软件工程实验

面向Apache的个人助手

|  |  |
| --- | --- |
| 队伍 | 弟归（GroupA） |
| 成员 | 王嘉凯 |
| 成员 | 王宁 |
| 成员 | 叶俊辰 |
| 成员 | 姬索肇 |
| 成员 | 韩慧敏 |

2019年 3 月

声明

本文档为软件工程实验文档，团队成员在编辑本文档时应遵守如下约定：

* 文档中中文正文字体一律使用宋体，章标题、节标题、图注、表头使用黑体，英文字体一律使用Times New Roman
* 文档中正文的字体字号为小四，图注、表头字号为小四，表格内容字号为五号
* 表格和图片的命名格式统一为：表1.1、图1.1样式
* 一级标题字号为小二，二级标题字号为小三，三级标题为小四，标题层次不宜超过三层
* 正文内容行距为1.5行距，段首首行缩进，缩进值为2字符

版本变更记录

|  |  |  |
| --- | --- | --- |
| 日期 | 变更人 | 变更内容 |
| 2019/3/8 | 王嘉凯 | 完成了软件工程试验文档创建 |
| 2019/3/11 | 王嘉凯 | 修改了需求文档组织结构 |
| 2019/3/11 | 王嘉凯 | 撰写了文档第一节内容以及第二、三节部分内容 |
| 2019/3/12 | 姬索肇 | 完成了功能需求outline |
| 2019/3/12 | 叶俊辰 | 完成第二节项目业务需求， |
| 2019/3/12 | 王嘉凯 | 完成第三节项目用户需求 |
| 2019/3/12 | 王宁 | 完成项目功能需求整体用例图 |
| 2019/3/12 | 姬索肇 | 完成项目功能需求RUCM |
| 2019/3/12 | 韩慧敏 | 完成项目功能需求文字描述 |
| 2019/3/13 | 王嘉凯 | 完成项目结构修改以及业务流程系统框架、Apache框架图 |
| 2019/3/13 | 姬索肇 | 完成第六节工作重点 |
| 2019/3/13 | 韩慧敏 | 完成框架文字描述 |
| 2019/3/13 | 叶俊辰 | 修改第一节项目背景内容，并整合文档 |
| 2019/3/13 | 韩慧敏 | 完成性能需求部分的修改 |

备注：任一成员在完成了本文档的编辑后应该在版本变更记录中登记

目录

[一、 项目背景 7](#_Toc3447933)

[1.1 项目概述 7](#_Toc3447934)

[1.2 项目目的 8](#_Toc3447935)

[1.3 软件需求分析目的 8](#_Toc3447936)

[1.4 文档概述 8](#_Toc3447937)

[1.5 术语和缩略语 9](#_Toc3447938)

[1.6 用户定义 9](#_Toc3447939)

[二、 项目业务需求 9](#_Toc3447940)

[2.1 ApacheAssistant业务需求概述 9](#_Toc3447941)

[2.2 便捷化Apache项目的管理 11](#_Toc3447942)

[2.3 提供Apache一体化管理监控 11](#_Toc3447943)

[2.4 减少Apache的上手难度 12](#_Toc3447944)

[2.5 增强Apache项目的可维护性 12](#_Toc3447945)

[三、 项目用户需求 12](#_Toc3447946)

[3.1 用户需求概述 12](#_Toc3447947)

[3.2 配置管理 12](#_Toc3447948)

[3.3 日志管理 13](#_Toc3447949)

[3.4 性能监控 13](#_Toc3447950)

[3.5 模块管理 13](#_Toc3447951)

[四、 项目功能需求 14](#_Toc3447952)

[4.1 项目功能需求概述 14](#_Toc3447953)

[4.2 配置管理 16](#_Toc3447954)

[4.2.1 设置配置文件路径 16](#_Toc3447955)

[4.2.2 配置项修改 17](#_Toc3447956)

[4.3 日志管理 18](#_Toc3447957)

[4.3.1 设置日志文件路径 18](#_Toc3447958)

[4.3.2 解析日志文件 19](#_Toc3447959)

[4.3.3 根据时间段展示 20](#_Toc3447960)

[4.3.4 根据HTTP请求展示 21](#_Toc3447961)

