespace,ideal} = \frac{r_{true}}{c}$为理想自由空间延迟，这里的$r_{true}$为星地几个距离，由卫星和地面站位置计算得出。

$\Delta t_{orbitEcc} = \frac{2(\vec{r} \cdot \vec{v})}{c^2}$为轨道偏心效应，是由偏近点角变化引起的一种相对论性效应，它是对自由空间延迟的一个修正，式中$\vec{r}$为卫星在 ECEF(地心地球固连坐标系)中的空间矢量，$\vec{v}$是卫星速度矢量。

$\Delta t_{sagnac} = \frac{2\vec{\Omega}_e}{c^2} \cdot \frac{\vec{r}_A \times \vec{r}_B}{2}$是 Sagnac 效应，是一种相对论性效应,用于对自由空间延迟的一个修正，$\vec{\Omega}_e$是地球角速度，$\vec{r}_A$是在 ECEF 坐标系中卫星位置矢量，$\vec{r}_B$是在 ECEF 坐标系中接收机位置。

(2)电离层延迟模型

高于大约 50km 的大气层称为电离层。电离层是一种微弱的电离气体，它能以各种方式影响电波传播。按电子密度的不同，电离层又分为 D、E、F1、F2 和 H 层，各层对导航信号的影响是不同的。电离层具有三大特性：

- 散射性，即其折射系数是电波频率的函数；
- 互补性，即伪距观测值和载波相位观测值的电离层延迟数值相同、符号相反；
- 变化大，在一天之内至少变化一个数量级，而且很难模拟。

由于这些特性，对于高精度的卫星定位来说，电离层是距离和距离变化率的重要误差来源。在天顶方向的电离层距离误差变化范围小则几米，大可达几十米。关于电离层对导航信号的延迟，目前已经建立起了多种模型，如 Klobuchar 模型、

IRI 模型和 Nequick 模型等[35-36]，在原始数据生成仿真试验中，我们采用精度较高的 Nequick 模型。

电离层对导航信号产生主要影响的参数是电离层的电子总数(TEC)，由一个横截面积为 $1m^2$ 的垂直柱体内的电子数目来表示，此柱体从卫星到观测者。电离层延迟 Δt_{iono} 是总电子数 TEC 的函数，定义如下：

$$\Delta t_{iono} = \frac{a \cdot TEC}{f^2} \tag{3-5}$$

这里，a=40.3e-9，是电离层延迟系数，f 是导航信号的频率，TEC 是总电子数。总电子数 TEC 通过对电子密度进行积分得到。目前，对于电子密度求法，已经建立了多种经验模型。在原始数据生成中，采用 ITU-R 的 Nequick 模型。该模型已有成熟的 fortran 代码实现。该模型通过输入用户位置、时间和卫星位置等参数获得电子总数。

总电子数 TEC 也可以采用 GPS 观测数据，通过插值的方法计算得到。

(3)对流层延迟模型

导航信号经过大气层传播到地面接收机时，会受到对流层的影响。对流层对导航信号的延迟从天顶方向的2m到地平高度角15°的25m延迟效应通常在2-25m，即使通过模型修正，也只能修正其中的 90%～95%，所以对流层延迟的改正精度一直制约着导航定位精度的提高。大气层是由干燥气体和水蒸气组成的，大气层的干湿两种成分对导航信号的传播延迟有着极不同的影响。

目前已经建立了计算对流层延迟的多种模型[37-38]，在这里我们采用基于 Hopefield 模型的对流层延迟算法：

$$\Delta t_{totalTropro} = \Delta t_{dryTropo} m_{dry}(E) + \Delta t_{wetTropo} m_{wet}(E) + (\Delta t_{noise} + \sigma_{noise} \cdot RAN) \tag{3-6}$$

其中 $\Delta t_{dryTropo}$ 和 $\Delta t_{wetTropo}$ 分别是对流层延迟的干项和湿项，

$m_{dry}(E) = \frac{1}{Sin\sqrt{E^2 + 6.25}}$ 和 $m_{wet}(E) = \frac{1}{Sin\sqrt{E^2 + 2.25}}$ 分别是干项和湿项对于某个仰角的延迟映射函数。噪声平均偏差 Δt_{noise} 和标准偏差 σ_{noise} 表征了由于气象条件的不确定性引起的误差，这两个参数都可以由用户设置的关键事件改写。随机数 RAN 是与时间相关的高斯函数。而对流层延迟的干项为：

$$\Delta t_{dryTropo} = 10^{-6} N_{d,0}^{Trop} \frac{1}{5h_d^4}[(h_d - h_r)^5] \tag{3-7}$$

其中 $N_d^{Trop} = 77.604\left(\frac{P_d}{T}\right)Z_d^{-1}$ 为干项折射率，逆向压缩率

$Z_d^{-1} = 1 + P_d[57.97 \times 10^{-8}(1 + 0.52/T) - 9.4611 \times 10^{-4}(T_c/T^2)]$。对流层延迟的湿项为：

$$\Delta t_{wetTropo} = 10^{-6} N_{w,0}^{Trop} \frac{1}{5h_w^4}[(h_w - h_r)^5] \tag{3-8}$$

其中 $N_w^{Trop}=64.79\left(\frac{e}{T}\right)Z_w^{-1}+377600\left(\frac{e}{T^2}\right)Z_w^{-1}$，逆向压缩率

$$Z_w^{-1}=1+1650\left(\frac{e}{T^3}\right)\left[1-0.01317T_c+1.75\times10^{-4}T_c^2+1.44\times10^{-6}T_c^3\right]$$

在以上各式中，e为接收机位置的水蒸气气压，p_d为干燥大气气压，T_c为接收机温度(摄氏度)，T为接收机温度(开氏度)，h_w为湿润大气高度，h_r为接收机高程，h_d为干燥大气高度，这些对流层参数可以通过观测数据获得。

(4)总的环境延迟

导航信号在环境段传播依次经过自由空间、电离层和对流层，其总的环境延迟为这三项延迟之和：

$$\Delta t_{totalEnv}=\Delta t_{freeSpace}+\Delta t_{iono}+\Delta_{totalTropo} \tag{3-9}$$

(5)关键事件

在环境段，由于电离层和和对流层变化的随机性强，在某些复杂气象或者其它条件下，式(3-5)和式(3-8)计算得出电离层延迟和对流层延迟可能会与真实数据差别较大。因此我们定义了导航信号在电离层和对流层传播过程中可能会发生的关键事件，用来模拟这些特殊情况。

电离层的关键事件(Ionospheric Feared Event, IonosphericFE)导致所有卫星导航信号产生过多的电离层延迟和相位周期数变化，对于过多的电离层延迟，可以通过添加一个与时间相关的正态分布实现，而对于相位周期数的变化，则直接增加载波相位的周期数。

对流层的关键事件(Tropospheric Feared Event, TroposphericFE)导致信号在对流层传播的延迟增加，而载波相位并没有发生变化。通过增加一个与时间相关的服从正态分布的延迟来实现。

3.1.3 地面段模型

在地面段，主要考虑分布在全球各地的传感器站，也称地面站，每个地面站点配备一只接收机和原子钟（也称站钟）。地面站的主要功能在于，接收导航信号并根据信号信息计算伪距、载波相位、多普勒频移和信噪比，并写入到 RINEX 文件中。对导航信号的接收和观测量的计算相关的，需要考虑地面站原子钟的钟差、信号传播到地面站的多径效应和接收过程中的噪声等因素。与这些因素对应，我们在地面段可以建立地面站模型、多径模型、地面站原子钟模型，并实现伪距、载波相位、多普勒频移和信噪比这些观测量的算法。

(1)地面站模型

在导航系统中，地面段主要由监视和控制导航卫星运行的监测站、注入站和运控中心构成[39]。在面向原始数据生成仿真领域，地面站用于接收可见卫星的导航信号，计算观测量的仿真值。为便于分析和计算，我们把接收机和地面站原子

钟作为地面站模型的子模型组件，观测量的计算也作为地面站模型的算法。在环境段模型计算过程中要用到地面站的位置和遮蔽角等信息，对这些模型而言，地面站模型主要功能是为它们提供计算所需的属性；同时，地面站模型计算观测量的仿真值需要用到环境段模型的计算结果。地面站模型需要为其它模型和算法提供的属性主要有：站点位置、遮蔽角、接收机温度、接收机类型、健康状况等参数。

(2)地面站原子钟模型

地面站原子钟模型为其它模型和算法提供仿真时刻的钟差数据，其算法与星载原子钟模型相同，它们的差别在于初始钟差、时钟漂移、时钟漂移率和 Allan 方差这些原子钟参数的不同。一般而言，地面站原子钟比星载原子钟具有更高的精度和稳定性。

与星载原子钟类似，地面站原子钟的钟差数据也可以通过读取外部数据文件获得，因此，地面站原子钟模型也应该提供获取外部钟差数据文件的方法。

(3)多径模型

多路径效应亦称多路径误差，是指接收机天线除直接受到卫星发射得信号外，还可能受到天线周围地物一次或多次反射得卫星信号，信号叠加将会引起测量参考点(相位中心点)位置的变化，从而使观测量产生误差，而且这种误差随天线周围反射面的性质而异，难以控制[40]。在星基精密引导系统中，多径效应是主要的误差源之一，对伪距观测值、载波相位观测值和多普勒频移都造成了误差。

由多径效应引起的测距误差 E_{mr}、相位误差 E_{mp} 和多普勒频移误差 E_{mf} 的算法如下：

$$E_{mr} = A_{br} + A_r \cdot K_{env} \cdot K_{rec} \cdot K_{ran} \cdot \cos(\omega \cdot Elev) \tag{3-10}$$

$$E_{mp} = A_{bp} + A_p \cdot K_{env} \cdot K_{rec} K_{ran} \cdot \cos(\omega \cdot Elev) \tag{3-11}$$

式中，A_r，A_p 和 A_f 是多路径效应的振幅；A_{br}，A_{bp} 和 A_{bf} 是多路径的剩余偏差项；K_{rec} 是接收机的灵敏度因子；ω 多路径的频率；Elev 卫星仰角；K_{env} 环境特征系数；K_{ran} 是时间相关的高斯分布。

由上述公式可以看出，多径误差的计算需要用到卫星仰角，从而需要获取卫星和地面站的位置信息。

(4)信噪比计算

信号由卫星发射到地面接收，有多种导致信号功率损耗的因素。这些损耗主要包括：自由空间损耗、环境损耗（主要是对流层）、地面站周围遮蔽损耗、树叶遮蔽损耗、线性极化损耗、指向偏离损耗、发射增益和接收增益等。导航信号的受到的干扰主要有导航系统内部的干扰，即同一个接收机受到同一个导航系统(如 GPS)内不同卫星的干扰；系统间干扰，即同一个接收机受到不同导航系统的卫星的干扰，例如在接收 GPS 卫星信号时受到 GLONASS 卫星信号的干扰。

● 接收功率计算

接收机接收到的信号功率是发射功率 P_T、自由空间损耗 P_{FSL}、大气(对流层) 损耗 P_{Tro}、地形遮蔽损耗 P_{RS}、植物遮蔽损耗 P_{FS}、线性极化损耗 P_{LP}、分离损耗 P_{DP}、发射增益 P_{TG} 和接收增益 P_{RG} 的求和，即

$$P_R = P_T - P_{FSL} - P_{Tro} - P_{RS} - P_{FS} - P_{LP} - P_{DP} + P_{TG} + P_{RG} \tag{3-12}$$

● 总的干扰－接收功率比计算

总的干扰功率比定义为接收功率和总的干扰信号功率的比值，即

$$j/s = j_r/s_r \text{ 或者 } (J/S)_{dB} = (J_r)_{dBw} - (S_r)_{dBw} \tag{3-13}$$

s_r 为上述的接收功率，而干扰信号的功率 j_r 为系统间干扰 I_{intra} 和系统内干扰 I_{inter} 之和：

$$j_r = I_{intra} + I_{inter} \text{ 或者 } (J_r)_{dBw} = 10 \cdot Log(I_{intra} + I_{inter}) \tag{3-14}$$

式中，系统间干扰 $I_{Intra} = \sum_{i=1}^{N_{VS}} 10^{(P_{R,i}+c_i)/10}$，系统内干扰 $I_{Inter} = \sum_{i=1}^{N_{VS}} 10^{(P_{R,i}+c_i)/10}$，

其中 N_{VS} 是可见干扰卫星数，$P_{R,j}$ 是同一卫星导航系统内的干扰信号功率，c_i 是干扰系数，是接收机与卫星之间的多普勒的函数。

● 载波噪声比

载波噪声应当足够低，使得接收机能以较低的噪声带宽进行跟踪(GPS 的噪声带宽为 10Hz)。导航信号的载波与噪声之比，源于热噪声及其内部信号损耗，其算法如下：

$$(C/N_0)_{dB} = (S_r + G_d) - 10 \cdot \log(k \cdot T) - A \tag{3-15}$$

这里，T 是用户可配置接收机温度，S_r 是接收到的载波信号功率，G_d 是天线增益，k 是波耳兹曼常数，A 是综合损耗，为用户配置参数。

● 总的信噪比计算

总的信噪比算法如下：

$$c/n_t = 1/\left(\frac{1}{c/n_0} + \frac{1}{c/n_u} + \frac{j_r}{s_r}\right) \tag{3-16}$$

$$\text{或者 } (C/N_t)_{dB} = -10 \cdot \log\left(10^{-\frac{C/N_0}{10}} + 10^{\frac{J/S}{10}} + 10^{-\frac{C/N_u}{10}}\right) \tag{3-17}$$

式中的 C/N_u 表示由于各种信号传播衰减因素导致的噪声，是用户可设置的参数。

(5)伪距计算

伪距定义为信号由发射到接收的时间差与光速的积，这里的时间差包括信号在环境段所用的时间、测距误差的时间和钟差，伪距算法如下所示：

$$r_{pseudo} = (\Delta t_{totalEnv} + \Delta t_{rangeError,m} + \Delta t_{clockDrift,m}) \cdot c \tag{3-18}$$

这里，$\Delta t_{totalEnv}$ 是总的环境延迟，$\Delta t_{rangeError,m}$ 是测距误差引起的延迟，$\Delta t_{clockDrift,m}$ 钟差引起的延迟，c 为光速。

测距误差由频率间干扰(群延迟)Δt_{IFB}、热噪声误差 Δr_{N_0}、多径误差 Δr_{mr} 组成，即

$$\Delta t_{rangeError,m} = \Delta t_{IFB} + (\Delta r_{N_0} + \Delta r_{mr})/c \tag{3-19}$$

其中接收机 i 与卫星 j 之间的群延迟为：

$$\Delta t_{IFB} = G \cdot (d_i^{\{f_m\}} + d_{\{f_m\}}^j) \tag{3-20}$$

式中，G 是频率干扰偏差的修正因子，取决于接收机类型；$d_i^{\{f_m\}}$ 是接收机 i 对于所有导航信号频率$\{f_m\}$的干扰偏差；$d_{\{f_m\}}^j$ 是卫星 j 对于所有导航信号频率$\{f_m\}$的干扰偏差。

(6)载波相位计算

导航信号在传播过程中，在环境段和接收过程中都会发生相位的变化，载波相位观测值的算法是：

$$\Phi_m = \Phi_{env} - \Phi_{offset} + \Phi_{cycleslip} + E_{phase} + f_{carrier} \cdot T_{drift} \tag{3-21}$$

式中，Φ_{env} 为由环境段延迟得出的真载波相位，Φ_{offset} 是初始相位偏差相位的偏差，$\Phi_{cycleslip}$ 是用户设置的关键事件产生的相位移动周期数，$f_{carrier}$ 和 $\lambda_{carrier}$ 分别是载波的频率和波长，T_{drift} 时钟漂移，而 Φ_{env} 是环境段计算得出的传播延迟与光速 c 的乘积：

$$\Phi_{env} = (\Delta t_{freeSpace,corr} - \Delta t_{iono} + \Delta t_{totalTropo}) \cdot c \tag{3-22}$$

相位测量误差 $E_{phase} = E_{mp}/\lambda_{carrier} + E_p$，$E_{mp}$ 是多径效应引起的相位误差，$\lambda_{carrier}$ 是载波的波长，而 E_p 是热噪声引起的相位差。

(7)多普勒频移计算

由于地面站接收信号时产生多径误差，此时多普勒频移算法是：

$$\Delta f = \frac{f_t}{c}\left[(\vec{v} - \vec{u}) \cdot \vec{a}\right] \tag{3-23}$$

这里，f_t 导航信号的发射频率，$\vec{v}$ 是卫星速度，$\vec{u}$ 是接收机的速度，$\vec{a}$ 是卫星和接收机之间的视线单位向量，c 是光速。

(8)关键事件

在地面段发生的关键事件主要有非正常时钟行为(Abnormal Clock Behavior, AbnClockFE)、过量的多径效应(Excessive Multipath Feared Event, ExMultipathFE)和接收机异常事件(Receiver Feared Event, RxFE)。

地面段的非正常时钟行为关键事件与空间段的非正常时钟行为关键事件原理完全相同。过量的多径效应是由信号的过多反射产生的，通过增大公式(3-10)、(3-11)中的系数达到增加多径延迟来表示由于信号的过多反射；接收机的关键事件包括：

过量干扰(Excessive Interference Feared Event, ExInterferenceFE)、残留信号噪声(Residual Signal Noise Feared Event, RsSigNoiseFE)和过量群延迟(Excessive Inter Frequency Bias Feared Event, ExInterFreBiasFE)，这些接收机关键事件都在接收机前端实现。

