<基于SDN的卫星网络仿真系统>

软件架构文档

版本 <1.0>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **作者** |
| <14/11/2023> | <1.0> | 第一版 | 王劭、余彦、林彤彦、顾芷瑜、徐培凯 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

1. 简介 4

1.1 目的 4

1.2 参考资料 4

2. 用例视图 4

3. 逻辑视图 4

3.1 概述 4

3.2 在构架方面具有重要意义的设计包 4

4. 进程视图 4

5. 部署视图 4

6. 实现视图 5

7. 技术视图 5

8. 数据视图（可选） 5

9. 核心算法设计（可选） 5

10. 质量属性的设计 5

软件架构文档

# 简介

## 目的

本文档将从构架方面对系统进行综合概述，其中会使用多种不同的构架视图来描述系统的各个方面。它用于记录并表述已对系统的构架方面作出的重要决策。

本文是为了提供系统整体设计和组织结构的清晰概述，以便项目团队成员了解系统的各个方面、功能和模块之间的关系。该文档旨在为开发人员、测试人员、项目经理和其他利益相关者提供一个统一的参考点，以促进协作、理解和沟通。

本软件系统的架构设计文档包括用例视图、逻辑视图、进程视图、部署视图、实现视图、技术视图、数据视图、核心算法设计以及质量属性的设计和介绍。用例视图展示系统核心功能和构架范围，突出关键用例和场景。逻辑视图详细说明设计模型的重要部分，包括子系统、包、类、类实用程序、关系、操作和属性。概述部分阐述采用的架构风格、模块分解和协同情况，并通过逻辑架构图进行文字描述。进程视图描述系统的进程组织、通信模式和交互方式。部署视图说明系统在物理网络上的配置，包括硬件节点、互连情况和进程到节点的映射。实现视图展示整体结构、层次和子系统，并强调构架中的重要构件。技术视图介绍技术栈，包括编程语言、开发工具、框架、数据库和中间件。质量属性的设计解释如何通过软件架构实现性能、可扩展性、可靠性、易用性、可移植性等系统能力。文档还提供了数据视图和核心算法设计，根据系统需求可选包含。整体而言，该文档为系统开发人员、架构师和相关利益方提供了全面而清晰的系统架构概览。