[4.3.5 根据IP地址展示 22](#_Toc3447962)

[4.3.6 根据文件类型展示 23](#_Toc3447963)

[4.3.7 根据关键词展示 23](#_Toc3447964)

[4.3.8 备份日志内容 24](#_Toc3447965)

[4.3.9 清除日志内容 24](#_Toc3447966)

[4.4 性能监控 25](#_Toc3447967)

[4.4.1 查看系统状态 26](#_Toc3447968)

[4.4.2 停止Apache HTTP Server 26](#_Toc3447969)

[4.4.3 启动Apache HTTP Server 27](#_Toc3447970)

[4.4.4 监控性能指标 27](#_Toc3447971)

[4.5 模块管理 28](#_Toc3447972)

[4.5.1 查看模块信息 29](#_Toc3447973)

[4.5.2 安装模块 29](#_Toc3447974)

[4.5.3 卸载模块 30](#_Toc3447975)

[五、 项目非功能需求 30](#_Toc3447976)

[5.1 项目非功能需求概述 30](#_Toc3447977)

[5.2 运行环境需求 30](#_Toc3447978)

[5.2.1 操作系统 30](#_Toc3447979)

[5.2.2 编译环境 31](#_Toc3447980)

[5.2.3 Apache版本要求 31](#_Toc3447981)

[5.3 性能需求 31](#_Toc3447982)

[5.3.1 兼容性 31](#_Toc3447983)

[5.3.2 可靠性 31](#_Toc3447984)

[5.3.3 可用性 31](#_Toc3447985)

[5.3.4 稳定性 31](#_Toc3447986)

[5.3.5 易维护性 31](#_Toc3447987)

[5.3.6 可扩展性 32](#_Toc3447988)

[六、 工作重点 32](#_Toc3447989)

[6.1 工作内容 32](#_Toc3447990)

[6.2 技术路线 32](#_Toc3447991)

[6.2.1 解析文件并提供接口 32](#_Toc3447992)

[6.2.2 界面调用接口实现具体功能 33](#_Toc3447993)

1. 项目背景
   1. 项目概述

Apache HTTP Server（简称Apache）是Apache软件基金会的一个开放源码的网页服务器，可以在大多数计算机操作系统中运行，由于其多平台和安全性被广泛使用，是最流行的Web服务器端软件之一。



它快速、可靠并且可通过简单的API扩展，将Perl/Python等解释器编译到服务器中。它可以运行在几乎所有广泛使用的计算机平台上。Apache源于NCSAhttpd服务器，经过多次修改，在不断加入的开发者的持续迭代下，Apache功能不断强大，特性越来越好，缺陷逐步修复，目前是世界上最流行的Web服务器软件之一。Apache的结构如图 1所示：



图 1 Apache结构

ApacheAssistant是一款面向开发者的Web应用程序，它通过可视化的方式帮助用户更好地上手Apache服务器，同时可以帮助用户监听Apache服务器的相关性能指标，是一款帮助开发者更好地管理服务器，提升开发效率的工作应用。

* 1. 项目目的

以Apache开源框架及相关资料为输入，分析软件设计需求，结合软件工程综合实验具体要求，输出基于Apache开源框架的管理软件的软件需求规格说明书，作为设计开发的依据并指导后续的开发工作。

* 1. 软件需求分析目的

软件需求分析(Software Requirement Analysis)是对软件预期想要实现的业务目标进行分析，在明确定义和描述用户的业务目标及与之对应的具体业务需求的基础上，进一步确定并详细描述对应的软件功能性需求和非功能性需求等内容。

需求分析的具体内容可以归纳为五个方面：软件的总体概述，软件的业务需求，软件的用户需求，软件的功能需求，软件的非功能性需求，软件的运行要求。

* 1. 文档概述

文档用途：本文档主要是介绍ApacheAssistant系统需求及规格说明。

主要内容如下：

1. 描述了ApacheAssistant系统的业务需求；
2. 以用例图的形式给出ApacheAssistant系统功能需求，并对用例模型进行详细的描述；
3. 用RUCM模型对功能需求进行建模；
4. 描述了ApacheAssistant的非功能性需求；
5. 描述了与ApacheAssistant实施相关的硬件环境的一些要求；
6. 描述了与ApacheAssistant实施相关的软件环境的要求。
   1. 术语和缩略语

