基于插件的卫星集成平台软件开发技术

范海涛[1] 常克武[2] 金煌煌[1] 张文静[3]
（1. 北京空间飞行器总体设计部，北京 100094；2. 中国第二代卫星导航系统专项管理办公室，北京 100054；
3. 北京计算机技术及应用研究所，北京 100854）

摘要：为了解决卫星研制过程中因需求的不断变化而导致软件开发和维护更改频繁的问题，提出了基于插件的卫星集成平台开发模式。设计了整个集成平台的总体架构，并针对其中的关键技术，如插件的接口设计、插件的自动识别与加载，插件消息通信机制等内容进行了详细的论述。以导航卫星为例，开发了基于插件的在轨维护核心功能平台，对提出的思路进行了验证。结果表明，该模式能够有效地降低维护和升级的工作量。

关键词：插件；集成平台；消息通信

The Software Development of Satellite Integration Platform based on Plug-in

Fan Haitao[1] Chang Kewu[2] Jin Huanghuang[1] Zhang Wenjing[3]
(1. Beijing Institute of Spacecraft System Engineering, Beijing 100094;
2. China Satellite Navigation Office, Beijing 100054;
3. Beijing Institute of Computer Technology and Applications, Beijing 100854)

Abstract: In order to solve the problems caused by changing demands of software development and maintenance of frequent changes in the process of satellite development, a satellite integration platform based on plug-in technology is proposed. The overall architecture of integration platform is designed. The key technologies, such as plug-in interface design, plug-in automatic identification and loading, the message communication mechanism of plug-in, are described in detail. At last, taking a navigation satellite as example, an integration platform of on-orbit maintenance core functionality based on plug-in technology is developed and the result shows that it can effectively reduce the maintenance and upgrade work.

Key words: plug-in; integration platform; message communication

1 引言

近年来，航天信息化技术得到了越来越广泛的应用。在软件开发过程中，当用户需求发生变化时，可能会对整个软件的设计思路产生重大影响。即使在原有系统的基础上新增加一个很小的功能时，都可能会导致整个系统架构发生改变而使得原系统崩溃，甚至需要重新设计软件架构，前期开发的软件经常刚刚上线运行即面临着过时下线的危险。这给现有的软件开发模式带来了极大的挑战[1]。

基于插件的集成平台开发模式是解决这一问题的有效解决方案。基于插件技术，就是在程序设计过程中将功能模块（插件程序）嵌入到主程序（宿主程序）中，插件与宿主程序之间通过相应的接口协议实现相互数据通信，在保证宿主程序不发生更改的情况下，通过开发新的插件来实现系统功能的更新[2, 3]。这种模式可以大大缩短软件开发时间，而且，即使用户的需求发生变更时，只需设计新的插件或者修改现有的插件即可满足用户的要求，避免了对整个系统进行重新设计。

本文以导航星座卫星运行管理为例，旨在通过搭建一个集成平台，将现有的 MATLAB 算法文件封装

作者简介：范海涛（1982–），博士，航空宇航制造工程专业；研究方向：工程数字化设计、计算机仿真。
基金项目：“十二五”国防基础科研项目（C0320110002）。
收稿日期：2014-05-20

成动态库插件的形式，集成到该平台中。一方面为设计师提供直观的卫星由于条件变化带来的工作状态变化，方便设计师了解卫星状态；另一方面为卫星平台常规操作提供自动化决策平台，为运控及测控操作提供建议。该平台具有以下几个特点：

a. 该平台能脱离 MATLAB 软件单独运行，能够有效节省计算机资源；

b. 利用 VC 的 MFC 进行设计，具有界面友好、操作方便、简单实用等特点；

c. 每个仿真模块均以插件的形式存在，各个模块之间相互独立，耦合度低、结构清晰、易修改；

d. 每个插件均预留相应的数据交换接口，方便不同模块间的集成，便于后续相关模块的扩展。

采用这种方法，可以大大降低各模块之间的耦合以及整个系统的复杂度，并且在主框架开发完成后，用户可以随时进行新的仿真模块的开发并将其加载到平台之中，就像其他内置模块一样使用和操作，实现仿真模块的“即插即用”。

2 总体架构

该集成平台采用的是“平台+插件”的思想。“平台+插件”的设计思想是将实现的功能以插件的形式通过平台统一地管理起来。通过建立完善的平台和插件之间以及不同插件之间的消息机制，将不同功能的插件有机地集成到一起，从而能够有效地进行协同工作[4, 5]。

插件的主要功能是在不修改主程序的情况下增加新的功能。在开发过程中，不会影响原有主程序及插件功能，是一种真正意义上的即插即用软件开发。该平台包括 2 个部分：主程序框架(main-app)和插件(plugin)。主程序是包含插件的程序，包含了标准的插件接口，用于与插件之间的通信。插件是遵循了某些指定规则的 DLL，每一个插件均表示了某一个指定功能。如图 1 所示。