开发人员可以通过逻辑视图、实现视图和技术视图深入了解系统的设计和实现细节。

测试人员可以通过用例视图和质量属性设计了解系统的功能需求和质量测试标准。

部署人员可以通过部署视图了解系统的部署环境和硬件配置。

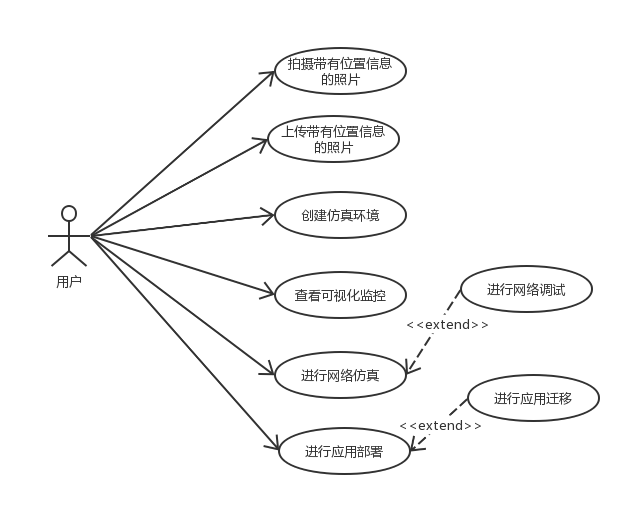
项目经理可以通过全文了解系统的整体结构和设计决策，以更好地管理项目进展和风险。

软件架构文档的编写和维护有助于确保系统的一致性、可维护性和可扩展性，为团队提供一个共享的知识库，以促进更高效的开发和协作。

## 参考资料

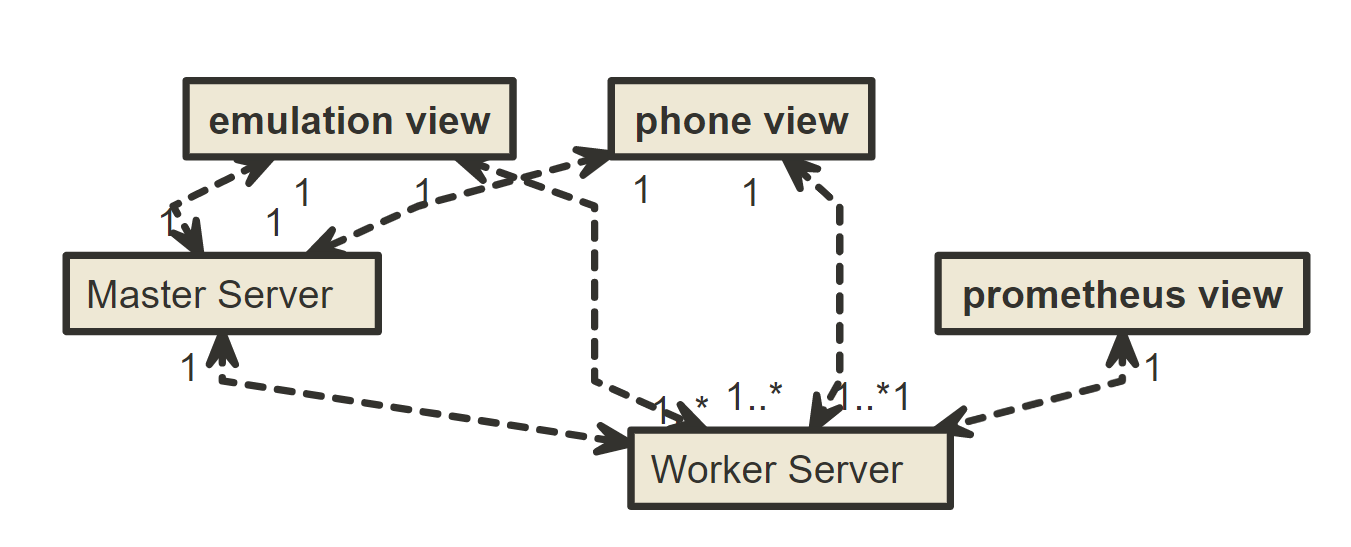
1. 《深入剖析Kubernetes》，张磊
2. 《Android开发精要》，范怀宇
3. Lai Z, Li H, Deng Y, et al. {StarryNet}: Empowering Researchers to Evaluate Futuristic Integrated Space and Terrestrial Networks[C]//20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). 2023: 1309-1324.
4. ers to Evaluate Futuristic Integrated Space and Terrestrial Networks[C]//20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23). 2023: 1309-1324.

# 用例视图



# 逻辑视图

## 概述

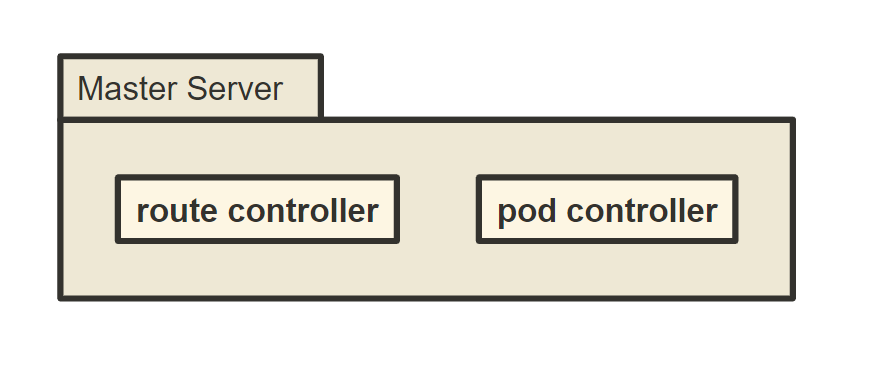


本项目由5个模块构成。其中emulation view、phone view和prometheus view分别表示卫星网络仿真展示界面、手机展示界面和prometheus数据监控界面。Master Server和Worker Server为服务器，且Master Server有且只有唯一一台，Worker Server可以有多台。