|  |  |  |
| --- | --- | --- |
| 编号 | 术语 | 说明 |
| 1 | HTTP | 超文本传输协议 |
| 2 | IP | 互联网协议地址 |
| 3 | 接口 | API，是一些预先定义的函数，目的是提供应用程序与开发人员基于某软件或硬件得以访问一组例程的能力，而又无需访问源码，或理解内部工作机制的细节 |
| 4 | RUCM | 限制性用例模型 |
| 5 | UCM | 用例建模 |
| 6 | 日志 | 网络设备、系统及服务程序等，在运作时产生的log事件记录 |
| 7 | Python | 一种面向对象的解释型计算机程序设计语言 |

* 1. 用户定义

ApacheAssistant的用户定义使用Apache的开发人员。

1. 项目业务需求
   1. ApacheAssistant业务需求概述

业务需求（Business requirement）表示组织或客户高层次的目标。业务需求通常来自项目投资人、购买产品的客户、实际用户的管理者、市场营销部门或产品策划部门。业务需求描述了组织为什么要开发一个系统，即组织希望达到的目标。使用前景和范围（vision and scope）文档来记录业务需求，这份文档有时也被称作项目轮廓图或市场需求（project charter 或 market requirement）文档。

在本项目中，ApacheAssistant需要帮助用户更便捷的使用Apache服务器，让用户可以不必直接操作Apache相对而言比较晦涩的系统，而通过图形化界面直观而简便的对Apache进行操作。ApacheAssistant的框架如图 2所示。



图 2 ApacheAssistant的框架图

ApacheAssistant由底层API和图形界面两部分组成，底层API与Apache HTTP Server交互，对Apache HTTP Server的配置文件、日志文件以及Apache HTTP Server的各项性能指标进行读取、解析、记录修改，根据各项功能的具体需求封装为可供界面调用的底层API接口。图形管理界面通过调用底层API实现配置管理、日志管理、性能监控、模块管理等具体的功能，以便开发者使用。

一个简单的业务流程如图 3所示。



图 3 业务流程图

开发者在使用ApacheAssistant时通过图形管理界面调用配置管理、日志管理、性能监控、模块管理等功能，做出一系列的操作，然后图形管理界面根据用户操作调用底层API，实现对Apache HTTP Server的配置文件、日志文件的增删改查操作。

* 1. 便捷化Apache项目的管理

Apache虽然有着较好的安全性，提供用户会话过程的跟踪，运行相对稳定等一系列优点，但是Apache的管理方法杂乱，配置文件数量繁多，位置难以定位。如果能在简单的交互界面上直观的修改Apache的配置文件达到用户的管理目的，会大大节省此方面的时间。

* 1. 提供Apache一体化管理监控

在一个Apache项目周期中，用户的需求不止在于修改配置文件上，常常需要管理模块，性能监控，管理日志等等一系列的操作，这些不同的需求操作的方法也不尽相同，繁多的需求和解决方法带来了大量的人力资源的浪费。如果能有一个统一化的服务平台，将开发维护中的需求集合在一起，能使整个项目的管理变得清晰。

* 1. 减少Apache的上手难度

Apache是世界使用排名第一的Web服务器软件。它可以运行在几乎所有广泛使用的计算机平台上，由于其跨平台和安全性被广泛使用，是最流行的Web服务器端软件之一。对于新手来说却难以上手，配置文件的数量和修改方式繁多，通常让初学者晕头转向，我们提供简洁大方的交互界面，让新手能对Apache项目的更快地上手。

* 1. 增强Apache项目的可维护性

Apache日志生成冗长，当用户要查看日志的时候常常发现日志丢失，或者难以找到自己所需求查看的信息，同时Apache在运行时用户可能需求对性能、模块等进行管理，Apache需要加强本身项目的可维护性。

1. 项目用户需求
   1. 用户需求概述

用户需求（User requirement）描述的是用户的目标，或用户要求系统必须能完成的任务。ApacheAssistant应用的主要用户是使用Apache进行完成相关任务的生产开发者。