综上所述，原始数据生成模型和算法众多，下表列出了原始数据生成主要的模型和算法及简单说明：

表 3.2 主要模型和算法列表

空间段	天体模型	包括太阳、月球、地球模型，采用 JPL DE-405 计算星体位置和速度
	星座模型	星座的生成方法：标准方法、用户定义、基于 Walker
	卫星模型	用于说明卫星的参数，包括位置、速度、钟差等，用于其它模型和算法的计算
	摄动力模型	包括太阳摄动、月球摄动、地球固体潮汐、海洋潮汐、相对论改正、大气摄动、太阳光压、重力梯度扭矩等
	地球重力模型	使用 JGM-3 重力模型
	空间段关键事件	包括外部摄动、伪轨道漂移、非正常时钟行为
	星载原子钟模型	用于计算星载原子钟的钟差，可以通过观测数据文件计算钟差
	轨道计算	用于计算各种摄动力对卫星轨道的影响。低精度计算使用 Runge-Kutta 积分，高精度计算使用 Adams-Moulton 积分，也可以通过观测数据计算
环境段	自由空间延迟模型	包括理想自由空间延迟、轨道偏心效应和 Sagnac 效应模型
	电离层延迟模型	有多个可替代的模型，主要有 Nequick 模型、Klobuchar 模型和 IRI 模型等，可以通过观测数据计算电离层延迟
	对流层延迟模型	有多个可替代模型，主要有 Hopefield 模型、Saastamoinen 模型、Black 模型和 Eisner 模型等，也可以通过观测数据计算
	环境段关键事件	包括电离层关键事件、对流层关键事件
	地面站模型	用于说明地面站的位置、遮蔽角等属性，这些属性用于其它模型的计算参数
	接收机模型	用于说明接收机类型、位置、温度等属

地面段		性
	地面站原子钟模型	用于计算原子钟钟差，也可以通过观测数据计算
	多径效应模型	用于计算对伪距、载波相位的误差
	伪距计算	需要用到环境段的模型以及钟差、群延迟、热噪声误差、多径模型等算法和模型
	载波相位计算	同伪距计算
	多普勒频移计算	需要用到轨道计算的所得的卫星位置和速度
	信噪比计算	需要用到自由空间损耗、大气(对流层)损耗、地形遮蔽损耗、植物遮蔽损耗、线性极化损耗、分离损耗、发射增益和接收增益以及信号干扰、载波噪声比等模型和算法
	地面段关键事件	包括非正常时钟行为、过量多径效应、过量干扰、残留信号噪声、过量群延迟、过量多径效应

3.2 模型关系分析

在前一节中我们从原始数据生成仿真的背景出发，对影响导航信号观测量的因素进行了分析，从总体上描述了原始数据生成仿真的所涉及的领域，并给出主要的模型和算法。

然而无论从总体上还是到各个具体的模型和算法的描述，我们都无法给出适当的标准去建立仿真模型体系。因此，我们必须进行模型和算法之间关系的分析。这些分析包括实体模型分析、数据流分析和面向对象分析。

3.2.1 实体模型分析

在 2.4 节提到，实体模型在 SMP2 的仿真模型设计中具有重要作用。在原始数据生成仿真中，实体模型众多，为了便于对模型和算法的关系分析，我们采用了以物理实体为单位组织模型的方式，原始数据生成的实体模型组成如图 3.1 所示。其中地面段的实体模型包括：地面站模型、接收机模型、地面站原子钟模型、植物遮蔽模型和地形遮蔽模型；环境段的实体模型包括：自由空间模型、对流层模型、电离层模型；空间段的实体模型是：天体模型、星座模型、卫星模型、星载原子钟模型、太阳引力模型、地球引力模型、月球引力模型、太阳光压模型等摄动力模型也都可视作实体模型。

表 3.2 列出了原始数据生成模型体系中涉及的除坐标变换、时间转换以及一些数学算法(如矩阵运算等)之外的模型和算法。图 3.1 则从实体模型的角度分析了模

型体系中存在的主要关系。对于一般的仿真问题而言，我们可以按照实体模型建立仿真模型。然而在原始数据生成仿真中，按照图 3.1 中所列的实体模型建立仿真模型，然而这样的做法并不合适。主要有下面两点原因：

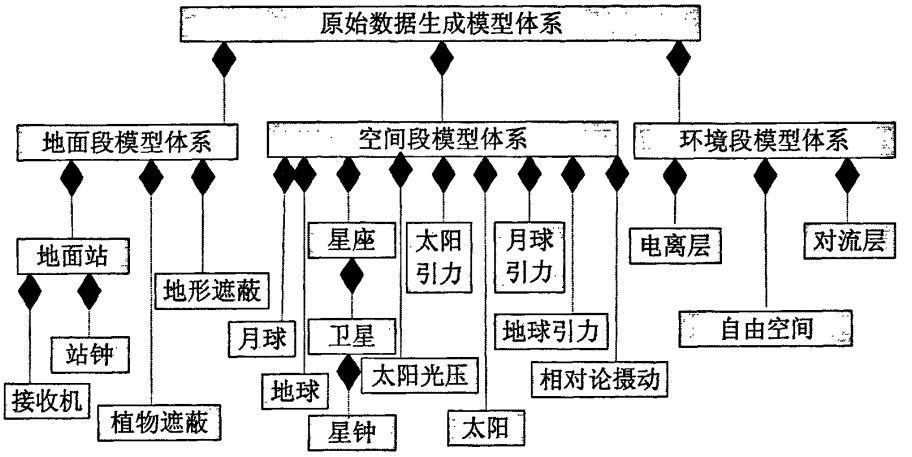

图 3.1 原始数据生成的实体模型

有些实体模型计算复杂，在实际计算中往往使用经验值，这些模型可以归结为某个模型的模块，比如植物遮蔽和地形遮蔽，它们可以作为地面站模型的一个算法；

从计算过程的角度看，某些非实体模型与实体模型具有一定的相似性，将其作为仿真模型更能减少模型集成的复杂程度，如多径模型和电离层模型等；

因此我们有必要进行更全面的分析，从模型的输入输出弄清模型的数据交互关系，便于我们认识模型的复杂程度也便于分析仿真模型之间的集成关系。下一节的数据流分析从总体上分析了模型的输入输出关系。

3.2.2 数据流分析

前面讲到，原始数据生成主要对伪距、载波相位、多普勒频移和信噪比这些观测数据进行仿真计算。因此，也就有必要围绕着这些观测数据仿真值的产生过程对原始数据生成的数据流进行分析。并且，在数据流分析的基础上分析各个模型和算法之间的接口关系。

模型接口分析也可视作模型集成设计，是总体建模人员在总体设计仿真中的主要工作之一。其目的是明确各子模型在仿真应用中的职责，明确各子模型与其它子模型之间的交互关系以及交互接口信息项的含义，以便后续阶段的配置式模型集成。

根据前面对原始数据生成算法的描述，下面的几张图给出了原始数据生成计算过程的数据流图。

根据多普勒频移的算法公式(3-23)，我们得出了其计算过程数据流图 3.2。其中的“卫星速度”和“卫星位置”对应前面的的模型和算法分析中的卫星模型，

而“卫星速度计算”和“卫星位置计算”对应轨道计算；根据伪距的计算公式(3-18)~(3-20)，以及这些公式中所包含的算法，我们可以得到伪距计算的数据流图3.3；根据载波相位的计算公式(3-21)、(3-22)以及这些公式中包含的算法，我们可以得到载波相位的计算过程数据流图 3.4；根据信噪比的计算公式(3-12)~(3-17)我们得出了信噪比计算过程的数据流图3.5。由于数据流图比较复杂，我们没有给出具体的数据流说明，而是从总体上把握数据流向，从而发现其间模型、算法之间的复杂关系。

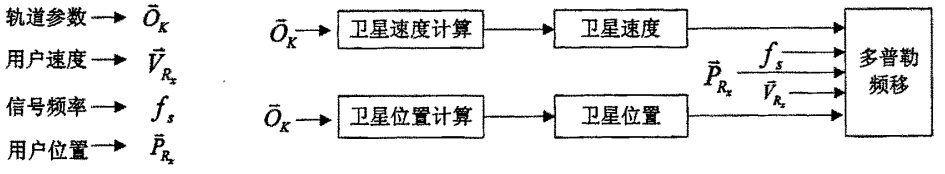

图3.2 多普勒频移计算数据流示意图

从图3.3~3.5可以看到以下事实：

(1)卫星速度和位置计算，也就是卫星轨道计算，是原始数据生成仿真试验中的重要计算过程，由前面模型和算法体系中所述，轨道计算根据任务需要可选择多种算法。

(2)多个算法在计算过程中同时需要卫星和地面站的位置信息，这些算法有星地几何距离计算、卫星仰角计算，而自由空间延迟、电离层延迟、对流层延迟、多径效应以及发射增益、接收增益等模型和算法则以星地几何距离和卫星仰角作为参数。可以考虑卫星模型、地面站模型提供获取位置的接口，使得自由空间延迟、电离层、对流层和多径模型等模型可以获取位置信息完成计算。另外，发射增益、接收增益、自由空间损耗和线性计划等这些算法较为简单而且固定，可以考虑将其作为地面站的算法，为避免接口的复杂性，这些算法所需的卫星信息(仰角、星地几何距离)可以通过同时使用地面站模型和卫星模型的接口的其它模型(如电离层模型)传递。多径效应模型虽然算法不是很复杂，但由于该模型要实现过量多径效应这个关键事件，我们仍要将其作为单独的模型看待。最后，星地几何距离和仰角的计算则可以在自由空间延迟、对流层延迟和电离层延迟这些模型中作为算法一部分实现。