和Model模块模块在

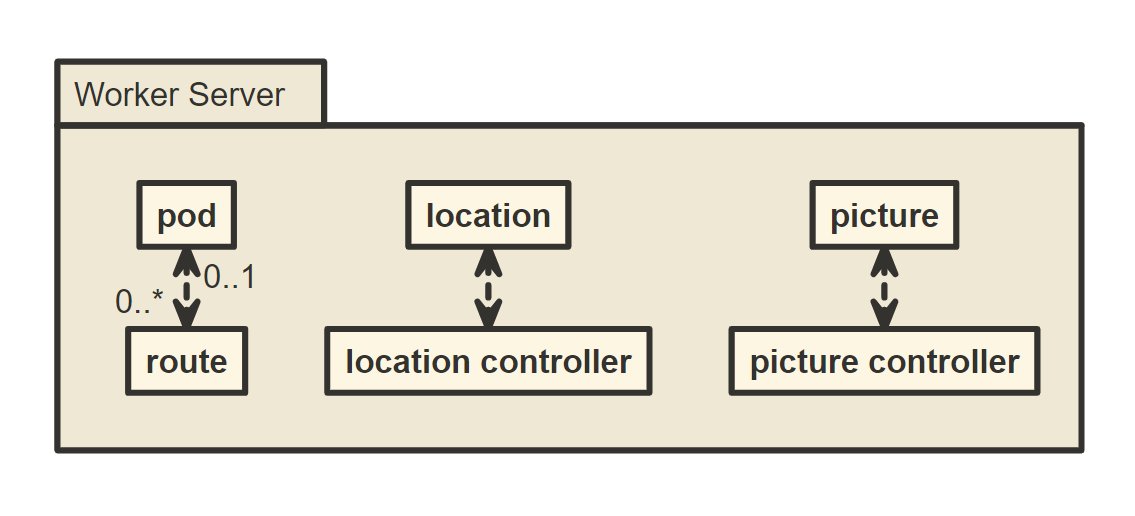
## 在构架方面具有重要意义的设计包

### Master Server



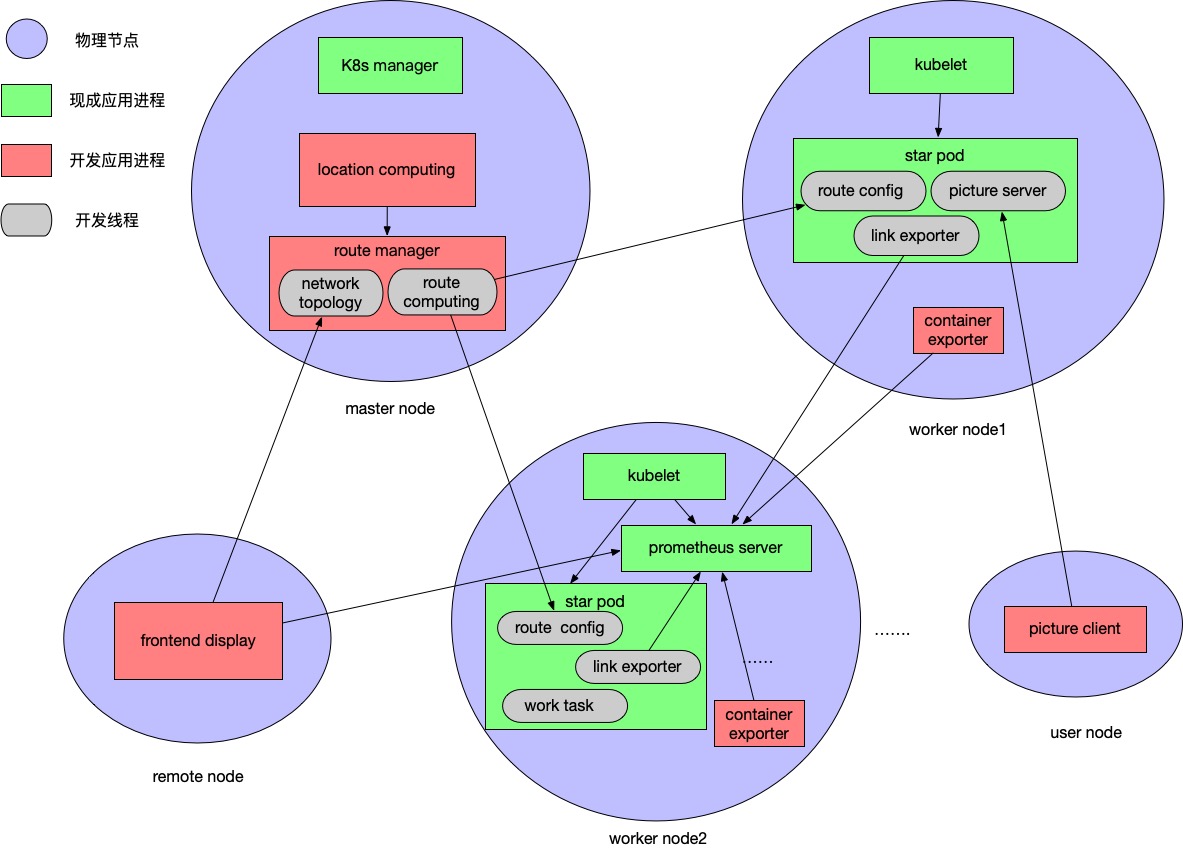
Master Server中有两个控制器，分别处理pod和route。

### Worker Server



在Worker Server中，既包含pod和route的模型，又包含location和picture的模型和对应的控制器。

# 进程视图



重量级进程： （1） 卫星Pod进程