生产开发者希望通过本应用达成一定的目的，例如通过图形化界面更好地管理Apache服务器，配置文件的查看、修改，对Apache日志的管理等。项目开发者希望能快速有效地参与到本项目中，要求项目文档、源码的可读性强。

* 1. 配置管理

由于Apache的配置项类目繁多，配置文件数目较多，用户使用ApacheAssistant时，能够通过应用对Apache HTTP Server的配置进行管理，主要包括查询当前配置，修改当前配置，设置配置参数。用户可以设置Apache的配置文件路径和配置文件内容，可以设置监听端口和协议，可以设置网页目录的设置Keep Alive开关和超时时间，可以设置服务器最大连接数，可以设置Apache Request超时时间，可以修改HTTP Header的内容，可以设置日志记录的格式。

* 1. 日志管理

用户在使用AapcheAssistant时，要能够对Apache生成的日志进行查询和管理。Apache日志文件的路径能够通过ApacheAssistant手动设置。其次，ApacheAssistant可以根据关键词对生成的日志进行搜索，有序地呈现搜索的内容。ApacheAssistant还应该支持多种关键词搜索，如时间段，HTTP请求，IP地址，文件类型等等。对于生成的日志，ApacheAssistant能够进行备份，清楚指定内容。

* 1. 性能监控

用户需要对Apache HTTP Server进行管理和性能的监控。用户能够通过ApacheAssistant得知当前系统的状态（Apache HTTP Server是启动还是停止），能够停止、重启当前的Apache HTTP Server。ApacheAssistant对Apache HTTP Server进行监控，给用户呈现Apache HTTP Server运行的基本情况。监控的内容包括并发连接数，网络速度，CPU占用率，内存占用率等等。

* 1. 模块管理

Apache官方文档给出的标准模块数量多达129个，其中核心模块8个，而常用模块也有数十个，熟悉并熟练使用如此多的模块对于用户而言是非常大的工作量，同时模块类型也分为两类，包括静态和动态两种，其中静态模块不能卸载。用户存在便捷管理模块的客观需要，因此，ApacheAssistant需要帮助用户便捷安装需要的模块或者卸载不再需要的模块，需要查询当前Apache服务器的模块状态。