通过在这一节中的数据流分析，我们基本明确了模型和算法之间的数据流关系、接口关系。在此基础上我们还需要通过面向对象分析了解模型之间的组合关系。

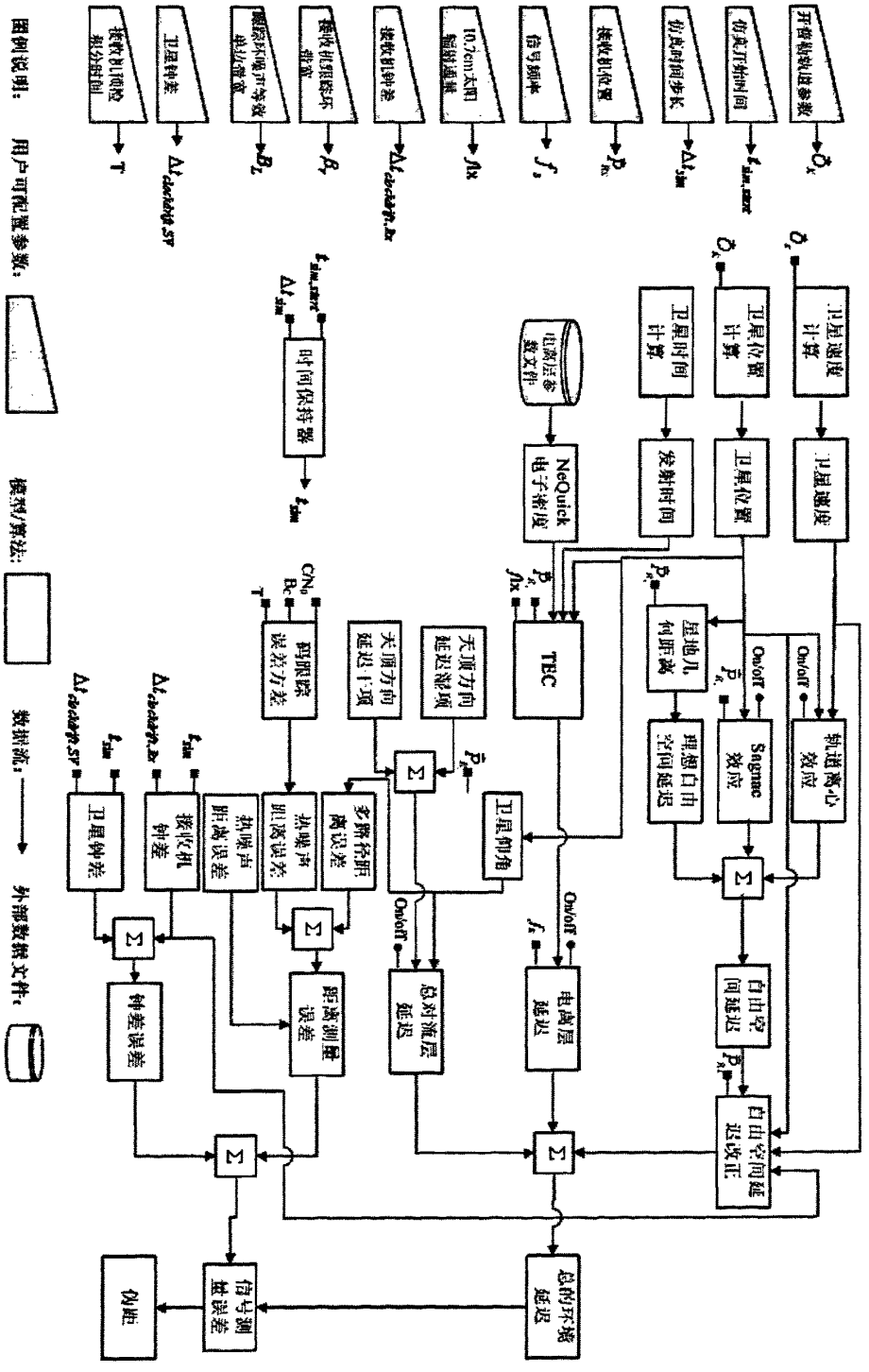

图 3.3 伪距计算数据流示意图

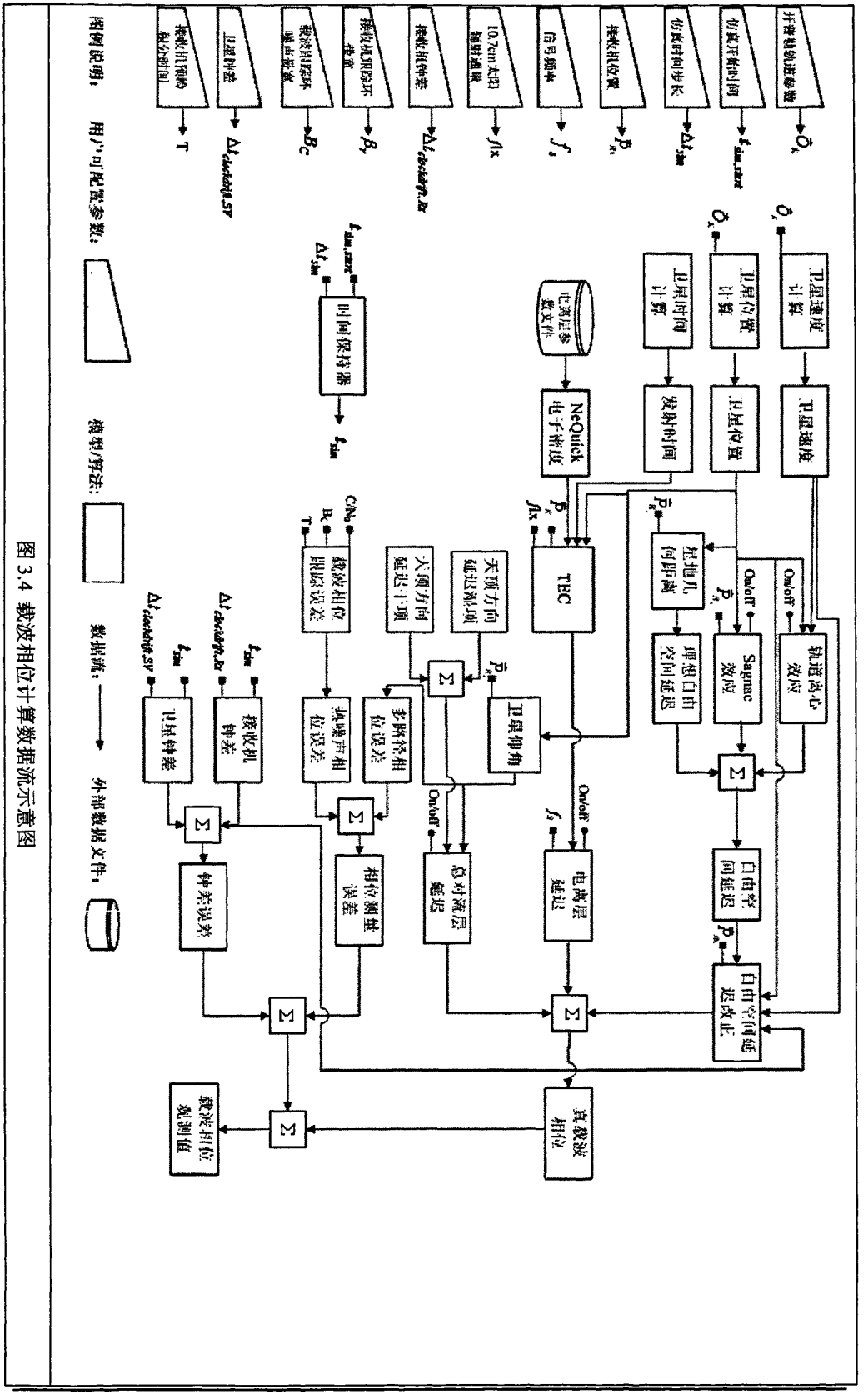

图 3.4 载波相位计算数据流示意图

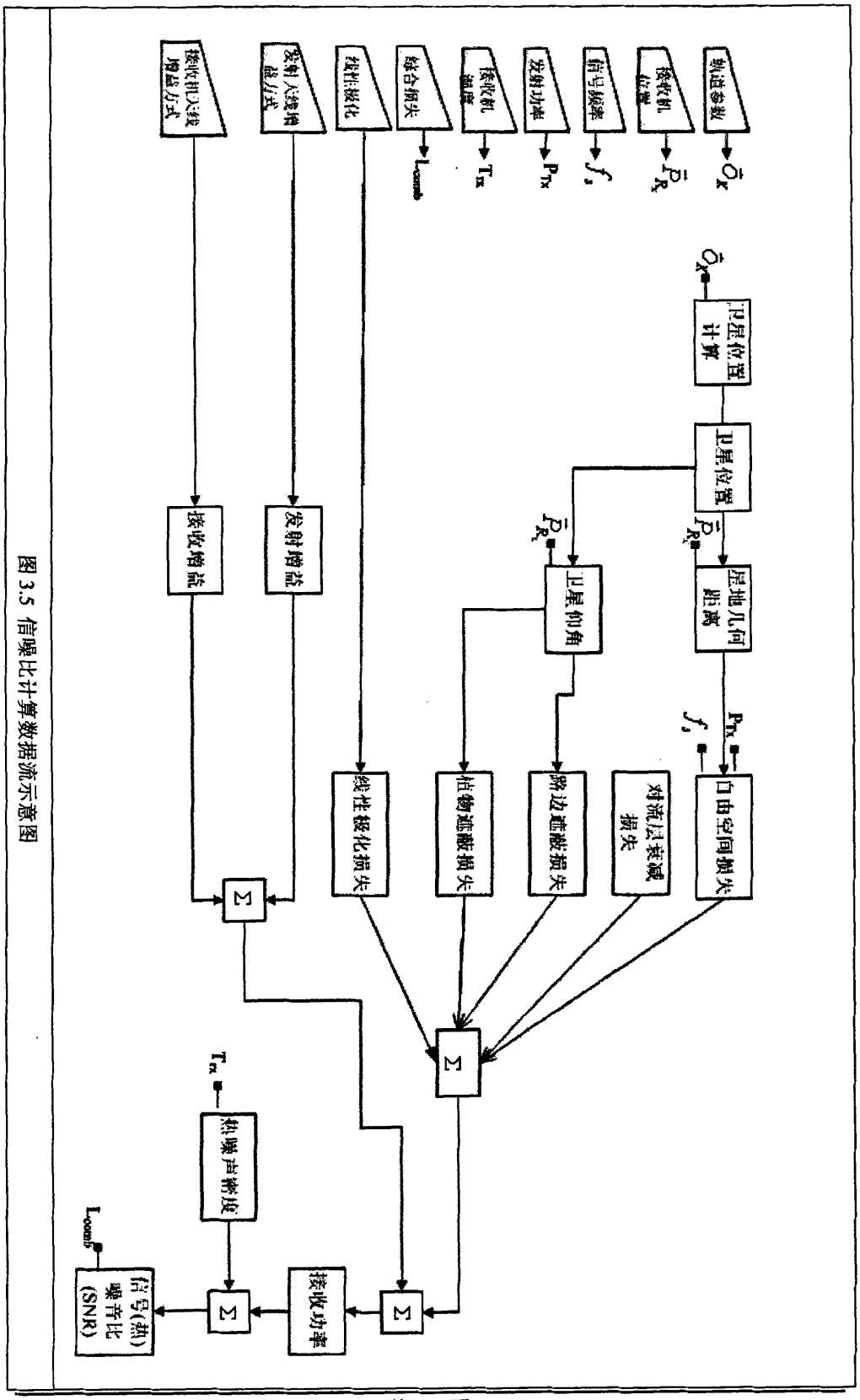

图 3.5 信噪比计算数据流示意图

3.2.3 面向对象分析

SMP 主要基于 MDA 的思想，从面向对象仿真的角度建立模型体系，便于用户按照与实际系统类似的方式建立想定、进行试验设置和数据分析，提高系统的集成性和可用性。在 GSSF 中，主要采用基于 SMP 的面向对象仿真集成导航基础模型框架。采用基于SMP的仿真模型集成，不同的导航模型计算模块只能作为SMP模型体系中对象模型的算法和方法。需要对导航系统进行面向对象分析，确定导航仿真基础模型体系框架。根据以上模型和算法的分析，图 3.6 给出了一个原始数据生成仿真的基础模型体系框架示意图。

由于卫星导航仿真的复杂性，不能随意对导航领域仿真模型进行面向对象建模，应该依据导航领域特点确立卫星导航仿真系统的模型顶层框架。这里可以考虑以下模型分类原则：

- 依据用户想定，适应不同仿真任务需要；
- 依据空间关系划分模型；
- 依据模型之间耦合关系确定模型框架的顶层元素分解；
- 依据模型内部的详细程度细化不同的顶层元素分解；
- 依据模型之间的关系接口调整模型组成；
- 依据模型算法的共享程度确定模型的层次；
- 依据模型之间接口的分辨率、模型计算的强度确定模型框架、计算回路以及不同模型框架之间的关系；
- 导航专业模块按照顶层框架进行布局，根据顶层框架模型的需求确定不同任务的模型框架或共享框架，进而形成面向不同应用分析的想定框架；
- 根据应用中计算模块的共享程度确定所有模型模块的共性模块和基础算法。

这里在顶层框架上主要从空间的角度考虑系统构成，如顶层框架可以考虑由太阳 Sun、月球 EarthMoon、地球 Earth、星座 Constellation 模型组件构成的太阳系 SolarSystem 组合模型。其中太阳模型组件派生于星体类 CelestialBody，而行星 Planet、卫星 Moon 和人造卫星 Satellite 则派生于轨道星体 OrbitingCelestialBody 对象，行星又由卫星 Moon 和人造卫星 Satellite 组成的星座构成。其中地球派生于行星类，月亮派生于卫星类，星座又包含多个人造卫星。可以考虑将太阳系组合模型基于组件的集成，而太阳系内的太阳月亮、地球等模型作为该组件的一部分。

在导航领域，星座由多个不同轨道的导航卫星组成，星座模型和卫星模型构成组合模型。导航卫星星座构成了空间段，其中每个导航卫星模型组件由多个模型组件构成，如：数据管理系统、推进系统、能源系统、轨道计算、星载原子钟以及通讯和恒温等模型组件。在原始数据生成中，导航卫星模型组件由以下模型组件组成：

- 轨道 Orbit；
- 星载原子钟 SatClock；

高精度卫星轨迹生成算法、卫星钟差、星地几何距离、卫星仰角、导航信号发射增益等模型算法和模块分别对应不同模型组件的接口和算法。

环境段主要考虑环境影响的信号延迟，所以其中自由空间传播延迟 FreeSpaceDelay、电离层延迟 IonosphericDelay 和对流层延迟 TroposphericDelay 等不同的延迟计算均派生于环境的信号延迟模型组件 Delay。环境延迟由自由空间延迟、电离层延迟和对流层延迟组成。Sagnac 效应、轨道偏心效应以及理想空间延迟等算法和模块对应于自由空间延迟模型的接口和算法；在电离层模型中，用于计算总电子数(TEC)的 Nequick 模型以及其它替代算法和实测数据模型将作为电离层延迟模型的不同算法和接口；与此相似，对流层延迟模型的计算需要用到外部实测数据，对流层延迟模型也应当提供读取实测数据的接口。

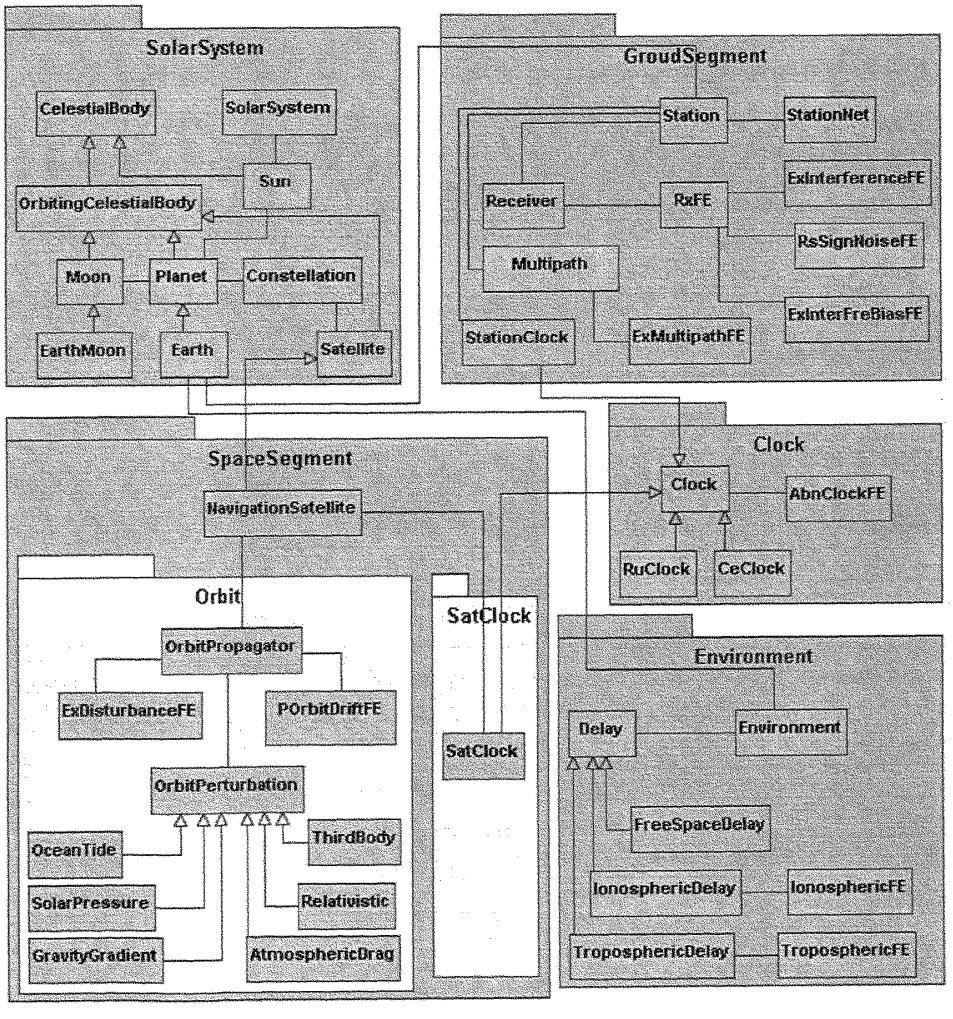

图 3.6 原始数据生成基础模型体系框架示意图

在导航系统的地面段主要由监视和控制导航卫星运行的监测站、注入站和运控中心构成，它们派生于表示和计算地球地理位置的地面站模型组件。在面向原始数据生成仿真领域，地面段主要指地面站 Station，分布在各地的地面站组成地面站网络 StationNet。地面站原子钟 StationClock、接收机 Receiver 作为地面站的模型组件，多径效应 Multipath 则与地面站关联。将伪距、载波相位、多普勒频移、信噪比这些观测量的计算及其所包含的子算法作为地面站模型的算法。这些算法所包含的子算法包括：热噪声误差、群延迟误差、地形遮蔽损耗、植物遮蔽损耗、自由空间损耗、对流层(大气) 损耗、接收增益、极化损耗、分离损耗、干扰/接收功率比、载波噪声比等。

关键事件可以根据仿真任务的需要触发或者关闭。关键事件的实现方式一般有两种，要么接收其它模型传递一组数据从而触发其计算，要么作为算法在其它模型中实现。我们把与关键事件相关联的模型称为实现模型。这些关键事件及其实现模型见下表：

表 3.3 关键事件及其实现模型

关键事件	相关模型和算法
伪轨道漂移(POrbitDriftFE)	轨道计算
外部扰动(ExDisturbanceFE)	轨道计算
非正常时钟行为(AbnClockFE)	时钟模型
电离层关键事件(IonosphericFE)	电离层延迟模型
对流层关键事件(TroposphericFE)	对流层延迟模型
过量多径效应(ExMultipath)	多径效应模型
过量干扰(ExInterferenceFE)	接收机模型
残留信号噪声(RsSignalNoiseFE)	接收机模型
过量群延迟(ExInterFreBiasFE)	接收机模型

在这些关键事件中，电离层关键事件和对流层关键事件相对独立，我们可以视其为单独的模型；而对于其它关键事件，则可以作为其实现模型中的一个算法。

3.3 基于 SMP2 的仿真模型体系

前文从实体模型、数据流和面向对象的角度对原始数据生成的模型和算法，由实体模型分析我们确定了以实体模型为基础的仿真模型系统；从数据流分析我们找出了数据流和接口复杂的模型；面向对象分析是对前文分析的补充，并且分析了模型之间的组合关系。综上所述，可以考虑将以下模型设计成 SMP 的仿真模型：

- 太阳系模型，包括太阳、月球、地球；
- 摄动力模型，包括日月的第三体摄动、太阳辐射光压、相对论改正和地球引力等；
- 星座模型；
- 卫星模型；
- 轨道计算；
- 星载原子钟；
- 环境段模型，包括自由空间延迟模型、电离层延迟模型和对流层延迟模型；
- 地面站模型，包括接收机模型和地面站原子钟；
- 多径效应；
- 关键事件模型，包括外部摄动、电离层关键事件和对流层关键事件；

基于 SMP2 的仿真模型开发与集成中，模型被看作组件，模型的实现是模型最重要的部分，因此，我们希望最大化这些模型实现的潜在重用性。前面提到，基于 SMP2 的仿真模型集成方法有 5 种：基于类的集成、基于接口的集成、基于组件的集成、基于事件的集成和基于数据流的集成，可以根据原始数据生成模型之间关系的特点，综合采用上述 5 种集成方法进行集成开发。而要选取适当的方法进行模型的组合集成，必须弄清楚模型之间的关系。前面我们已经讨论了原始数据生成模型和算法间的各种关系，在这里，我们将讨论上述划分的各个 SMP 仿真模型之间的集成关系。针对原始数据生成模型的特点，可以考虑以下原则：

- 对于物理上存在组合关系的仿真模型，为便于理解，可以考虑基于组件的集成，组合模型通过模型容器访问其子模型；
- 对于存在数据传输的模型，如果数据传输是可以选择关闭的，为减少实现过程中进行代码上的改动，采用基于数据流的集成；
- 对于数据传输类型多的模型，并且这些传输的数据是固定不变，不存在选择关闭性，考虑基于接口的集成，提供数据的模型提供这些接口的实现；
- 对于需要触发事件的模型，可以考虑基于事件的集成，接收事件的模型根据其自身设置条件、参数等决定是否执行相应的行为；
- 对于计算过程中需要相同接口的多个模型，为减少接口的复杂性，考虑建立新的模型管理这些接口，而这些使用接口的模型与新的模型基于组件集成并作为其子模型。

太阳模型计算太阳的位置和速度，太阳摄动模型和太阳光压模型则需要太阳模型计算出的太阳位置信息。为实现这一功能，有两种集成方法可以采用：基于接口的集成和基于数据流的集成。考虑到在不同仿真任务对摄动力模型的选择性，在 SMP2 中采用基于数据流集成其数据的传输可以在调度文档中完成，而无需改动代码。因此太阳模型与摄动力模型之间采用基于数据流的集成较为合适。

月球模型用于为月球摄动提供位置信息。与对太阳模型与其相关摄动力模型关系类似，月球模型与月球摄动模型之间应当采用基于数据流集成。

太阳摄动、太阳光压、地球引力、月球摄动、相对论摄动等摄动力模型用于为轨道积分提供摄动力，与太阳模型的分析类似，为便于根据仿真任务需要选择摄动力的类型，可以考虑摄动力模型与轨道计算之间基于数据流集成。这样在调整摄动力模型时，只需在调度文档中决定是否进行数据传输，而无需代码中作更改。。同时，轨道计算的结果也将作为摄动力模型下一个仿真时刻计算摄动力的参数，这个数据传输过程也可以基于数据流的集成。

轨道计算模型用于计算卫星的位置和速度，其计算结果将传送到卫星模型，当轨道计算完成时，计算所得数据将作为卫星的位置和速度数据。考虑到轨道是卫星的重要属性，可以将轨道计算模型作为卫星模型的子模型，采用基于组件的集成，卫星模型通过组合机制方位轨道计算模型。前面指出，伪轨道漂移和外部摄动这两个关键事件作为算法在轨道计算模型中实现，其中伪轨道漂移关键事件并不影响后面的轨道计算，为便于实现关键事件，传送给摄动力和卫星模型的卫星轨道和速度信息应该由不同的字段保存和传送。

从卫星的组成情况看，星载原子钟是卫星的组成部分，因此在 SMP2 的仿真模型中，二者应当基于组件的集成，卫星模型通过组合机制(模型容器)访问星载原子钟以便获取钟差数据。

卫星模型为其它模型提供导航信号发射时间、卫星位置和速度以及由位置决定的卫星仰角、星地几何距离等卫星信息。由于仰角、星地几何距离需要用到地面站位置信息，卫星模型计算这些参数显然加大了接口的复杂关系，因而我们可以将计算放在需要用到这些参数的模型。如前面所述，可以采用基于接口的集成或者数据流的集成将卫星的这些信息发送给其它模型和算法。然而在原始数据生成中，卫星这些信息对于自由空间延迟、电离层延迟等模型的计算是必不可少的，不存在选择性，这里采用基于接口的集成易于实现，即卫星模型提供获取钟差、位置和速度等信息的接口并且实现这些接口，而其它模型通过对接口的引用访问这些接口。

与卫星模型相似，地面站模型也应当为其它模型提供获取位置、遮蔽角等信息的接口。另外，从地面站的组成看，接收机和地面站原子钟是地面站的组成部分，可以将它们基于组件的集成，地面站通过模型容器仿真这些子模型。

环境段的自由空间延迟模型、对流层延迟模型和电离层延迟模型的计算需要用到卫星和地面站的信息，这些模型应当通过引用来访问卫星和地面站的接口。通过接口获得计算所需信息后，执行计算，并将计算结果通过地面站模型的接口发送给地面站模型，地面站模型的接口应当设计相应的方法来实现。可以看出对流层延迟、电离层延迟等模型的这些计算过程是相同的，这些重复的接口操作增加了复杂性，我们可以通过建立环境模型来管理这些接口的操作，即这些模型作