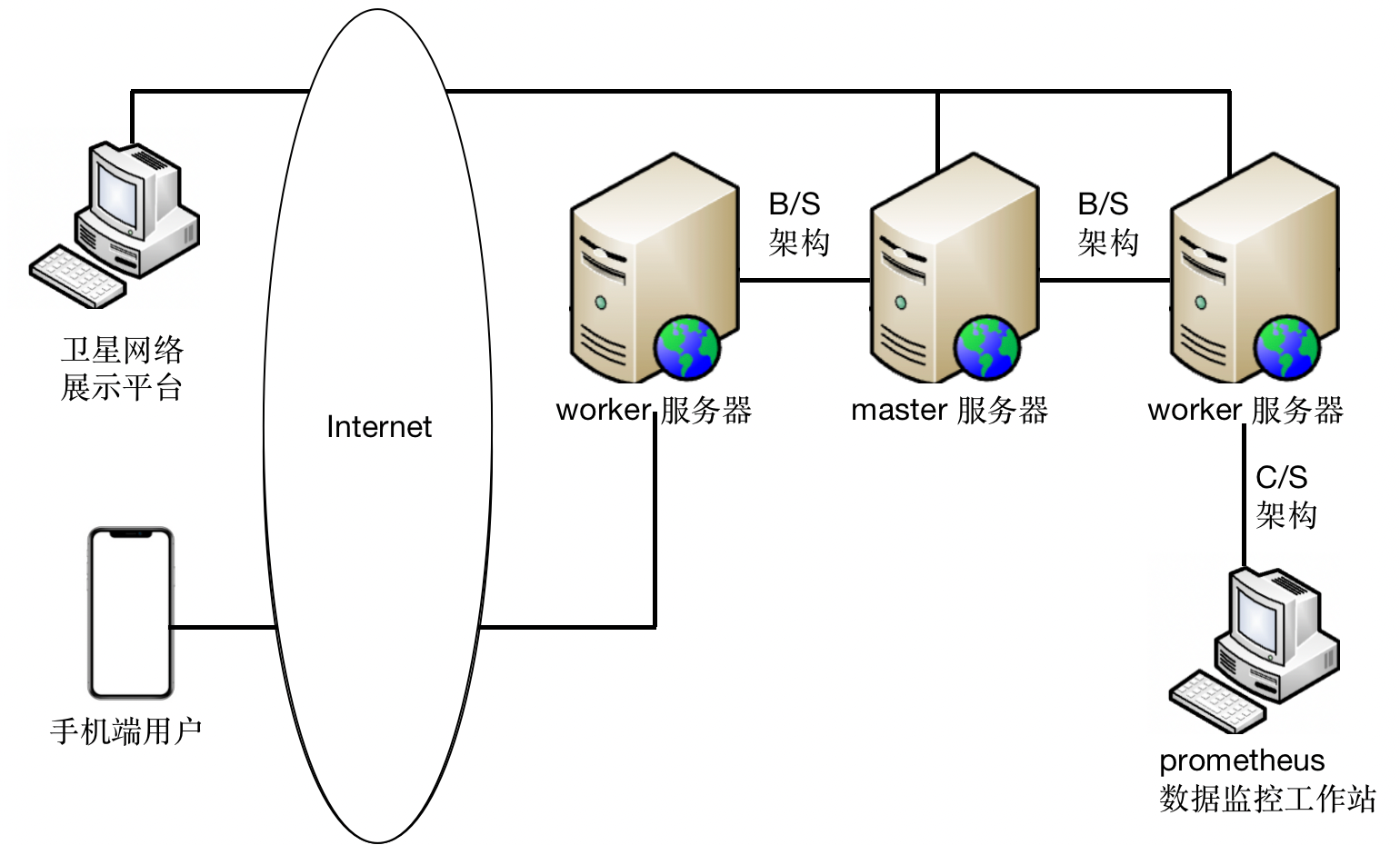
（2） 路由管理器进程

（3） prometheus服务进程

轻量级进程: （1） 容器采集exporter进程

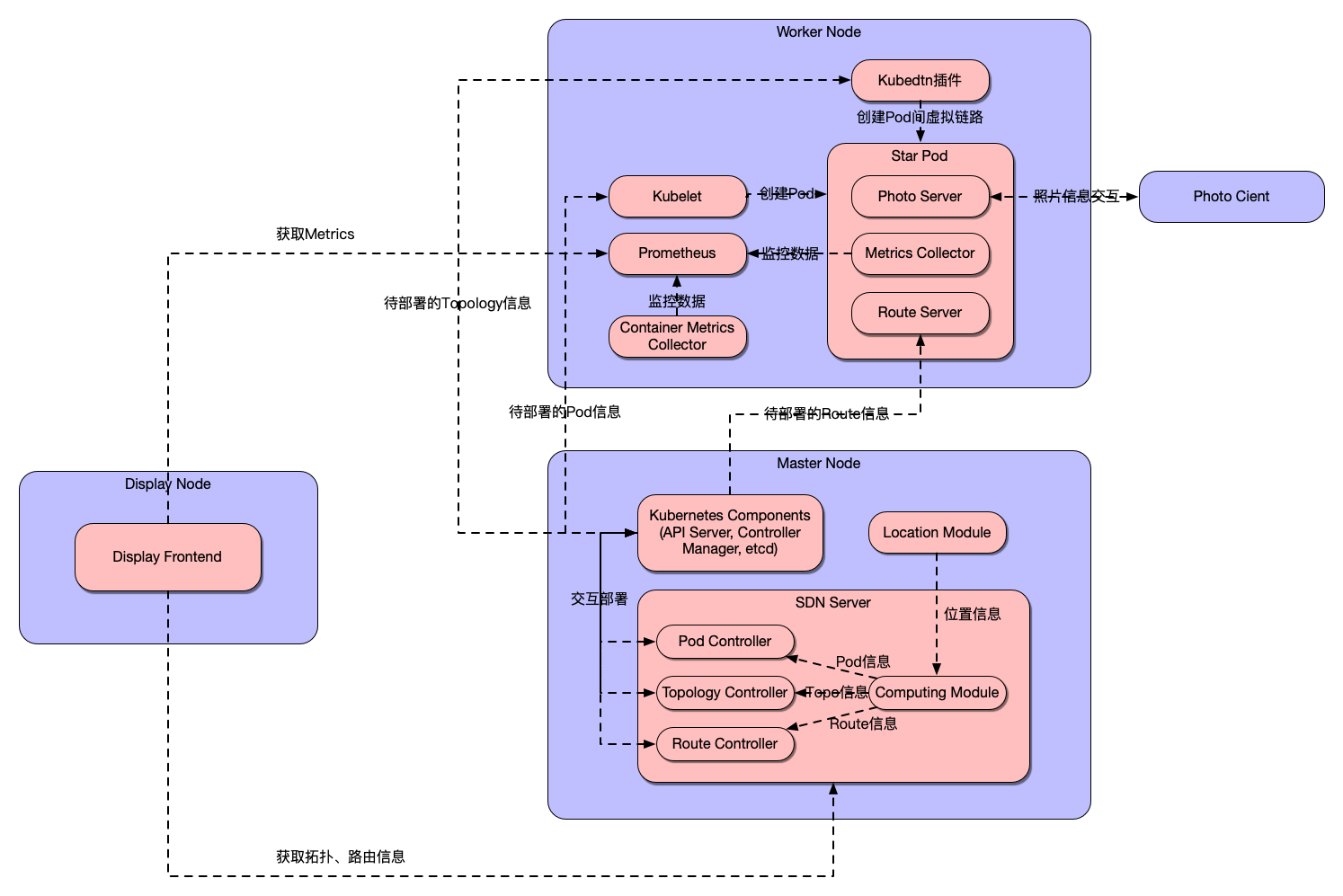
1. 卫星前端展示进程
2. 手机端进程

# 部署视图



master服务器运行路由管理模块，其计算出的拓扑通过网络传到卫星网络展示平台，将卫星的网络拓扑信息展示出来，而woker服务器上的prometheus服务为其提供对应容器的资源占用情况与链路情况，此外还有的worker服务器运行手机端拍照上传服务对应的后端服务，获取手机端的照片并存储和传输。