1. 项目功能需求
   1. 项目功能需求概述

ApacheAssistant的主要功能需求包括：配置管理，日志管理，性能监控和模块管理。ApacheAssistant的用例图如下图 4所示：

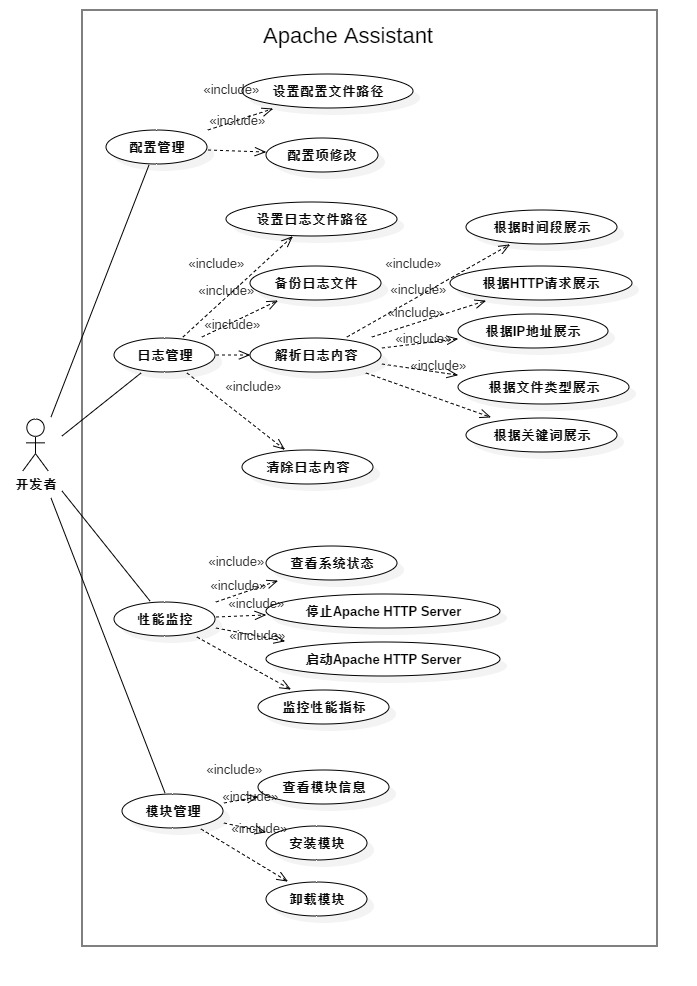


图 4 用例图

下面使用RUCM模型描述用例，通过RUCM模型能够对用例进行规范的描述。RUCM即限制性用例建模，它的目标是：

1. 使UCMs更加可理解并且更精确。
2. 从UCMs自动生成分析模型。

RUCM由以下两部分组成：

1. 一个用于系统组织UCSs的用例模板。
2. 限制用户写UCSs的一系列规则。
   1. 配置管理



图 5 配置管理

开发者可以通过ApacheAssistant对Apache HTTP Server进行配置管理，配置管理的一般流程如下：首先开发者进入ApacheAssistant的配置管理模块，之后开发者根据自己的需求进行配置文件路径的设置或对配置项进行修改，配置管理模块实现开发者的需求之后，开发者关闭配置管理模块即可。

* + 1. 设置配置文件路径



图 6 设置配置文件路径

开发者可以通过ApacheAssistant设置配置文件路径，设置配置文件路径的一般流程如下：首先开发者进入配置管理模块，然后开发者设置配置文件路径，然后ApacheAssistant验证设置的配置文件路径是否存在，若文件路径不存在，则配置文件路径设置失败，若文件路径存在，则ApacheAssistant将开发者设置的配置文件路径保存。

* + 1. 配置项修改



图 7 配置项修改

开发者可以通过ApacheAssistant进行配置项修改，配置项修改的一般流程如下：首先开发者进入配置管理模块，然后开发者选择修改修改的配置项，包括设置监听端口及协议、设置网页目录位置、设置Keep Alive开关及超时时间、设置最大连接数、设置Request超时时间、设置HTTP Header或设置日志记录格式，之后开发者修改某些配置项的内容，然后开发者点击保存按钮，ApacheAssistant将配置项的修改内容保存。

* 1. 日志管理



图 8 日志管理

开发者可以通过ApacheAssistant对Apache运行过程中生成的日志进行管理，日志管理的一般流程如下：首先开发者进入ApacheAssistant的日志管理模块，之后开发者根据自己的具体日志管理需求进行日志文件路径的设置、日志文件的解析、日志内容的备份或日志内容的清除，日志管理模块实现开发者的日志管理需求之后，开发者关闭日志管理模块即可。

* + 1. 设置日志文件路径



图 9 设置日志文件路径

开发者可以通过ApacheAssistant设置日志文件路径，设置日志文件路径的一般流程如下：首先开发者进入日志管理模块，然后开发者设置日志文件路径，然后ApacheAssistant验证该日志文件路径是否存在，若文件路径不存在，则日志文件路径设置失败，若文件路径存在，则ApacheAssistant将开发者设置的日志文件路径保存。

* + 1. 解析日志文件



图 10 解析日志文件

开发者可以通过ApacheAssistant解析日志文件，解析日志文件的一般流程如下：首先开发者进入日志管理模块，然后开发者使用解析日志文件功能，之后开发者根据自己的解析需求对日志文件内容根据时间段展示、根据HTTP请求展示、根据IP地址展示、根据文件类型展示或者根据关键词展示日志内容，之后系统根据开发者需求展示解析出的日志文件内容。

* + 1. 根据时间段展示



图 11 根据时间段展示

开发者可以通过ApacheAssistant根据时间段展示日志内容，根据时间段展示日志内容的一般流程如下：首先开发者进入日志管理模块，之后开发者使用解析日志文件功能，之后开发者选取时间段的开始时间和结束时间，ApacheAssistant验证开始时间是否早于结束时间，若结束时间早于开始时间，则根据时间段展示日志内容失败，若开始时间早于结束时间，ApacheAssistant将日志内容根据时间段展示。

* + 1. 根据HTTP请求展示



图 12 根据HTTP请求展示

开发者可以通过ApacheAssistant根据HTTP请求展示日志内容，根据HTTP请求展示日志内容的一般流程如下：首先开发者进入日志管理模块，之后开发者使用解析日志文件功能，之后开发者选取HTTP请求，之后ApacheAssistant将日志内容根据HTTP请求展示。