为环境模型的子模型。其实现过程是，环境模型通过接口获取卫星和地面站的信息，并将这些信息通过容器发布给自由空间延迟等子模型执行其计算，计算的结果统一由环境模型通过接口发布给地面站模型。环境模型与这些模型之间的数据流关系如图 3.7 所示。多径模型的集成方式与环境模型类似，都要使用地面站模型和卫星模型的接口，考虑到多径模型属于地面段模型，我们不将其作为环境模型的子模型。

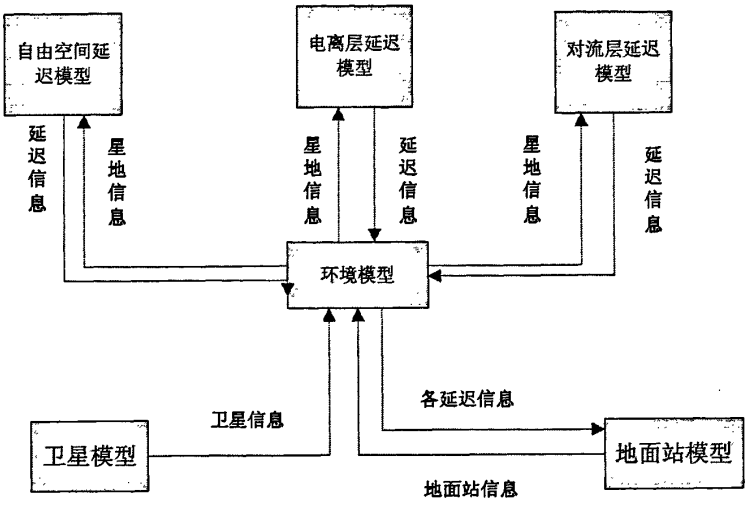

图 3.7 环境模型与其它模型的数据流关系

电离层关键事件模型根据时间判定是否触发关键事件。我们可以通过电离层延迟模型传递时间给电离层关键事件模型，如果时间在关键事件触发的时间范围内，则计算关键事件。通过电离层延迟模型与关键事件模型之间基于事件的集成达到上述目的。另外，该关键事件可以作为环境模型的子模型，环境模型通过容器访问关键事件获取其对信号传播的延迟。与此类似，对流层关键事件也可以这样实现。

第四章 基于 SMP2 的原始数据生成模型框架

SMP2 采用基于组件的体系结构，其中每个仿真模型都是一个模型组件，而且采用了专用机制支持同其它模型的通信。SMP 的仿真模型一般都有初始化入口点，用于对仿真模型进行初始化，本文中初始化入口点函数为 initEP()。理论上讲，SMP 的仿真模型开发可以采用多种平台，如 Java 和 C++等。依赖于使用的平台，也可能存在语言特定的本地机制，在本文，我们采用 C++平台。对于 C++平台，每个仿真模型用一个类(class)表示。每个类可以访问其它类的公共成员，也可以通过友元(friend)机制访问保护和私有成员，然而，这要求每个类知道其它类的实现，使得类之间存在着依赖性，这种集成方式仍然是基于类的集成。提供可以访问操作的接口是模型间通信最重要的概念，也是 SMP2 的核心概念。另外，属性也可被用于控制对数据的访问。对于基于事件的编程，SMP2 提供了可选的事件源(Event Sources)和事件槽(Event Sinks)机制。而涉及基于数据流集成的仿真模型必须提供输出字段的读访问，提供输入字段的写访问。最后，基于组件集成的仿真模型综合了上述集成方式的机制，可以根据仿真模型的特点综合采用上述集成机制。

本章将在第三章模型体系分析的基础上，分别建立了基于接口、事件、数据流和组件的模型框架，在此基础上建立基于 SMP2 的原始数据生成模型框架。

4.1 基于接口集成的模型框架

基于接口的集成是一种基于公共接口与实现相分离的程序设计方法。使用清晰的接口可以使软件，特别是大型应用软件易于维护和扩展，C++和 Smalltalk 语言是倡导基于接口设计的先驱。Java 的创造者意识到了基于接口程序设计的优点，随后直接使接口机制成为 Java 的一部分，同样，接口也是微软.NET 平台特别是 C#程序设计语言的基础概念。

一个接口声明了一组公共特征，一般为方法和属性，这些特征由模型提供，而且通过提供的接口公布给外界。因而接口不包含任何关于这些特征实际实现的信息。但是，如果一个模型提供了一个接口，则它必须确保支持接口声明的所有特征。

其它模型可以使用接口的特征，此时需要一个提供了接口实现的模型。使用者可以通过提供的接口使用需要的方法或属性，而无需了解或依赖于提供接口的模型的具体实现。图 4.1 显示了基于接口集成的机制。

如图所示，接口 Interface 根据模型间交互的需要声明了一系列的方法和属性，实现模型 Provider 提供了接口 Interface 所声明的方法和属性的具体实现，使用接口的模型 Consumer 通过对接口的引用访问 Provider 所实现的方法和属性，而不需

要了解 Provider 的具体实现和其它设计细节。在仿真模型集成过程中采用这种方式，接口使用者 Consumer 可以：

- 获取接口提供模型的信息；
- 设置接口提供模型的信息。

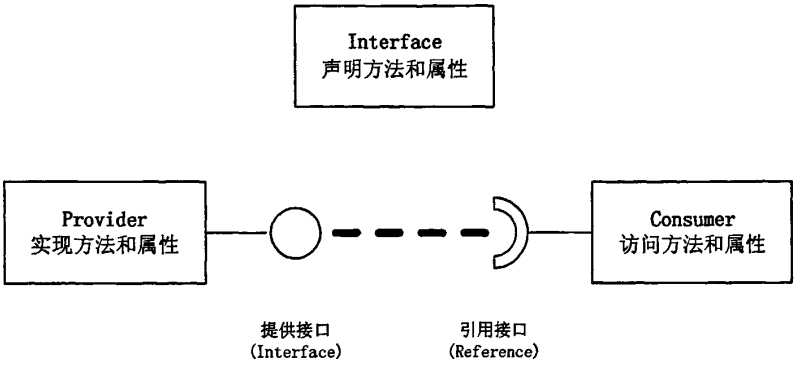

图 4.1 基于接口的集成机制示意图

这也正是基于接口集成的模型交换方式，即仿真模型的信息交换方式，设计仿真模型时应当充分利用这种集成方式的优点。

在第三章提到，在原始数据生成的仿真模型体系中，采用基于接口集成的模型是卫星模型、地面站模型与其它使用这些模型信息的模型。卫星模型和地面站模型为自由空间延迟模型、电离层延迟模型和对流层延迟模型以及多径模型提供位置、速度、钟差、遮蔽角等信息，为减少模型接口的复杂性，我们把前三个模型与环境模型基于组件的集成，而对于地面着模型、卫星模型的访问完全通过环境模型进行，多径模型则直接访问卫星模型和地面站模型的接口。下面我们分些卫星模型、地面站模型与环境段模型、多径模型之间的接口关系。图 4.2 显示了在原始数据生成中基于接口集成的模型框架，图中显示了卫星接口 ISatellite、地面站接口 IStation、地面站模型 StationModel、卫星模型 SatelliteModel 和环境模型 EnvironmentModel、多径模型 MultipathModel。为说明基于接口集成的机制并使图形简明，图中只注明了 IStation、ISatellite 接口的部分方法和 EnvironmentModel、MultiPathModel 的部分字段，而省略了其它的设计细节。

在图 4.2 中我们设计了 ISatellite 和 IStation 两个接口，接口中设计了多个方法，分别用于获取卫星模型和地面站模型的有关信息，其中 ISatellite 接口还设计了接口使用者对地面站模型进行设置的方法。各方法的说明参见表 4.1。

环境模型 EnvironmentModel 在计算过程中需要用到地面站位置 m_pStationPos、卫星位置 m_pSatPos、卫星速度 m_pSatVel 等数据，电离层延迟模型等环境模型的子模型使用这些数据进行计算，环境模型再把这些计算结果通过地面站接口发布给地面站模型。在图中只列出了把计算结果之一的电离层延迟 m_flonosphericDelay。由于在数据的获取和发布之间还要执行环境模型子模型的计

算，也就是说 EnvironmentModel 需要调度两次以执行不同的功能，为适应这两种不同的数据交换，我们给 EnvironmentModel 设计了两个不同的入口点，其中

- getEP()，用于获取数据卫星模型和地面站模型信息并发布给其子模型；
- setEP()，用于获取其子模型计算结果并发布数据给地面站模型。

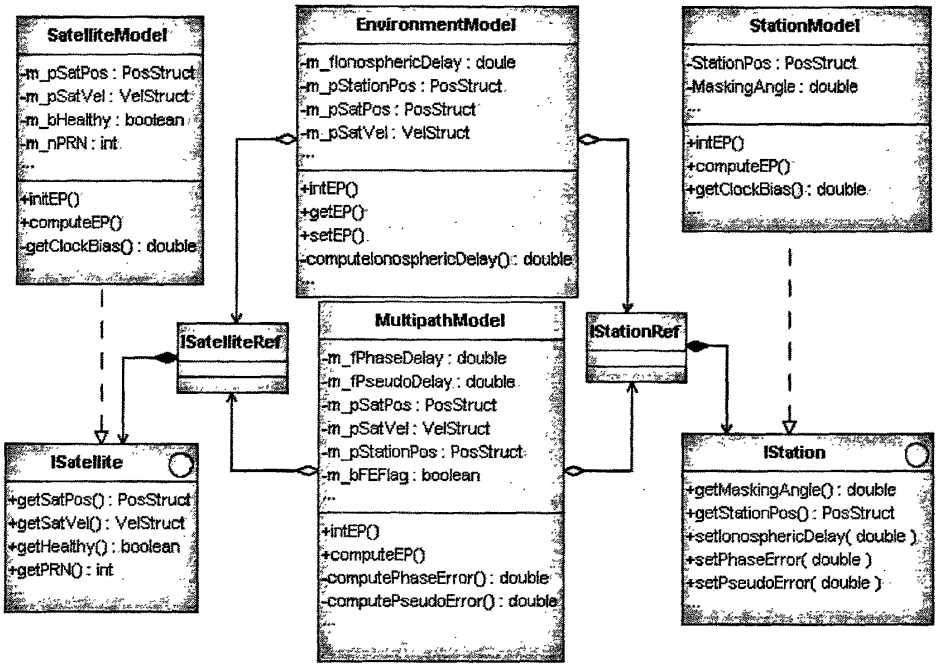

图 4.2 基于接口集成的模型框架

而 MultipathModel 不存在子模型，在运行过程中不需要调度两次以执行不同功能，所以设置一个入口点函数即可。在图中，EnvironmentModel 通过 ISatellite

表 4.1 接口说明

ISatellite		IStation	
getSatPos()	用于获取卫星位置	getMaskingAngle()	用于获取遮蔽角
getSatVel()	用于获取卫星速度	getStationPos()	用于获取地面站位置
getHealthy()	用于获取卫星健康状态	setIonosphericDelay()	用于设置电离层延迟
getPRN()	用于获取卫星 PRN 号	setPhaseError()	用于设置多径相位误差
		setPseudoError()	用于设置多径伪距误差

接口的 getSatPos()和 getSatVel()方法访问 SatelliteModel，并获取该模型的卫星位置和速度信息；通过 IStation 接口中 getStationPos()方法访问 StationModel 中的地面站位置信息；环境模型获取这些信息后，将计算结果之一的电离层延迟值通过 IStation 接口中的 setIonosphericDelay()方法发送到 StationModel。MultipathModel 使用接口的方式和过程与此类似。

这样的设计，充分利用了基于接口的集成的模型信息交互。对于基于接口的集成，使用接口的模型需要在代码中获取接口的方法，下面 EnvironmentModel.cpp 中的代码片断显示了这个实现过程：

```
::RDG::RDGModel::SpaceSegment::ISatellite *pISat;//卫星接口指针
m_pSatPos=pISat->getSatPos();//获取卫星当前位置
m_SatVel=pISat->getVel();//获取卫星当前速度
...
::RDG::RDGModel::GroundSegment::IStation *pIStation;//地面站指针
m_pStationPos= pIStation->getStationPos();//获取地面站位置
...
pIStation->setIonoDelay(m_fIonosphericDelay);//设置电离层延迟
...
```

另外，基于接口的集成必须在仿真模型装配阶段提供接口与接口实现模型之间的链接。我们将在 4.4 中说明接口与模型之间的链接。

4.2 基于事件集成的模型框架

基于事件的集成允许模型间自动的数据传播。拥有事件源（Event Sources）的模型(称为发布者 Publisher)可以被连接到一个或多个拥有同样事件类型的事件槽（Event Sinks）的模型(称为使用者 Consumer)。无论何时， 事件源模型中的事件被触发，将激活对应的所有事件槽的事件句柄。为了区分不同类型的事件，每个事件具有一个事件类型。该类型定义了在事件激活时传递给事件句柄的数据（事件定义了数据签名）。基于事件集成的机制如图 4.3 所示。

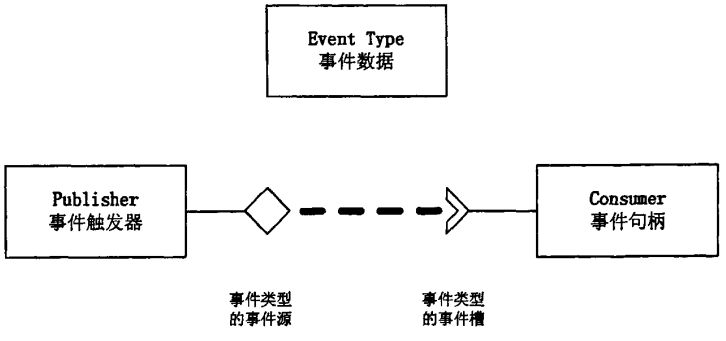

图 4.3 基于事件的集成机制示意图

如图所示，基于事件的集成必须声明事件类型(Event Type)，事件类型可以是 SMP2 支持的事件数据类型。目前 SMP2 只支持基本数据类型，如布尔类型 Bool、8 位的字符类型 Char8、32 位的整型 UInt32 等。事件发布模型和事件响应模型必须使用一致的事件类型，并且在装配过程中，将事件源和事件槽链接。

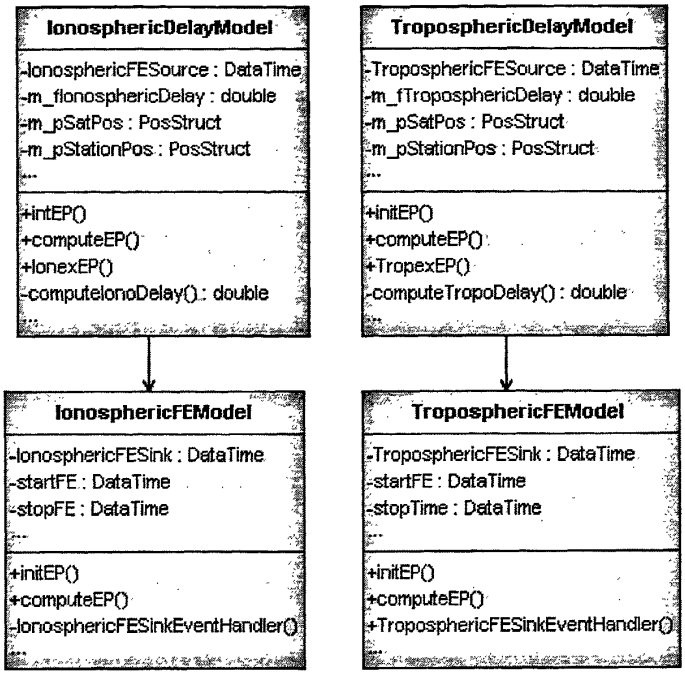

图 4.4 基于事件集成的模型框架

在第三章中提到的基于事件集成的模型有：电离层延迟模型与电离层关键事件、对流层延迟模型与对流层关键事件。电离层关键事件的触发条件可以有多种形式。在这里，我们在电离层延迟模型中定义了时间类型的事件源，电离层延迟模型将当前的仿真时间以事件的形式发送给电离层关键事件模型，电离层关键事件模型收到事件消息后，判断是否计算关键事件导致的电离层延迟时间和相位变化。对流层延迟模型与对流层关键事件与此类似。图 4.4 给出了基于事件集成的模型框架。

事件类型在模型之外定义，在这里我们在这些模型的同一个名字空间中定义了时间类型 DataTime 的事件类型。在图中可以看到，电离层延迟模型 IonosphericDelayModel 定义了 DataTime 类型的事件源 IonosphericFESource，电离层关键事件模型 IonosphericFEModel 中定义了 DataTime 的事件槽 IonosphericFESink 和处理事件消息的函数 IonoFESinkEventHandler()。我们设计 IonosphericDelayModel 在每一次仿真计算中都会通过事件源 IonosphericFESource 的 Emit 函数(该函数由 SMP 提供)发送事件消息给 IonosphericFEModel，后者根据由事件句柄 IonosphericFESinkEventHandler()处理事件消息，如果事件的 DataTime 处于 startFE 和 stopFE 之间，则句柄函数按照关键事件触发来出具，否则按不触发关键事件。下面的代码片断显示了这一事件传递和处理机制。

在 IonosphericDelayModel.h 中事件源的定义：

```
// 关键事件源
protected:
  // internal event-source of void type
  Smp::Mdk::EventSource< ::Smp::DateTime>* _IonosphericFESource;
public:
  // exported event-source field
  Smp::IEventSource* IonosphericFESource;
```

在 IonosphericFEModel.h 中事件槽和事件处理函数的定义：

```
//关键事件槽
protected:
// internal handler function, called from the EventSink base class
void IonoFESinkEventHandler( Smp::IObject* sender, ::Smp::DateTime arg );
// internal event-sink of void type
Smp::Mdk::EventSink< ::Smp::DateTime>* _IonoFESink;
public:
  // exported event-sink field
  Smp::IEventSink* IonoFESink;
```