# 实现视图



# 技术视图

编程语言：Go、Java、Dart

开发工具：vscode、Intellij IDEA、Android Studio

框架：k8s Operator、Flutter、Springboot

# 数据视图（可选）

# 核心算法设计（可选）

# 质量属性的设计

在系统架构设计中，质量属性的设计是至关重要的，它直接影响着系统的性能、可靠性、可维护性等方面。

**10.1性能**

首先，性能方面的设计旨在确保系统具有高效的响应时间和吞吐量，通过合理的资源管理和优化算法来提升用户体验。

在性能方面，拍照发送照片和位置的App实现一下功能：

1. 平均时间3ms，最长时间5ms

2.系统每秒应该处理100个以上事务 进阶版本中每秒应处理200个以上事务

3.系统支持50个并发用户，进阶版本中应支持5000个用户

4.内存占用不超过32M，硬盘占用不超过200M

卫星网络仿真：界面渲染平均时间为0.2s

**10.2可扩展性**

可扩展性是另一个关键方面，系统应能够在不同负载下实现水平或垂直的扩展，以应对未来的业务需求。

节点扩展性： 能够通过k8s起pod，轻松地增加仿真模型中的卫星节点数量，以应对对系统规模的扩大。这包括水平扩展（增加节点的数量）和垂直扩展（增加每个节点的处理能力），确保系统在不同规模的卫星网络仿真中都能有效运行。