* + 1. 根据IP地址展示



图 13 根据IP地址展示

开发者可以通过ApacheAssistant根据IP地址展示日志内容，根据IP地址展示日志内容的一般流程如下：首先开发者进入日志管理模块，之后开发者使用解析日志文件功能，之后开发者设置IP地址，之后ApacheAssistant将日志内容根据IP地址展示。

* + 1. 根据文件类型展示



图 14 根据文件类型展示

开发者可以通过ApacheAssistant根据文件类型展示日志内容，根据文件类型展示日志内容的一般流程如下：首先开发者进入日志管理模块，之后开发者使用解析日志文件功能，之后开发者设置文件类型，之后ApacheAssistant将日志内容根据文件类型展示。

* + 1. 根据关键词展示



图 15 根据关键词展示

开发者可以通过ApacheAssistant根据关键词展示日志内容，根据关键词展示日志内容的一般流程如下：首先开发者进入日志管理模块，之后开发者使用解析日志文件功能，之后开发者输入关键词，之后ApacheAssistant将日志内容根据关键词进行搜索并展示。

* + 1. 备份日志内容



图 16 备份日志内容

开发者可以通过ApacheAssistant备份日志内容，备份日志内容的一般流程如下：首先开发者进入日志管理模块，之后开发者使用备份日志内容功能，之后开发者设置备份日志文件路径并指定需备份日志内容的时间段，之后ApacheAssistant将需备份的日志内容备份保存至设置好的文件路径。

* + 1. 清除日志内容



图 17 清楚日志内容

开发者可以通过ApacheAssistant清除日志内容，清除日志内容的一般流程如下：首先开发者进入日志管理模块，之后开发者使用清除日志内容功能，之后开发者设置需要被清除的日志文件路径并指定需要被清除日志的时间段，之后ApacheAssistant将需要被清除的日志内容清除。

* 1. 性能监控



图 18 性能监控

开发者可以通过ApacheAssistant在Apache运行过程中进行性能监控，性能监控的一般流程如下：首先开发者进入ApacheAssistant的性能监控模块，之后开发者根据自己的需求查看系统状态、停止Apache HTTP Server、启动Apache HTTP Server或监控性能指标，之后性能监控模块满足开发者需求，开发者关闭性能监控模块即可。

* + 1. 查看系统状态



图 19 查看系统状态

开发者可以通过ApacheAssistant查看系统状态，查看系统状态的一般流程如下：首先开发者进入性能监控模块，之后开发者使用查看系统状态功能，之后ApacheAssistant将Apache HTTP Server的运行状态展示给开发者。

* + 1. 停止Apache HTTP Server



图 20 停止Apache HTTP Server

开发者可以通过ApacheAssistant停止Apache HTTP Server，停止Apache HTTP Server的一般流程如下：首先开发者进入性能监控模块，之后开发者将Apache HTTP Server的运行状态设置为停止，之后ApacheAssistant停止运行Apache HTTP Server。

* + 1. 启动Apache HTTP Server



图 21 启动Apache HTTP Server

开发者可以通过ApacheAssistant启动Apache HTTP Server，启动Apache HTTP Server的一般流程如下：首先开发者进入性能监控模块，之后开发者将Apache HTTP Server的运行状态设置为启动，之后ApacheAssistant启动Apache HTTP Server。

* + 1. 监控性能指标



图 22 监控性能指标

开发者可以通过ApacheAssistant监控性能指标，监控性能指标的一般流程如下：首先开发者进入性能监控模块，之后开发者查看Apache HTTP Server的并发连接数、网络速度、CPU占用、内存占用等性能指标，之后ApacheAssistant将相应的性能指标展示给开发者。

* 1. 模块管理



图 23 模块管理

开发者可以通过ApacheAssistant在Apache运行过程中进行模块管理，模块管理的一般流程如下：首先开发者进入ApacheAssistant的模块模块，之后开发者根据自己的需求查看模块信息、安装模块或卸载模块，之后模块管理模块满足开发者需求，开发者关闭模块管理模块即可。