在 IonosphericFEModel.cpp 中，通过 IonosphericFESinkEventHandler 函数处理事件消息。基于事件的集成，事件源与事件槽之间需要在装配过程中进行链接，二者的链接代码及其说明请参见附录 B.1。

4.3 基于数据流集成的模型框架

在基于数据流的体系结构下，模型实例之间的数据传输基于输入字段和输出字段之间的链接。SMP2 中的这种模型间通信机制，与其它机制不同的是模型实例并不主动同其它模型交互：数据传输由调度文档设计，仿真器完成，数据传输需要读取源模型实例的输出字段值，并将其写入目标模型实例的输入字段。

基于数据流体系结构的软件由通过数据链或数据通道进行通讯的并行运行的独立组件集合构成。这种设计可以用图 4.5 进行简洁地描述。提供数据的模型我们称为源模型(Source)，使用数据的模型称之为目标模型(Target)。源模型定义了用于提供数据的字段，设置其属性为输出，目标模型则定义数据类型相同的字段，并且设置其属性为输入。源模型和目标模型需要在仿真模型装配过程中对这两个字段进行链接，并且在运行调度阶段在调度文档中指定传输数据，仿真器读取调度指令后执行数据的传输。需要指出的是，目标模型可以有多个，相互间并不影响。这种集成方式，使得仿真模型可以进行交互与通信，并使得仿真模型具有较低的耦合性和较大的重用性。

在基于数据流集成的两个模型之间，数据流可以是一组数据或者单个数据，这里的数据可以是 SMP 定义的数据类型，也可以是自定义的数据类型。

在第三章提到的基于数据流的集成有：星体模型与摄动力模型，以星体位置为数据流；摄动力模型与轨道计算模型 OrbitCombuteModel，以摄动力和卫星位置、速度为数据流。这里，星体模型包括太阳模型 SunModel、地球模型 EarthModel 和

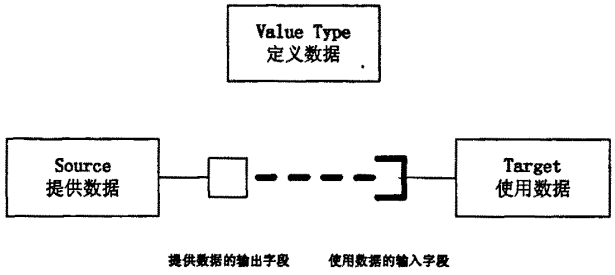

图 4.5 基于数据流的集成机制示意图

月球模型 MoonModel；摄动力模型包括前面所述的太阳摄动模型 SolarPerturbationModel、太阳光压模型 SolarPressureModel 月球摄动模型 MoonPerturbationModel、地球引力模型 EarthGravityModel、相对论效应 RelativisticModel，其它的摄动力模型设计方式与此类似。图 4.6 给出了原始数据生成模型中基于数据流集成的模型框架，为使框架清晰，图中各模型省略了其设计细节。

UML 中并不支持字段的输入输出属性，在图中我们并没有标注这些属性，这个步骤在仿真模型设计阶段由其开发工具设置。太阳模型 SunModel 的字段 m_pSunPos 保存太阳位置，该字段设置输出属性，用于传输将太阳位置传输给相关摄动力模型。太阳摄动模型 SolarPerturbationModel 的字段 m_pSunPos 保存太阳位置，该字段设置输入属性，用于接收 SunModel 的 m_pSunPos 的数据；字段 m_Perturbation 保存计算出的摄动力，设置输出属性，用于向轨道计算模型传输摄动力；字段 m_pSatPos 和 m_pSatVel 分别用于接收从轨道计算模型 OrbitComputeModel 的卫星位置和卫星速度，设置了输入属性。在 OrbitComputeModel 中字段 Perturbation 设置了输入属性，用于接收来自各个摄动力模型传输过来的摄动力；字段 m_pSatPosToPer 和 m_pSatVelToPer 设置了输出属性，分别用于传输计算所得的卫星位置和速度给各摄动力模型。

第二章提到轨道计算模型生成轨道用到三种方法：Runge-Kutta 积分、Adams-Moulton 积分、和 SP3 精密星历生成。为适应不同轨道生成方法，在这里的 OrbitCombuteModel 可以分别设置三个入口点对应三种轨道生成方法，便于仿真调度。

基于数据流的集成，各模型具有输入输出属性的字段(如上述 m_pSunPos 字段)并没有在 c++代码中体现。其数据流的实现的过程是：

- 在装配文档中建立输入与输出字段间的链接；
- 在运行阶段按调度文档指令进行传输。

基于数据流集成的模型较多，多个字段需要在装配文档中进行链接和在调度文档中进行传输。为了说明基于数据流集成的模型之间装配和传输，本文在附录

B.2 中给出了 SunModel 和 SolarPerturbationModel 中的 m_pSunPos 字段链接与传输。其它字段的链接与传输与此类似。

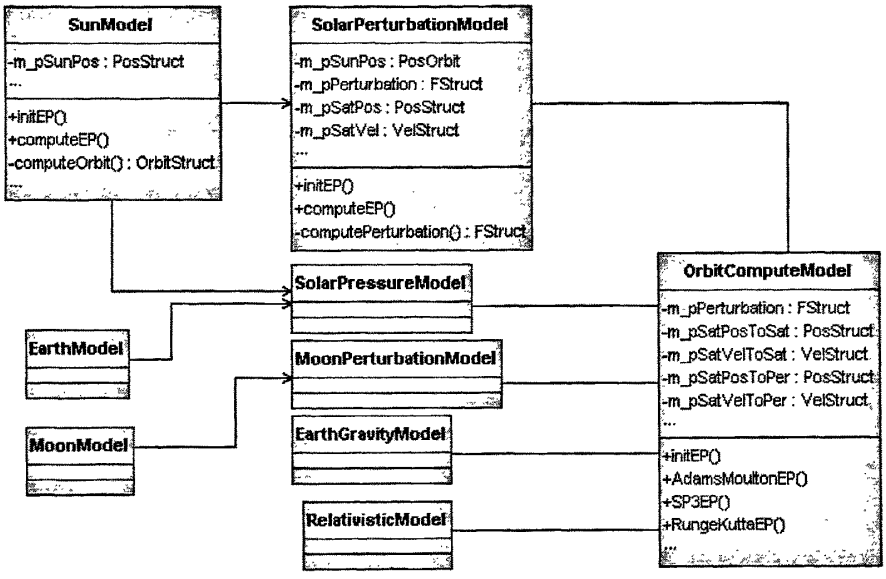

图 4.6 基于数据流集成的模型框架

4.4 基于组件集成的模型框架

在这里，我们把基于组件集成的模型称为组件模型。在第二章中提到，基于组件集成的组件模型通过两种方式管理其模型组件：聚集和组合。每一个被聚集的模型都提供一个接口，并由一个接口容器 Reference 管理，组件模型通过接口容器 Reference 获取对被聚集的模型的引用；每一个被组合的子模型都对应一个模型容器 Container，组件模型通过 Container 获取子模型实例并进行访问。

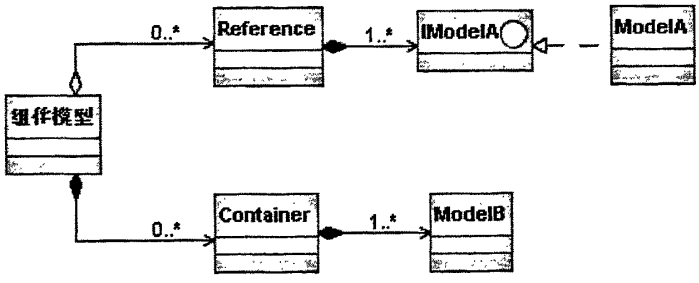

图 4.7 基于组件的集成机制示意图

基于组件的集成机制如图 4.7 所示。组件模型拥有若干个接口容器 Reference 和模型容器 Container。每一个接口容器 Reference 可以拥有若干个具有实现模型的接口 IModelA；而对于模型容器 Container，则可拥有若干个模型 ModelB。

在第三章提到，基于组件集成的模型有：太阳系和星体模型(太阳、地球和月球)；星座模型与卫星模型；卫星模型与轨道计算模型、星载原子钟模型；环境模

型与自由空间延迟模型、电离层延迟模型、对流层延迟模型、电离层关键事件模型、对流层关键事件模型以及卫星模型和地面站模型；地面站模型与接收机模型、地面站原子钟模型；多径模型与卫星模型、地面站模型；地面站网络模型与地面站模型。为方便说明，我们在这里给出了环境模型的组件集成框架，并且省略了框架内各模型的设计细节。其它基于组件集成的模型与此类似，不一一描述。

在图 4.8 中，环境模型 EnvironmentModel 通过 5 个模型容器 IonosphericContainer 、TroposphericContainer 、FreeSpaceContainer 、IonosphericFEContainer 以及 TroposphericFEContainer 分别管理其子模型组件电离层延迟模型 IonosphericDelayModel、对流层延迟模型 TroposphericDelayModel、自由空间延迟模型 FreeSpaceDelayModel、电离层关键事件模型 IonosphericFEModel 和对流层关键事件模型 TroposphericFE。

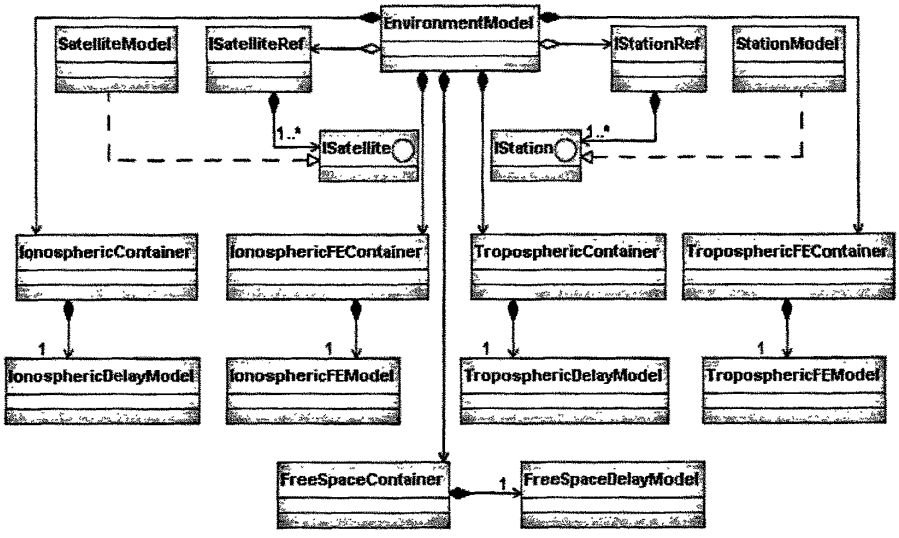

图 4.8 基于组件集成的环境模型框架

EnvironmentModel 可以通过相应的模型容器访问这些子模型组件：通过 2 个接口容器 ISatelliteRef 和 IStationRef 管理接口 ISatellite 和 IStation，环境模型 EnvironmentModel 通过这两个接口容器器访问相应的接口，从而获取实现这些接口的模型的有关功能。关于通过接口容器访问接口的过程我们在 4.1 中已经讨论，这里我们以电离层延迟模型为例，介绍环境模型 EnvironmentModel 如何对电离层模型 IonosphericDelayModel 进行交互。下面的 EnvironmentModel.cpp 代码片断显示了这里过程。

在这段代码中，首先通过电离层模型容器迭代器 IonoItr 获得电离层模型实例的指针*pIon，通过该指针可以对电离层模型计算所需的地面站位置、卫星位置等参数进行设置，计算完成，EnvironmentModel 通过 getIonosphericDelay()获得电离层延迟。

基于组件的集成包含有基于接口的集成，接口与实现模型之间在仿真模型装配阶段需要进行链接。本文在附录 B.3 给出了 EnvironmentModel 引用的 ISatellite 接口与其实现模型 SataelliteModel 之间的链接关系。

```
_IonosphericContainerIterator  IonoItr;//电离层模型容器的迭代器
::RDG::RDGModel::EnvironmentSegment::IonosphericDelayModel  *pIon;//电离层延迟模型]
IonoItr=_IonosphericContainer->Begin();
pIon=*IonoIt;
pIon->setStationPos(stationPos);//设置地面站位置
pIon->setSatPos(satPos);//设置卫星位置
pIon->setSatVel(satVel);//设置卫星速度
    ...
m_pIonoDelay=pIon->getIonosphericDelay();//获取电离层延迟
```

4.5 基于 SMP2 的模型框架及拓展

作为总体，原始数据生成的模型框架是在综合考虑各种集成方式的基础上建立的。综合本章前 4 节，我们在本节中建立了基于 SMP2 的原始数据生成模型框架。从本章的前面四节，我们得出原始数据生成中各模型的集成方式：

- 基于接口的集成

卫星模型、地面站模型与环境模型和多径模型；

- 基于数据流的集成

星体模型与各摄动力模型；各摄动力模型与轨道计算模型；轨道计算模型与卫星模型；

- 基于事件的集成

电离层延迟模型与电离层关键事件；对流层延迟模型与对流层关键事件；

- 基于组件的集成

太阳系模型与太阳模型、地球模型、月球模型；星座模型与卫星模型；卫星模型与轨道计算模型、星载原子钟模型；环境模型与自由空间延迟模型、电离层延迟模型、对流层延迟模型、电离层关键事件模型、对流层关键事件模型、卫星模型、地面站模型；多径模型与卫星模型、地面站模型；地面站模型与地面站原子钟模型、接收机模型。

基于以上分析，我们在图 4.9 中给出了基于 SMP2 的原始数据生成模型框架。为了能够展示一个整体的模型框架，图中省略了各模型应有的字段和函数，只保留了模型名称。图中各类名称含义参照前文。

在第三章关于原始数据生成的模型体系中提到，要进行更为精确的卫星轨道计算，需要考虑更多的摄动力模型，如地球固体潮、海洋潮、大气阻力摄动以及太阳系的其它行星的第三体摄动等。考虑链路预算分析、误差预算分析、OSPF 和 IPF 的仿真分析，也需要增加新的模型组件。我们的仿真模型集成方式能够很好的解决这些模型体系的拓展问题：

(1)增加行星模型。与太阳模型类似，我们可以增加新的行星模型与太阳系模型基于组件的集成；与太阳摄动模型类似，增加新的星体模型的摄动力模型，并与星体模型基于数据流集成，以行星位置作为数据流。

(2)增加摄动力模型。与其它摄动力模型类似，新的摄动力模型与轨道计算模型基于数据流集成，以摄动力、卫星位置和速度作为数据流，我们只需要在新的摄动力模型中增加相应的字段，并设置其输入输出属性。

(3)增加分析任务。数据的接口，可以考虑将链路预算、误差预算等用于分析的模型与地面站模型基于接口的集成，通过接口获取这些分析模块所需要的数据。

可见，我们所建立的模型框架具有良好的柔性，能够有效适应新模块的加入，而不需要重新设计模型框架；对其它模型造成影响很小，只需要对少数模型进行部分修改。

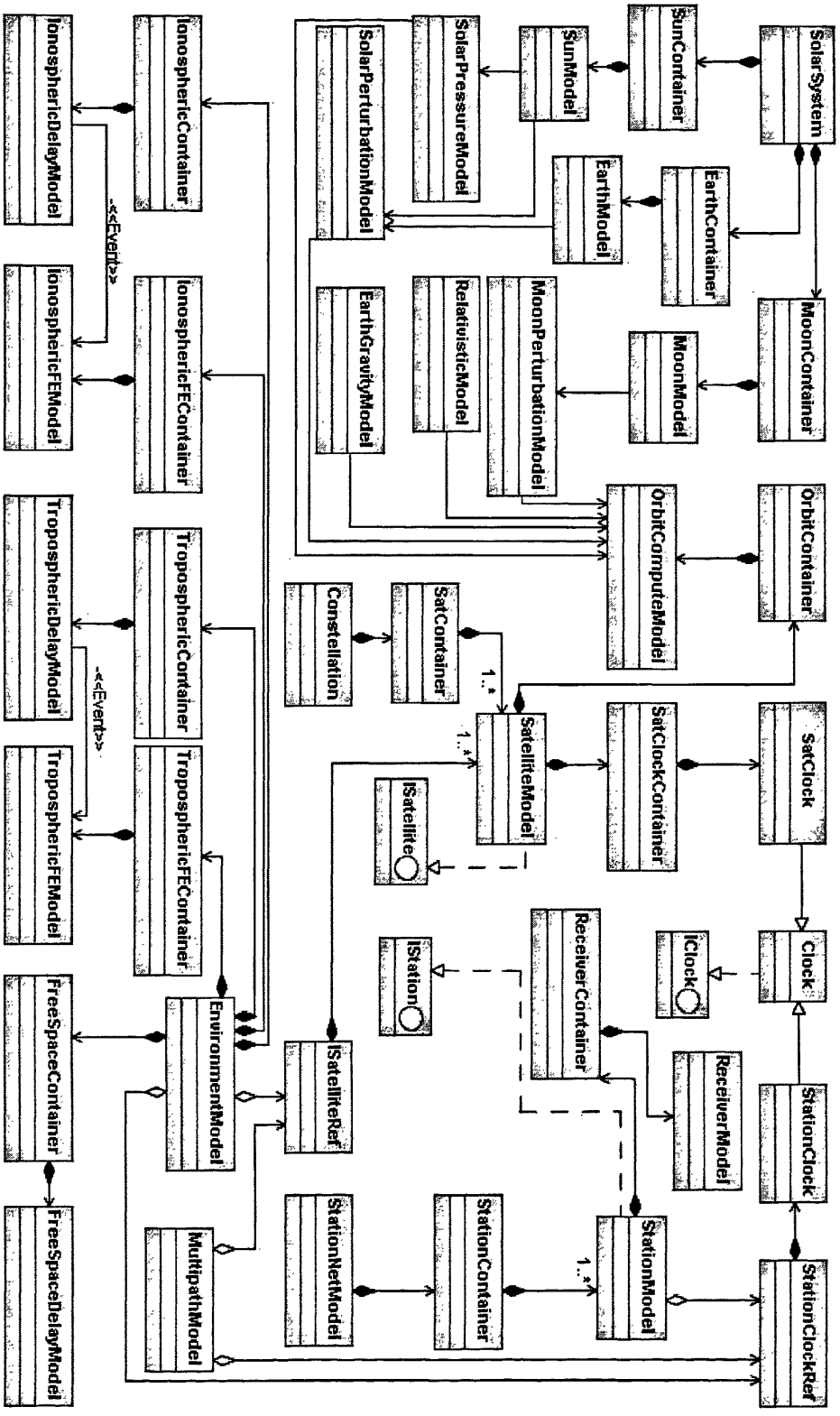

图 4.9 基于 SMP2 的原始数据生成模型框架

第五章 试验调度框架及仿真试验

前面提到，原始数据生成仿真涉及多学科领域模型。各领域模型的算法实现需要专业人员开发，因此要完全实现仿真试验，只有在各领域的专业人员的密切协作才能完成。本论文在第四章已经从系统工程的角度出发，建立了一个初步的基于 SMP2 的模型框架，然而，实现这一初步的模型框架仍存在许多困难。本章第一节在基于 SMP2 的模型框架基础上，根据 SMP2 的运行调度机制，建立了试验调度框架。在第二节中，考虑应用精密星历文件生成卫星轨道数据，建立了相应的调度框架，并进行了仿真试验分析。