通信通道扩展性： 允许在仿真中模拟更多的通信通道，以适应更复杂的卫星通信场景。这包括增加卫星之间的通信链路、地面站与卫星之间的通信等。

仿真对象的扩展性：设计了包括低轨卫星，高轨卫星，地面站，导弹，无人机等的数据结构，便于后续相应数据的导入，进行仿真计算。

模型参数化和定制扩展性：设计了相应的接口允许用户根据特定需求定制仿真模型的参数，以适应不同的卫星网络配置和特性，从而提高模型的通用性。

**10.3可靠性与可维护性**

在可靠性方面，系统的设计考虑到容错机制和恢复策略，以确保在面临硬件故障或其他异常情况时，系统能够保持稳定运行。

(1)容错机制：引入容错机制，以处理仿真过程中可能发生的错误，确保系统在异常情况下能够继续运行或迅速恢复。

(2)数据备份与恢复：实施定期的数据备份策略，以防止数据丢失，并提供快速的数据恢复机制，确保系统数据的完整性。

(3)稳定性测试：进行系统稳定性测试，包括长时间运行测试和大规模仿真测试，以验证系统在不同负载和运行时间条件下的可靠性。

可维护性方面的设计旨在降低系统维护的难度，通过清晰的代码结构、文档化和模块化设计来提高代码的可读性和可理解性。

(1)模块化设计：采用模块化的设计，使系统的各个组件相互独立，易于理解和修改。这有助于简化维护工作，降低引入新功能或修复错误的风险。

(2)文档化：提供详细而清晰的文档，包括系统架构、模型说明、代码注释等。这有助于团队成员理解系统，并为日后的维护和更新提供支持。

(3)版本控制：使用版本控制系统，如Git，以跟踪代码变更并管理不同版本。这样可以方便地回滚到之前的稳定版本，同时促进多人协作和团队开发。

(4)远程维护和更新：考虑实现远程维护和更新功能，使管理员能够远程管理系统、应用更新和修复漏洞，减少维护的时间和成本。

(5)自动化测试：实施自动化测试，包括单元测试、集成测试和系统测试，以确保修改或新增的代码不会引入新的错误，并提高维护过程中的代码质量。

(6)定期维护计划 制定定期的维护计划，包括系统巡检、性能优化、安全性检查等，以保持系统在长期运行中的稳定性和可靠性。

目前，实现了一下方面的功能：

可用性：总的有效运行时间 / 总运行时间，要求达到90%，要求平均每天可以使用23小时以上，在系统维护时会关闭软件的部分功能，进行降级模式操作，尽可能减少对用户体验感的影响。

平均故障间隔时间应不低于2000h。

平均修复时间：2小时

精确度：误差小于5%

最高错误或缺陷率：bugs/KLOC为3个

最高错误或缺陷率低于1%

**10.4易用性**

易用性方面的设计注重用户界面的友好性和交互体验，以确保用户能够轻松地使用系统，并通过良好的反馈机制提高用户满意度。可移植性方面的设计考虑到系统在不同平台和环境中的可部署性，以便在不同的硬件和操作系统上运行而不受影响。

因此，系统的卫星网络仿真功能要求有卫星网络相关知识储备的高级用户可以通过查看说明文档清楚地知道如何使用。系统的拍照发送照片和位置的App要求普通的用户不用经过经过培训也可以知道功能大致如何使用。系统在集群上部署应用的功能对于有docker和kubernates使用经验的高级用户可以快速理解，对于没有相关经验的用户能够通过学习说明文档而快速上手。

系统的拍照发送照片和位置的App类似于微信发朋友圈的功能和界面。系统在集群上部署应用的功能类似于kubernates对应的功能。

一般用户访问（CUA）标准是要提出标准按键和鼠标方式的概念，即运行任何程序时，按键和鼠标的使用都有固定的意义。

**10.5安全性**

最后，安全性是质量属性设计中不可或缺的一部分。系统具备了严密的身份验证、访问控制和数据加密机制，以保护用户数据和系统免受潜在的安全威胁。

数据保密性： 保护仿真模型中涉及到的敏感数据，例如通信内容、任务信息等，确保只有授权人员能够访问这些数据。采用数据加密技术，以防止未经授权的访问或信息泄露。

身份认证和访问控制： 实施有效的身份认证机制，确保只有经过授权的用户能够访问和修改仿真模型。使用访问控制策略，限制用户对系统资源和仿真数据的访问权限。

网络安全： 采取必要的安全措施，防范网络攻击，包括但不限于防火墙、入侵检测系统（IDS）、虚拟专用网络（VPN）等，以保护仿真模型在网络中的传输过程中的安全性。

通过这些质量属性的设计，系统可以在不同维度上得到全面的保障，为用户提供高性能、可靠性、可维护性、易用性、可移植性和安全性的全面保障。