* + 1. 查看模块信息



图 24 查看模块信息

开发者可以通过ApacheAssistant查看模块信息，查看模块信息的一般流程如下：首先开发者进入模块管理模块，之后开发者使用查看模块信息功能查看模块名称、模块类型、模块文件位置、模块状态、模块说明等模块信息，之后ApacheAssistant将相关模块信息展示给开发者。

* + 1. 安装模块



图 25 安装模块

开发者可以通过ApacheAssistant安装模块，安装模块的一般流程如下：首先开发者进入模块管理模块，之后开发者选择指定的模块，之后开发者点击安装模块，之后ApacheAssistant将指定模块安装至Apache HTTP Server。

* + 1. 卸载模块



图 26 卸载模块

开发者可以通过ApacheAssistant卸载模块，卸载模块的一般流程如下：首先开发者进入模块管理模块，之后开发者选择指定的模块，之后开发者点击卸载模块，之后ApacheAssistant将指定模块从Apache HTTP Server卸载。

1. 项目非功能需求
   1. 项目非功能需求概述

软件非功能性需求是指软件功能需求以外的需求，包括产品必须遵从的标准、规范和合约，性能要求，设计或实现的宏观约束条件及质量属性。

* 1. 运行环境需求
     1. 操作系统

Windows7版本及以上、Mac OS X 10.7.3 和更高版本、Oracle Linux 5.5 以上、Red Hat Enterprise Linux 5.5 和更高版本、Suse Linux Enterprise Server 10 SP2，11.x及以上、Ubuntu Linux 10.04 和更高版本

* + 1. 编译环境

python2.7版本及以上

* + 1. Apache版本要求

Apache2.0.0及以上

* 1. 性能需求
     1. 兼容性

该Web应用应能在当前主流平台上运行，包括Windows7、Windows8、Windows10及Linux内核操作系统。

* + 1. 可靠性

该应用保证本身运行的可靠性，作为一款辅助开发者对Apache平台进行操作的应用，应用本身的操作不应破坏Apache平台本身的正确性，具体表现即是不影响Apache本身的功能，并不会潜在的影响Apache平台本身的功能。

* + 1. 可用性

该应用应能够稳定运行，且运行中能提供给用户前述功能。

* + 1. 稳定性

系统在运行时能够始终稳定运行，不经常出现故障，且出现中小型故障时，能够发出故障警告并给出故障诊断素材供运维人员使用，出现大型故障时，能够保存上下文信息。

* + 1. 易维护性

应用部分模块或方法的更新和更换不应影响整个应用的正常运行。

* + 1. 可扩展性

作为一款开源软件，应用需要具备可扩展性，根据实际情况的不同（例如开发者的变化，用户需求的变化，物理设备的变化），对于应用本身的功能变动应该被支持。

1. 工作重点
   1. 工作内容

项目的后续计划是完成图形化配置工具ApacheAssistant的界面和配置管理、日志管理、性能监控、模块管理等具体功能。

配置管理模块需要实现对Apache HTTP Server的配置文件读取、解析、修改，对不同的配置项提供相应的修改选项，方便用户使用。

日志管理模块需要实现对Apache HTTP Server的日志文件读取、解析、修改，支持根据关键词、HTTP请求、IP地址、文件类型检索展示日志内容，支持备份、清除指定的日志内容。

性能监控模块需要实现查看Apache HTTP Server的工作状态，停止、重启Apache HTTP Server以及查看并发连接数、网络速度、CPU及内存占用等指标，并绘制成统计图表展示。

模块管理模块需要实现查看现有模块的模块名称、模块类型、模块文件位置、模块状态、模块状态、模块说明等信息，同时支持对动态模块的安装和卸载。

* 1. 技术路线

为了实现工作内容，本节拟定了如下的技术路线。

* + 1. 解析文件并提供接口

首先对Apache HTTP Server的配置文件、日志文件以及Apache HTTP Server的各项性能指标进行读取、解析、记录修改，根据各项功能的具体需求封装为可供界面调用的底层API接口。

* + 1. 界面调用接口实现具体功能

ApacheAssistant的图形管理界面通过调用底层的API接口来实现配置管理、日志管理、性能监控、模块管理等具体功能。