5.1 试验调度框架

第二章提到，基于 SMP2 的仿真模型运行依赖于调度文档，调度文档定义了仿真任务和事件信息。在仿真模型设计阶段定义的仿真模型不一定都要调用，同一个仿真模型设置的多个入口点也要根据仿真任务的需要进行选择。因而在仿真试验分析阶段，涉及到仿真模型的调度问题：仿真模型的选择及其调度顺序；指定所调度的仿真模型的入口点函数；以及在基于数据流集成的仿真模型间进行数据传输。调度问题与仿真任务相关，在同一个模型框架内，不同的仿真任务往往选择不同的仿真模型，这些仿真模型一般是模型框架的一个子集，即仿真任务根

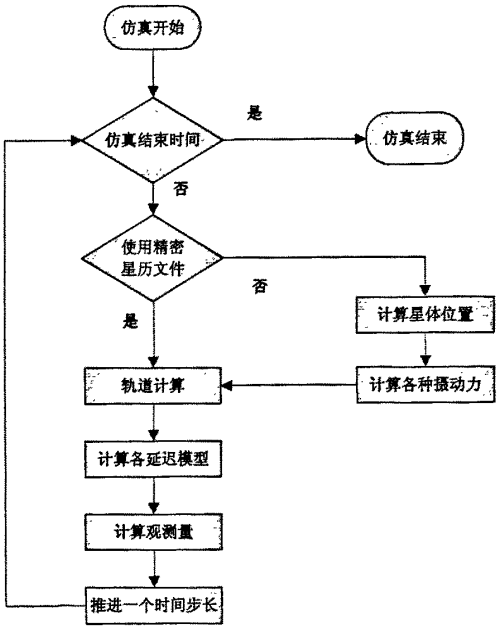

图 5.1 试验调度框架

据需要选择仿真模型，而不必调用全部仿真模型；而对于计算和功能复杂的仿真模型，往往需要设计多个入口点函数以便仿真模型执行不同的仿真计算功能；而对于基于数据流集成的仿真模型，其数据传输的时间也需要在调度文档中指定，

传输时间可以影响到计算的结果。一般而言，应该从仿真任务出发，根据任务选择仿真模型，根据模型功能指定入口点，根据计算过程确定数据传输。从计算过程看，原始数据生成仿真试验的调度比较固定，图 5.1 给出了原始数据生成仿真试验的一般性调度框架。

对于 4.5 节所建立的模型框架，考虑所包含的摄动力模型的卫星轨道积分，依据原始数据生成的数据流分析，以及我们所建立的仿真模型所要执行的功能，我们给出了以下调度顺序：

(1)触发太阳模型 SunModel，计算太阳位置；触发地球模型 EarthModel，计算地球位置；触发月球模型 MoonModel，计算月球位置；

(2)传输 SunModel 计算的太阳位置数据给太阳摄动模型 SolarPerturbationModel、太阳光压模型 SolarPresssureModel；传输 EarthModel 计算的地球位置给太阳光压模型 SolarPresssureModel；传输月球模型 MoonModel 计算的月球位置给月球摄动模型 MoonPerturbationModel；

(3)传输轨道计算模型 OrbitComputeModel 的卫星位置和速度数据给太阳摄动模型 SolarPerturbationModel、太阳光压模型 SolarPresssureModel、地球重力模型 EarthGravityModel、月球摄动模型 MoonPerturbationModel、相对论摄动模型 RelativistocModel；

(4)触发 SolarPerturbationModel，计算太阳摄动力；触发 SolarPresssureModel，计算太阳光压；触发 EarthGravityModel，计算地球摄动力；触发 MoonPerturbationModel，计算月球摄动力；触发 RelativisticModel，计算相对论摄动力；

(5)分别传输 SolarPerturbationModel、SolarPresssureModel、EarthGravityModel、MoonPerturbationModel、RelativistocModel 所计算出的摄动力数据给 OrbitComputeModel；

(6)触发 OrbitComputeModel 计算卫星位置和速度。根据仿真精度需要，我们分别为 Runge-Kutta 和 Adams-Moulton 轨道积分方法定义入口点，可以根据仿真任务的需要选择这两个入口点；

(7)触发星载原子钟模型 SatClockModel 及地面站原子钟模型 StationClockModel，计算钟差数据；

(8)触发卫星模型 SatelliteModel，获取 OrbitComputeModel 计算的轨道数据和 SatClockModel 计算所得的钟差数据；

(9)触发环境模型 EnvironmentModel，指定调度获取 SatelliteModel 及 StationModel 信息的入口点，并通过 EnvironmentModel 将这些信息发送给电离层延迟模型 IonosphericDelayModel，对流层延迟模型 TroposphericDelayModel、自由空间延迟模型 FreeSpaceDelayModel；

(10) 触发 IonosphericDelayModel、TroposphericDelayModel、FreeSpaceDelayModel，进行电离层延迟模型、对流层延迟模型和自由空间延迟模型的计算，而电离层关键事件模型和对流层关键事件模型计算与否则根据其自身时间设置决定是否触发关键事件的计算；触发多径模型 MultipathModel，获取 SatelliteModel 及 StationModel 的有关数据，计算多径延迟，并将计算结果发布给 StationModel；

(11)触发 EnvironmentModel，并指定调度发布数据给 StationModel 的入口点，获取前面计算的电离层、对流层和自由空间延迟，通过接口发布给 StationModel；

(12)触发 StationModel，计算伪距、载波相位、多普勒频移和信噪比等观测数据仿真值，并存入数据文件。

图 5.2 是使用 SMP2 辅助工具软件 XSim 设计调度文档的开发界面，显示了上述调度过程。对应的开发文档和调度文档设计参见附录 A。

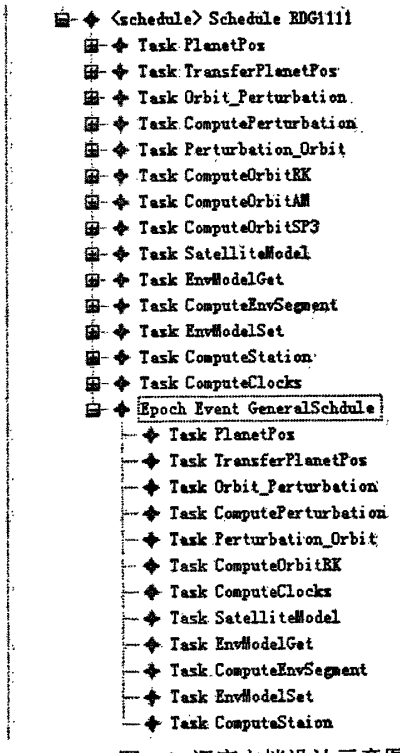

图 5.2 调度文档设计示意图

如图 5.2 所示，我们设计了 12 个 Task，分别对应上述的 12 个调度步骤，并且以一个 Epoch Event(纪元时间事件)组织这些 Task。这些 Task 中，PlanetPos 用于计算行星位置，包含触发 SunModel、SunModel 和 MoonModel 的三个 Trigger；TransferPlanetPos 包含 4 个 Transfer，用于传输步骤(1)计算所得的行星位置给相应的摄动力模型；Orbit_Perturbation 包含了 5 个 Transfer，用于将 OrbitComputeModel 的卫星位置和速度传输给相关的摄动力模型；ComputePerturbation 包含 5 个 Trigger，用于触发摄动力模型，计算各模型的摄动力；Perturbation_Orbit 包含 5 个

Transfer，用于传输摄动力模型计算所得的摄动力给轨道计算模型；ComputeOrbitRK 包含一个 Trigger，用于触发轨道计算模型，在这里我们使用 Runge-Kutta 积分，所以指定相应的入口点函数；ComputeClocks 包含两个 Trigger，用于触发星载原子钟模型和地面站原子钟模型，计算钟差；SatelliteModel 是一个包含触发卫星模型的 Task，使卫星模型获取其两个子模型 OrbitComputeModel 和 SatClockModel 的轨道和钟差数据；EnvModelGet 包含一个 Trigger，用于触发 EvironmentModel，指定获取卫星和地面站模型的数据的入口点函数；ComputeEnvSegment 包含 4 个 Trigger，分别用于触发 IonosphericDelayModel、TroposphericDelayModel、FreeSpaceDelayModel 和 MultipathModel，计算这些模型，其中触发 MultipathModel 包括发布其结果给 StationModel；EnvModelSet 包含一个 Trigger，用于触发 EvironmentModel，发布环境段各模型的计算所得的延迟值给地面站模型；ComoputeStation 用于计算观测量的仿真值。最后我们以一个纪元时间类型的事件(Epoch Event)组织这些 Task 的先后顺序。

仿真运行依赖调度文档，在调度文档中定义了这些 Task，并指明其传输的字段和触发的模型及其入口点函数，这些 Task 必须采用适当的事件排列其顺序，事件可以是 2.3 节中所列的 4 种时间类型事件，从上述原始数据生成仿真调度分析看，可以采用纪元时间类型事件。为了说明调度文档中 Task、Trigger、Transfer 和 Epoch Event 的表示方式，本文在附录 B.4 中给出了 Task PlanetPos、Task TransferPlanetPos 和 Epoch Event GeneralSchedule 的 XML 格式代码。

这里我们给出的是一个一般性的原始数据生成仿真的调度过程。在不影响计算逻辑混乱的情况下，这些调度顺序在局部是可以调整的，例如 ComputeClocks 可以放在 SatelliteModel 之前的任何一个位置。另外，我们可以通过触发或者不触发某些摄动力模型来考虑到摄动力的增加和减少，例如，如果不需要考虑相对论摄动，我们在 ComputePerturbation 中可以不设置触发 RelativisticModel 的 Trigger，在 Perturbation_Orbit 中取消 RelativisticModel 与 OrbitComputeModel 之间数据传输的 Transfer，反之亦然；如果在后续的仿真任务中需要增加对链路预算分析、误差预算等分析模型，则我们可以在 ComputeStation 后增加触发这些分析模型的 Task 即可。

5.2 基于精密星历生成轨道的仿真试验

GPS 观测技术近十年多年来发展迅速，GPS 数据处理中，精密星历是不可或缺的。IGS 精密星历自 1994 年开始发布以来，在国际上一直被广泛应用，精度不断提高。IGS 精密星历可以在其网站(http://igscb.jpl.nasa.gov)上得到，是数个 IGS 数据分析中心日常结算的卫星轨道加权平均的结果。IGS 的最终精密星历滞后 13 天产生，并且每个 GPS 周提供一次报告，给出每颗卫星的轨道精度。综合这些说

明文件，1994 年至今 IGS 精密星历的标称精度逐年提高，到 2001 年后，其标称精度提高至 1～2cm。在本节中，我们使用 SP3 精密星历生成卫星轨道，用于原始数据生成的仿真试验。

5.1.1 试验调度

在第四章提到，为便于仿真调度，我们在轨道计算模型 OrbitComputeModel 中定义用于 SP3 精密星历生成轨道的入口点，以便仿真调度。在基于精密星历生成卫星轨道的原始数据生成仿真试验中，我们的试验从调用这个入口点生成卫星轨道，而不用考虑用于 Runge-Kutta 积分和 Adams-Moulton 积分生成卫星轨道的星体模型、摄动力模型。这个仿真调度过程可以有不同的调度步骤，参考图 4.9 所提出的模型框架，我们给出了一个可行的调度步骤：

(1)触发 OrbitComputModel，指定入口点函数为计算 SP3 轨道入口点，计算卫星轨道的位置和速度；

(2) 触发星载原子钟模型 SatClockModel 和地面站原子钟模型 StationClockModel，计算钟差数据；

(3)触发卫星模型 SatelliteModel，使其获取 SatClockModel 的钟差数据和卫星轨道数据；

(4)触发环境模型 EnvironmentModel，指定调度获取 SatelliteModel 及地面站模型 StationModel 的入口点，通过基于组件集成机制将卫星和地面站信息发布给电离层延迟 IonosphericDelayModel 等子模型；

(5) 触发电离层延迟模型 IonosphericDelayModel、对流层延迟模型 TroposphericDelayModel 和自由空间延迟模型 FreeSpaceDelayModel，通过 EnvironmentModel 获取的 SatelliteModel 以及 StationModel 的有关数据进行计算；触发多径模型 MultipathModel，获取 SatelliteModel 及 StationModel 的有关数据，进行计算，并将计算结果发布给 StationModel；

(6)触发环境模型 EnvironmentModel，并指定调度发布数据给地面站模型的入口点，获取电离层延迟 IonosphericDelayModel 等各环境模型子模型的计算结果，通过接口发布给地面站模型 StationModel；

(7)触发地面站模型 StationModel，计算伪距、载波相位、信噪比和多普勒频移。

调度文档开发示意图 5.3 显示了上述步骤。在图中可以看到，我们定义了 7 个 Task。其中 OrbitCompute 只有一个 Trigger，用于触发 OrbitComputeModel，指定其基于精密文件生成卫星轨道的入口点；ClockBias 包含两个 Trigger，分别触发 SatClockModel 和 StationClockModel，用于计算星载原子钟和地面站原子钟的钟差；ComputeSatellite 包含一个触发 SatelliteModel 的 Trigger，使卫星模型获取星载原子钟钟差数据；EnvModelGet 有一个 Trigger，用于触发 EnvironmentModel，指定获

取卫星和地面站数据的入口点；ComputeEnvironment 包含 4 个 Trigger，分别用于触发电离层延迟模型、对流层延迟模型、自由空间延迟模型和多径模型，计算其延迟；EnvModelSet，包含一个触发 EnvironmentModel 的 Trigger，使 EnvironmentModel 获取 ComputeEnvironment 的计算结果，并通过接口将其发布给 StationModel；ComputeStation 用于触发地面站模型，计算观测数据。最后我们以一个纪元时间类型的事件(Epoch Event)组织这些 Task 的调度先后顺序。

为说明调度文档中 Epoch Event BasedSP3Simulation 的表示方式，本文在附录 B.4 中给出了其 XML 格式的调度文档代码。

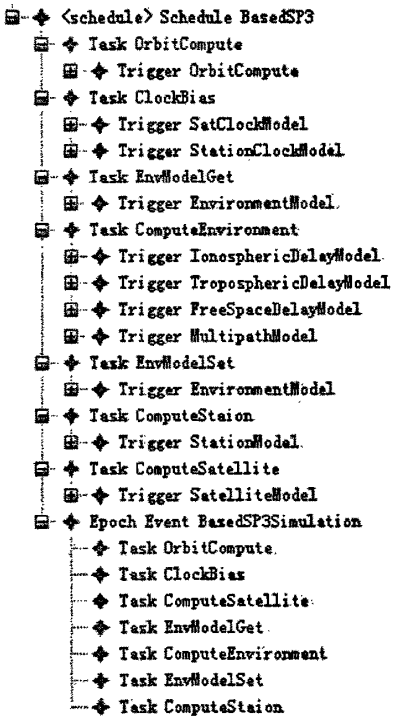

图 5.3 调度文档设计示意图

5.1.2 仿真示例

本次仿真试验分析选择 GPS 导航系统编号为 G08 的卫星的观测数据。试验需要设置包括卫星属性、对流层、多径效应、地面站、原子钟以及与信号干扰和跟踪等有关的多个参数设置，不触发关键事件。表 5.1 只列出了本次试验的主要设置。

表 5.1 试验主要配置

仿真基本设置	仿真开始时间	2002 年 10 月 12 日 8 时 0 分 0 秒
	仿真结束时间	2002 年 10 月 12 日 12 时 25 分 0 秒
	仿真时间步长	1 秒
卫星	卫星编号	G08
	遮蔽角	10 度
	视准轴	0.8168rad

属性设置	发射增益	0dB
	信号频率	1575.42MHz
	星载原子钟参数	初始钟差、时钟漂移、时钟漂移率和 Allan 方差采用 G08 号卫星原子钟参数
地面站属性设置	地面站位置 (ECEF，单位米)	(-2104083.2465,4815058.9145,3603414.3590)
	遮蔽角	10 度
	接收增益	根据发射增益表线性插值
	地面站温度	290K
	地面站原子钟参数	初始钟差、时钟漂移、时钟漂移率和 Allan 方差采用 ESA 提供的 BAN2 地面站原子钟参数
干扰相关设置	热噪声误差	是
	干扰系数因子	0
	综合损耗	7dB
多径效应设置	相位振幅误差	0.2 米
	相位偏差	0 米
	伪距振幅误差	2 米
	伪距偏差	0 米
	多径效应频率	0.8 周/弧度
	接收机灵敏度因子	1
信号跟踪设置	相干性	否
	接收阀值	5dB
	群延迟改正因子	0
环境段设置	自由空间	计算包括 Sagnac 效应和轨道偏心效应
	电离层	10.7 厘米太阳辐射通量为 120×10^{-22} W/m^{2}Hz
	对流层	对流层高度 11 公里，水蒸汽气压 0hPa

在下列图表中依次显示了这次仿真试验所得出的 L1 波段的伪距、载波相位、多普勒频移和信噪比的仿真时序图。

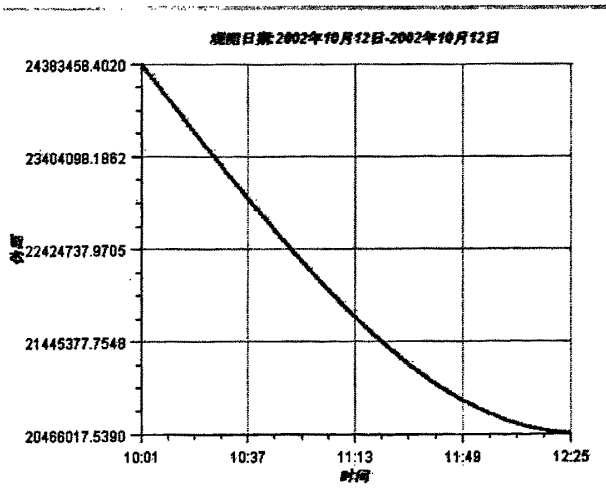

图 5.4 伪距仿真时序图

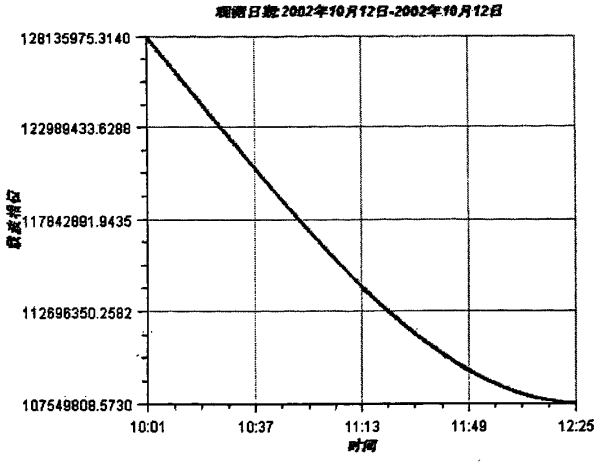

图 5.5 载波相位仿真时序图

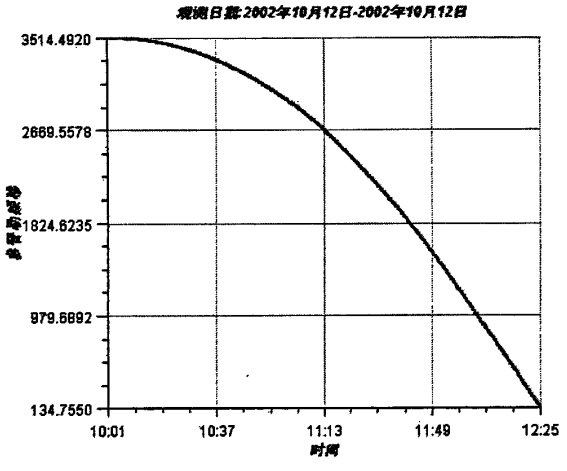

图 5.6 多普勒频移仿真时序图

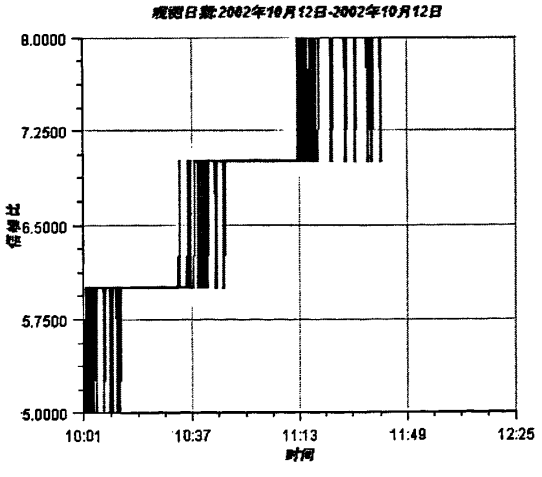

图 5.7 信噪比仿真时序图

由图 5.4~5.6 可以看出，随着伪距、载波相位和多普勒频移变化规律相同，即随着伪距的减小，载波相位和多普勒频移同时也减小；图 5.7 则可以看出，随着伪

距减小，卫星与地面站距离减小，从而卫星仰角随着变小，信噪比增大，即信号强度相对噪声变大，当低于接收阀值时，接收机拒绝接收信号。这说明仿真试验是符合实际的。

结 束 语

原始数据生成仿真试验是导航系统研制过程中总体仿真论证的重要组成部分，本文从原始数据生成的仿真背景出发，分析了影响导航信号观测量的因素，并对这些因素建立了一个初步的模型体系。在对模型体系进行了深入分析的基础上，根据 SMP2 的有关规范、规范，提出了一个基于 SMP2 的原始数据生成模型框架并已初步应用于卫星导航系统总体论证工作中。本文的主要贡献有：

(1)研究了 SMP 的建模仿真机制，重点研究了基于 SMP2 的仿真模型集成方法，根据 SMP2 集成仿真模型的特点提出了开发原始数据生成模型框架的开发过程，指出过程中各个阶段需要处理的问题和解决的方法，具有一定的代表性，本文的研究情况也验证了这个开发过程的可行性；

(2)参考 GSSF 以及有关文献，针对原始数据生成仿真的特点，综合考虑了影响导航信号观测数据的各种因素，提出了一个初步的原始数据生成模型体系，根据 SMP2 仿真模型集成方法的特点，对体系内的模型和算法进行实体模型分析、数据流分析和面向对象分析，在此基础上提出了基于 SMP2 的原始数据生成仿真模型体系，并明确了这些仿真模型之间的集成方式；

(3)依据 SMP2 的建模机制，分别建立了基于接口集成的模型框架、基于事件集成的模型框架、基于数据流集成的模型框架和基于组件集成的环境段模型框架，并讨论了采用各种集成方式的仿真模型装配和调度机制，在此基础上建立了基于 SMP2 的原始数据生成模型框架，并对该框架的扩展性进行了讨论；

(4)针对该模型框架，依据 SMP 的仿真运行机制，提出了仿真调度框架，并进行了基于精密星历生成轨道的原始数据生成仿真试验，部分的验证了模型框架的正确性。

攻读硕士学位的时间毕竟是短暂的，同时由于研究时间和所掌握的知识有限，在研究中难免会留下一些遗憾。本文虽然建立了原始数据生成仿真试验的模型体系，提出了基于 SMP2 的原始数据生成模型框架，并且对模型框架进行了部分验证，但是并没有实现框架内的所有模型，从而无法完全实现原始数据生成的仿真功能。实现所有模型和算法，并不是个别单位和个人能够胜任的事，正如绪论中所谈到的，需要有关领域的专业人员协同研究才能完成。

在下一步工作中，可以在以下方面努力：

(1)完全实现模型框架内的所有模型和算法，并进行仿真试验，生成观测数据的仿真值；

(2)建立仿真系统，支持仿真试验，研究原始数据生成的想定任务生成，使仿真系统能够充分支持用户想定试验配置；

(3)将这些观测量仿真值应用于论文中提到的几种用途，支持导航系统研制工作；

(4)结合导航系统的其它仿真论证项目，研究基于 SMP2 的导航仿真模型可重用性问题；

(5)原始数据生成所用的模型精度高、计算时间长，可以考虑研究如何提高 SMP 仿真模型的计算速度问题。

致 谢

在国防科技大学就读硕士的两年半时间中，得到了许多老师、同学、朋友的悉心指导、真诚鼓励和无私帮助。没有他们，我无法获得现在的成绩。在此对他们一并表示感谢！特别要对以下诸位表示由衷的感谢：

衷心感谢我的导师李群副教授！本文的工作从研究方向的确定、论文的选题到定稿都是在李老师的悉心指导下完成的。本人在研究生的课程学习、项目实践以及论文写作期间受到李老师的亲切关怀和悉心指导，他一直在思想和学习上给我教诲和帮助。李老师敏锐的洞察力、严谨求实的治学态度、博大精深的学术造诣时刻感染着我，使我受益匪浅。李老师待人宽厚仁慈，处世达观宽容，工作认真负责。他的辛勤耕耘、淡泊名利的人生哲学也将对我今后的工作、学习和生活产生深远的影响。

在研究生学习和研究期间，受到朱一凡教授的关心和帮助，在此表示衷心的感谢。还要特别感谢从我进实验室以来就坐我旁边给我巨大帮助和指导的侯洪涛老师，他为人谦逊，做事细心，教人耐心，是我完成实验室科研工作和撰写论文的良师益友。李竞杰老师高超的编程技术也深深的影响了我。还要向苏年乐、王晓双、王超、石福丽、刘喜春等博士在科研、学习和生活过程中给予的支持和帮助表示感谢！跟他们一起学习、研究和生活的快乐永远值得怀念，祝他们早日完成博士阶段的学业！

参考文献

[1] 帅平，田广吉，向开恒，现代卫星系统技术的研究与进展，中国空间科学技术，2004.6.

[2] GSSF Operations Manual Volume 1, GSSFP2.OM.001, Issue 2 Rev. 1, September 2004.

[3] GSSF Operations Manual, Volume 2, Algorithms and Models, GSSFP2.OM.002, Issue 3 Rev. 1, September 2004.

[4] GSSF Validation Testing Specification, GSSFP2.REP.035, Issue 5, September 2004.

[5] 阳振辉，李群，苏年乐，基于 SMP2 的导航系统原始数据生成模型框架及其仿真分析，系统仿真技术及其应用学术年会论文集，2007，9：286.

[6] J.R.Martin, I.Castrillo, A.B.Martin, S.Cilla, E.Mora,Galileo Orbitography and Synchronization Processing Facility(OSPF):Preliminary Design,ION GNSS 19th International Technical Meeting of the Satellite Division,2006,Fort Worth,TX.

[7] 韩玲，朱耀文，Galileo 系统及其在中国的应用，天文学进展，2005，23（1）:10

[8] E.Sardon,E.Mora,C.Hemandez,J.R.Martin,GMV S.A., Galileo Integrity Processing Facility:Preliminary Design, ION GNSS 19th International Technical Meeting of the Satellite Division,2006,Fort Worth,TX.

[9] Frank Zimmermann, Thomas Haak, Eduard Steindl. Generating Galileo Raw Data-Approach and Application. Data System in Aerospace 2005.

[10] STK 资料，www.stk.com.

[11] Rolf Dach,Urs Hugentobler,Pierre Fridez,Michael Meindl,User manual of the Bernese GPS Software Version 5.0. Astronomical Institute,University of Bern,Janiary 2007.

[12] 王华，唐国金，李海阳，航天系统分析与仿真基础程序库 AstroLib，系统仿真学报，2007，19(13)：2917.

[13] 王维平、朱一凡等，离散事件建模与仿真，湖南长沙：国防科技大学出版社，1997.

[14] Paul Gustavson and the Volunteers of the Base Object Model(BOM) Study Group(SG),"BOM Study Group Final Report"[EB/OL],http://www.sisostds.org,2001.

[15] SISO BOM PDG. "Base Object Model(BOM) Template Specification V0.12"[EB/OL], http://www.sisostds.org/, 2005.

[16] Steven W.Reichenthal. SRML-Simulation Reference Markup Language.W3C Note.2003,http://www.w3.org/TR/SRML/.

[17] European Space Agency. SMP 2.0 Handbook[R]. October, 2005.

[18] European Space Agency. SMP 2.0 Component Model[R]. October, 2005.

[19] European Space Agency. SMP 2.0 Metamodel[R]. October, 2005.

[20] European Space Agency. SMP 2.0 Cpp Mapping[R]. October, 2005.

[21] EuroSim SOFTWARE USER MANUAL, 2002.
[22] GSTB Operations Concept Document, 2003.
[23] GSTB SYSTEM ARCHITECTURE DOCUMENT, 2003
[24] ESA 项目资料，www.esa.int.
[25] Model Driven Architecture – Applying MDA to Enterprise Computing, D.S. Frankel, OMG Press, Wiley Publishing, 2004.
[26] 吴晓进，谢洪华等译，GPS 理论与应用，西安：西安导航技术研究所，1999.6
[27] 卡普兰 著 邱致和等译. GPS 原理与应用[M]. 北京 :电子工业出版社,2002.
[28] 刘林，胡松杰，王歆编著，航天动力学引论，南京：南京大学出版社，2006
[29] JPL-405 资料，http://ssd.jpl.nasa.gov/iau-comm4/relateds.html.
[30] 郭金运，由星载 GPS 数据进行 CHAMP 卫星定轨和地球重力场模型解算，山东科技大学博士论文，2004.
[31] 胡松杰，陈力，刘林，卫星星座的结构演化，天文学报，2003.2,44(1):46.
[32] 徐福详主编，卫星工程，北京：中国宇航出版社，2002.10.
[33] 李济生等编，航天器轨道确定，北京：国防工业出版社，2003.4.
[34] 夏南银等编，航天测控系统，背景：国防工业出版社，2002.10.
[35] 章红平，平劲松等，电离层延迟改正模型综述，天文学进展，2006.3，24(1).
[36] 冯铁军，唐家兵，谢世杰，GPS 定位中的电离层误差，全球定位系统，2002，27(1):13.
[37] 殷海涛，黄丁发等，GPS 信号对流层延迟改正新模型研究，武汉大学学报(信息科学版)，2007，32(5).
[38] 朱国辉，张大棚，戴钢，张宁，GPS 定位系统中的几种对流层延迟模型，2006，4：35.
[39] 余丹，廖凯宁，GPS 全球定位系统的改进与进展，全球定位系统，2006，1：23.
[40] 孙常建，杨晓超，GPS 多路径效应规律的研究，测绘通报，2006，11：12.
[41] 曾庆化，刘建业，彭文明，于永军，我国导航系统相关技术发真分析，航天控制，2006，24(4)：91.
[42] John P. Hancock and the Volunteers of the RFOM SG. "Reference FOM Study Group Final Report Version 1.0" [EB/OL], HTUhttp://www.sisostds.org/UTH, 1998.
[43] Graham Shanks. "Real-time Platform Reference Federation Object Model (RPR FOM) Version 2.0D17" [EB/OL], http://www.sisostds.org/, 2003.
[44] 彭琼芝等，SRML——种基于 XML 的仿真模型描述语言，计算机仿真，第 22 卷 12 期，2005 年 12 月.
[45]Larson.W.J.,Wertz.J.R.,"Space Mission Analysis and Design-Second Edition",Microcosom,Inc. and Kluwer Academic Publisher,1992,P.189-191.
[46] B.Hofmann-Wellenhof,H.Lichtenegger,J.Collins,GPS Therory and Practice, Fourth

Edition,section6.3.3 Tropospheric Refrection,Springer-Verlag.

[47] 美国 GPS 现代化概述，陈俊勇，测绘通报，2000.10.

[48] 王维平，面向对象的多媒体仿真方法研究，博士论文，1997.

[49] 李群，柔性仿真的理论与实践，博士学位论文，湖南长沙：国防科技大学，1999.

[50] 王维平，李群等，柔性仿真原理与应用，湖南长沙：国防科技大学出版社，2003.

[51] 谭跃进、陈英武、易进先，系统工程原理，湖南长沙：国防科技大学出版社，1999.

[52] 黄柯棣，系统仿真技术，湖南长沙：国防科技大学出版社，1999.

作者在读学期间取得的学术成果

一、发表的相关论文

[1] 阳振辉，苏年乐，李群．基于 SMP2 的导航系统原始数据生成模型框架及仿真分析．系统仿真技术及其应用学术年会论文集．2007 年 8 月．

附录 A 仿真模型设计与装配示意图

附录 A 给出了使用 SMP2 辅助工具软件 XSim 开发原始数据生成仿真模型的界面，包括仿真模型设计文档开发界面、装配文档开发界面。

A.1 设计文档开发示意图

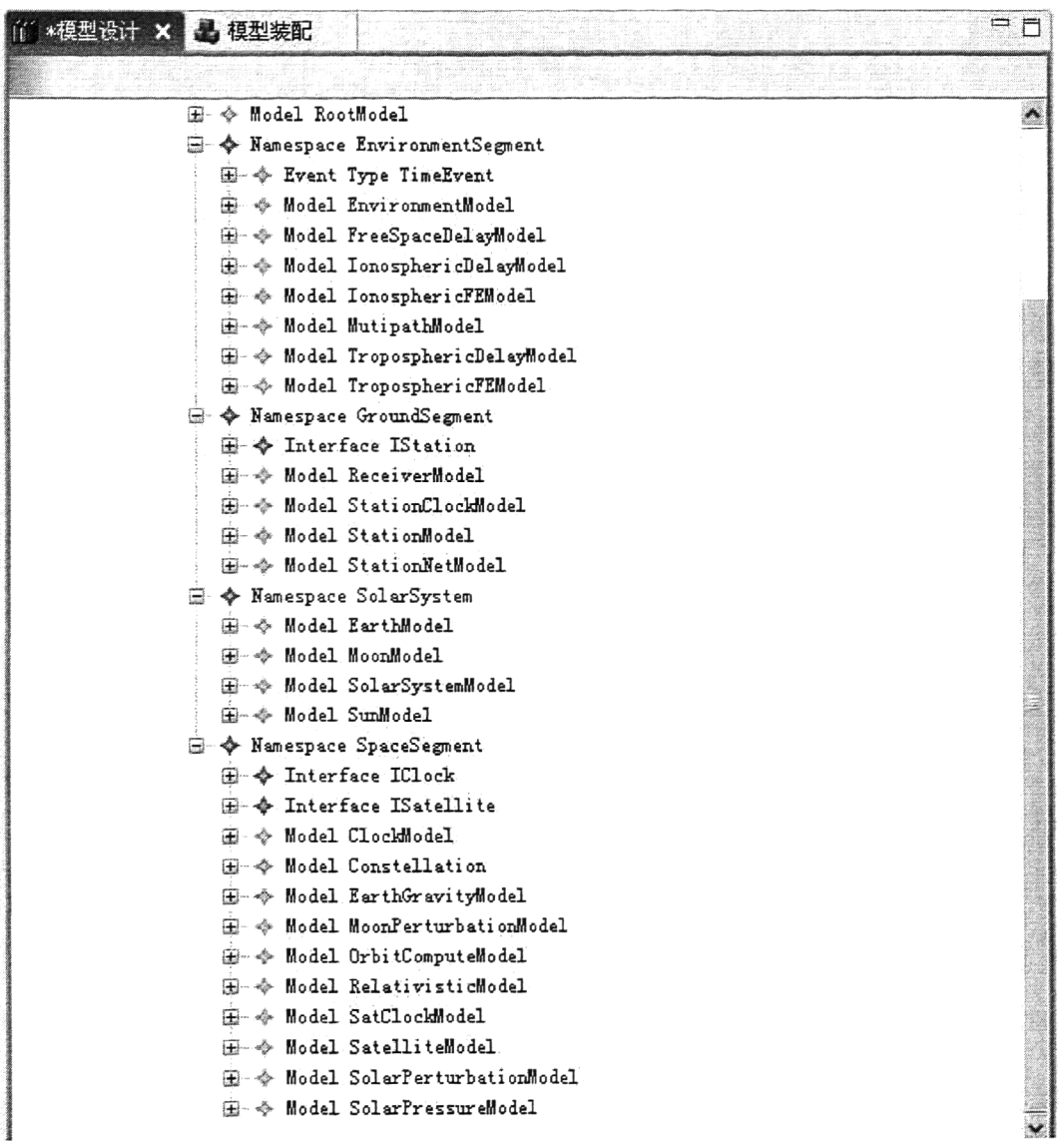

图 A1.1 设计文档开发示意图

图 A1.1 显示了基于 SMP2 的原始数据生成仿真模型设计界面，这些仿真模型分别置于 EvironmentSegment、GroundSegment、SolarSystem 和 SpaceSegment 这四个 namespace，对应于 C++代码中的 namespace。总共有 25 个 Model，每一个 Model 对应 C++代码中的模型类，包括头文件(.h)、实现文件(.cpp)以及 SMP2 的相关文件。在 EvironmentSegment 中定义了一个时间类型的事件类型 TimeEvent，用于定义电

离层关键事件和对流层关键事件的事件类型。此外还定义了两个用于基于接口集成的接口 IStation、ISatellite，其对应的实现模型是 StationModel 和 SatelliteModel。

A.2 装配文档开发界面

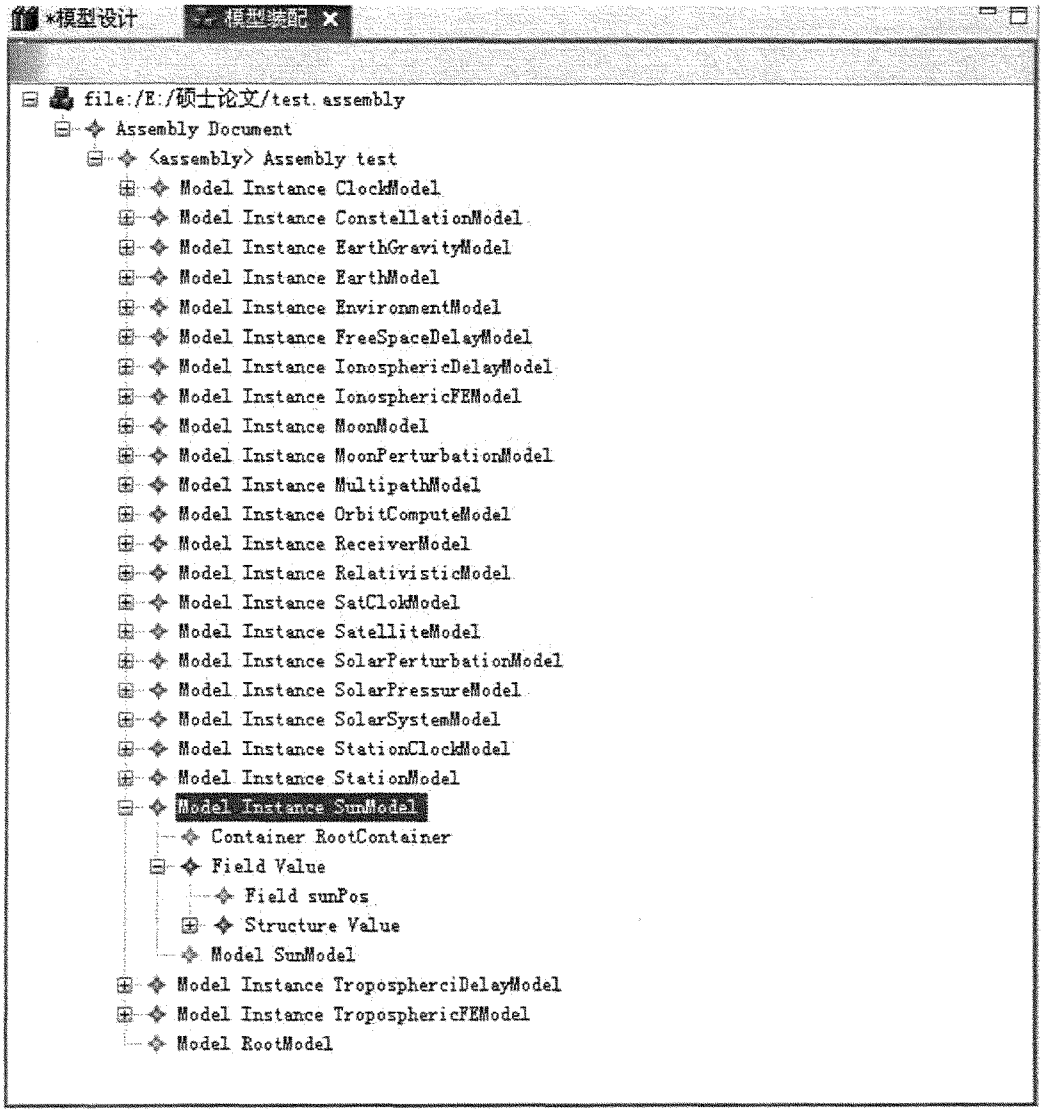

图 A1.2 装配文档开发示意图

图 A1.2 中显示了图 A1.1 中设计的基于 SMP2 的原始数据生成仿真模型的装配文档设计界面。图 A1.2 中显示了图 A1.1 中设计的基于 SMP2 的原始数据生成仿真模型的装配文档设计界面。装配文档中可以看到设计文档中所设计的仿真模型实例，如 Model Instance SunModel 是图 A1.1 中 SunModel 的模型实例。

附录 B 相关代码

附录 B 中给出了本文中提到的原始数据生成仿真模型的设计文档、装配文档和调度文档部分代码。

B.1 基于事件集成的事件链接

下面这段代码显示了 4.2 中提到的基于事件集成的电离层延迟模型 IonosphericDelayModel 与电离层关键事件模型 IonosphericFEModel 在装配文档中的事件链接代码：

```
<Link xsi:type="assembly:EventLink"
Id="IonosphericFEModel.IonosphericFEModel_IonosphericFESink__IonosphericDelayModel
_IonosphericFESource"
Name="IonosphericFEModel_IonosphericFESink__IonosphericDelayModel_IonosphericFESource">
<EventSink xlink:href="RDG1001.catalogue#RDG.RDGModel.EnvironmentSegment.IonosphericFEModel.
    IonosphericFESink" xlink:title="IonosphericFESink"/>
<Provider xlink:href="test.assembly#IonosphericDelayModel" xlink:title="IonosphericDelayModel"/>
<EventSource xlink:href="RDG1001.catalogue#RDG.RDGModel.EnvironmentSegment.IonosphericDelayModel.
    IonosphericFESource" xlink:title="IonosphericFESource"/>
</Link>
```

代码的说明如下：

- <Provider>指明了事件的提供者是 test.assembly 中的电离层延迟模型 IonosphericDelayModel；
- <EventSource> 指明了 IonosphericDelayModel 中的事件源 IonosphericFESource；
- <EventSink>指明了 IonosphericFEModel 中的事件槽 IonosphericFESink。

代码中的 xlink:href=”RDG1001.catalogue”指明了该装配文档是基于设计文档 RDG1001.catalogue 而设计的。

B.2 基于数据数据流集成的字段链接与传输

下面的装配文档片断显示了 4.3 中太阳模型 SunModel 的 m_pSunPos 字段与太阳摄动模型 SolarPerturbationModel 的 m_pSunPos 字段之间的链接关系：

```
<Link  xsi:type="assembly:FieldLink"
Id="SolarPerturbationModel.SolarPerturbationModel_m_pSunPos__SunModel_m_pSunPos"
Name="SolarPerturbationModel_m_pSunPos__SunModel_m_pSunPos">
 <Input xlink:href="RDG1001.catalogue#RDG.RDGModel.SpaceSegment.SolarPerturbation.m_pSunPos"
        xlink:title="m_pSunPos"/>
 <Source xlink:href="test.assembly#SunModel" xlink:title="SunModel"/>
 <Output xlink:href="RDG1001.catalogue#RDG.RDGModel.SolarSystem.SunModel.m_pSunPos"
         xlink:title="sunPos"/>
</Link>
```

代码的说明如下：

- <Source>指明了数据的提供者是 test.assembly 中的太阳模型 SunModel；
- <Output>指明了输出字段是 SunModel 中的 m_pSunPos 字段；
- <Input>指明了输入字段是 SolarPerturbationModel 中的 pSunPos 字段；

代码中的 xlink:href="RDG1001.catalogue"指明了该装配文档是基于设计文档 RDG1001.catalogue 而设计的。

下面的调度文档片断显示了二者之间在调度过程中的调度指令：

```
<Activity  xsi:type="schedule:Transfer"  Id="TransferPlanetPos.SunPos_SolarPer"
          Name="SunPos_SolarPer">
 <FieldLink
xlink:href="test.assembly#SolarPerturbationModel.SolarPerturbationModel_m_pSunPos_
_SunModel_m_pSunPos"
xlink:title="FieldLink SolarPerturbationModel_m_pSunPos__SunModel_m_pSunPos"/>
</Activity>
```

这段XML代码对装配文档test.assembly中建立的字段链接(下划线标明的代码)发出字段传送指令。

B.3 基于组件集成的装配

下面这段代码显示了 4.4 中提到的基于组件集成的模型中接口与实现模型之间的链接。我们以 EnvironmentModel 中引用的 ISatellite 接口与其实现模型之间的链接关系为例说明：

```
<Link xsi:type="assembly:InterfaceLink"
    Id="EnvironmentModel.EnvironmentModel_ISatelliteRef__SatelliteModel"
  Name="EnvironmentModel_ISatelliteRef__SatelliteModel">
  <Reference
    xlink:href="RDG1001.catalogue#RDG.RDGModel.EnvironmentSegment.Enviro
    nmentModel.ISatelliteRef" xlink:title="ISatelliteRef"/>
  <Provider xlink:href="test.assembly#SatelliteModel" xlink:title="SatelliteModel"/>
 </Link>
```

代码说明如下：

- <Reference>指明了引用的接口；
- <Provider>指明了接口的实现模型。

B.4 调度文档代码说明

(1)调度文档中 Task PlanetPos 的代码

这段代码显示了 Task PlanetPos 由三个 Activity 组成，Activity 可以是触发模型也可以是传输字段，在这段代码中的每个 Activity 都用于触发模型。以第一个 Activity 为例，它触发了 test.assembly 中装配的 SunModel 实例，指定其入口点为 computeEP()。代码的其它部分含义与此相同。

```
<Task Id="PlanetPos" Name="PlanetPos">
  <Activity xsi:type="schedule:Trigger" Id="PlanetPos.SunModel" Name="SunModel">
    <Provider xlink:href="test.assembly#SunModel" xlink:title="Model Instance SunModel"/>
    <EntryPoint xlink:href="RDG1001.catalogue#RDG.RDGModel.SolarSystem.SunModel.computeEP"
        xlink:title="EntryPoint computeEP"/>
  </Activity>
  <Activity xsi:type="schedule:Trigger" Id="PlanetPos.EarthModel" Name="EarthModel">
    <Provider xlink:href="test.assembly#EarthModel" xlink:title="Model Instance EarthModel"/>
    <EntryPoint xlink:href="RDG1001.catalogue#RDG.RDGModel.SolarSystem.EarthModel.computeEP"
        xlink:title="EntryPoint computeEP"/>
  </Activity>
  <Activity xsi:type="schedule:Trigger" Id="PlanetPos.MoonModel" Name="MoonModel">
    <Provider xlink:href="test.assembly#MoonModel" xlink:title="Model Instance MoonModel"/>
    <EntryPoint xlink:href="RDG1001.catalogue#RDG.RDGModel.SolarSystem.MoonModel.computeEP"
      xlink:title="EntryPoint computeEP"/>
  </Activity>
 </Task>
```

(2)调度文档中 Task TransferPlanet 的代码

```
<Task Id="TransferPlanetPos" Name="TransferPlanetPos">
  <Activity xsi:type="schedule:Transfer"
     Id="TransferPlanetPos.SunPos_SolarPer" Name="SunPos_SolarPer">
    <FieldLink xlink:href="test.assembly#
       SolarPerturbationModel.SolarPerturbationModel_m_pSunPos__SunModel_m_pSunPos"
       xlink:title="FieldLink SolarPerturbationModel_m_pSunPos__SunModel_m_pSunPos"/>
  </Activity>
  <Activity xsi:type="schedule:Transfer"
     Id="TransferPlanetPos.SunPos_SolarPre" Name="SunPos_SolarPre">
    <FieldLink xlink:href="test.assembly#
       SolarPressureModel.SolarPressureModel_m_pSunPos__SunModel_sunPos"
       xlink:title="FieldLink SolarPressureModel_m_pSunPos__SunModel_m_pSunPos"/>
  </Activity>
  <Activity xsi:type="schedule:Transfer"
     Id="TransferPlanetPos.MoonPos_MoonPer" Name="MoonPos_MoonPer">
    <FieldLink xlink:href="test.assembly#
       MoonPerturbationModel.MoonPerturbationModel_m_pMoonPos__MoonModel_m_pMoonPos"
       xlink:title="FieldLink MoonPerturbationModel_m_pMoonPos__MoonModel_m_pMoonPos"/>
  </Activity>
</Task>
```

这段代码显示了 Task TransferPlanet 中包含三个 Activity，这里的每个 Activity 都用于传输字段。以第一个 Activity 为例，它传输了在 test.assembly 中建立的字段链接。

(3)调度文档中 Epoch Event GeneralSchedule 的代码

这段代码显示了 Epoch Event GeneralSchedule 由 12 个组成，按从上到下的顺序执行。其中第一、二个 Task 的代码参见本节中的(1)和(2)。其它 Task 的形式与(1)、(2)类似，这里不再赘述。

```
<Event xsi:type="schedule:EpochEvent"
Id="GeneralSchdule" Name="GeneralSchdule" EpochTime="2007-10-15T12:40:50Z">
  <Task xlink:href="#PlanetPos" xlink:title="Task PlanetPos"/>
  <Task xlink:href="#TransferPlanetPos" xlink:title="Task TransferPlanetPos"/>
  <Task xlink:href="#Orbit_Perturbation" xlink:title="Task Orbit_Perturbation"/>
  <Task xlink:href="#ComputePerturbation" xlink:title="Task ComputePerturbation"/>
  <Task xlink:href="#Perturbation_Orbit" xlink:title="Task Perturbation_Orbit"/>
  <Task xlink:href="#ComputeOrbitRK" xlink:title="Task ComputeOrbitRK"/>
  <Task xlink:href="#ComputeClocks" xlink:title="Task ComputeClocks"/>
  <Task xlink:href="#SatelliteModel" xlink:title="Task SatelliteModel"/>
  <Task xlink:href="#EnvModelGet" xlink:title="Task EnvModelGet"/>
  <Task xlink:href="#ComputeEnvSegment" xlink:title="Task ComputeEnvSegment"/>
  <Task xlink:href="#EnvModelSet" xlink:title="Task EnvModelSet"/>
  <Task xlink:href="#ComputeStaion" xlink:title="Task ComputeStaion"/>
</Event>
```

(4)调度文档中 Epoch Event BasedSP3Simulation 的代码

```
<Event xsi:type="schedule:EpochEvent"
    Id="BasedSP3Simulation"
    Name="BasedSP3Simulation" EpochTime="2002-10-12T08:00:00Z">
  <Task xlink:href="#OrbitCompute" xlink:title="Task OrbitCompute"/>
  <Task xlink:href="#ClockBias" xlink:title="Task ClockBias"/>
  <Task xlink:href="#ComputeSatellite" xlink:title="Task ComputeSatellite"/>
  <Task xlink:href="#EnvModelGet" xlink:title="Task EnvModelGet"/>
  <Task xlink:href="#ComputeEnvironment" xlink:title="Task ComputeEnvironment"/>
  <Task xlink:href="#EnvModelSet" xlink:title="Task EnvModelSet"/>
  <Task xlink:href="#ComputeStaion" xlink:title="Task ComputeStaion"/>
</Event>
```

这段代码显示了 Epoch Event BasedSP3Simulation 由 7 个 Task 组成，每个 Task 的形式与本节中的(1)、(2)类似，这里不在赘述。