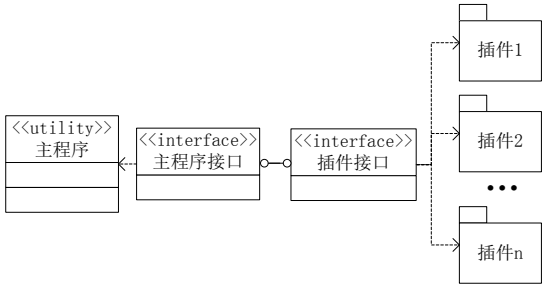

图 1 插件工作原理

宿主程序(main-app)：宿主程序是一个可执行程序，是用来加载插件的载体，负责整个系统的启动，并对插件进行统一管理。VC++提供了一个十分便利的 MFC 框架，提供了直观的界面框架和丰富的接口。本平台采用 MFC 的 MDI 框架作为主程序框架。

插件(plug-in)：插件定义了该平台所具有的功能，以动态库 DLL 的形式存在，即将每一个功能模块封装到 DLL 中。插件需要动态地加载到系统平台中，因此，插件需要给宿主程序提供调用的接口，以完成宿主程序与插件之间的通讯与交互。

通过构建一个通用的仿真系统平台，其个性化的功能以插件的形式进行加载，使系统功能不断完善和扩展，以满足导航星座的有效管理。系统体系总体架构如图 2 所示。

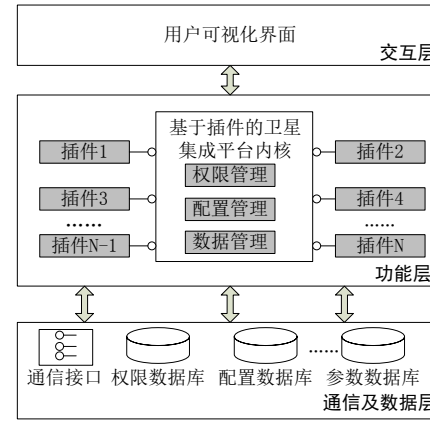

图 2 系统体系总体架构

交互层：可视化的算法界面，用户通过该界面实现与内部数据的通信及功能模块的计算。

功能层：这是本平台的核心层，包括权限管理、配置管理、数据管理等基础管理，并实现与各个插件之间的信息交互。

通信及数据层：这是本平台的基础层，主要包括通信接口的定义、权限数据库、配置数据库、参数数据库等，是系统正常运行的基础。

3 关键技术

3.1 平台的接口设计

插件与宿主程序之间的接口设计是整个平台开发过程中最重要的组成部分。基于插件的卫星集成平台包括两类接口：宿主程序接口和插件接口。只有这两类接口按照统一的协议进行设计，才能实现宿主程

序与插件之间的数据通信，这两类接口的设计如下：

```
typedef struct PlugInterfaceHost        //宿主程序接口，用于插件与宿主程序的通信
{
    CWnd * pParentWnd;          //该宿主程序的句柄
    UINT nIDResource;           //该宿主程序的框架资源ID
} PLUGINTERFACEHOST;
typedef struct PlugInterface            //插件接口, 用于宿主程序访问该插件
{
    HICON ico;                  //加载插件的图标
    Cstring strIconName;        //插件图标的名称
    Cstring strFileName;        //插件的名称
    Cstring strPlugVersion;     //插件的版本号
    int iToolNo;                //插件所属工具栏ID
    int iPlugNo;                //插件ID
  } PLUGINTERFACE;
```

PlugInterfaceHost 为宿主程序设计接口，向插件提供宿主程序的句柄、资源等信息，用于插件与宿主程序的通信，每一个插件均通过该接口访问宿主程序的资源。PlugInterface 为插件的设计接口，通过该接口，宿主程序可访问插件的相关信息，包括加载插件的图标、插件图标的名称、插件的名称、插件的版本号、插件所属的工具栏 ID 及插件 ID 等信息。在以后进行插件开发时，只需该插件继承 PlugInterface 接口即可。

3.2 插件的自动识别与加载

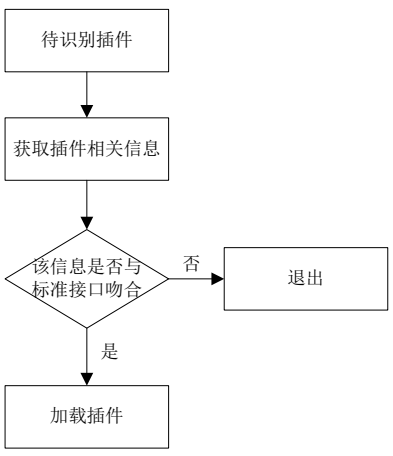

图 3 插件的识别与加载设计思路

在实际应用中，后续要开发的插件会提供何种功能以及需要的资源往往无法预先获知，因此该平台必须能够自动识别插件并进行加载，其识别的思路如图 3 所示。

在进行开发时，要进行接口的控制管理，通常而言，大多数新的接口是在已有的接口上增加新的功能，在后续的开发中，一方面，要保证新的接口能够兼容以前的版本；另一方面，要严格管理控制接口的版本，防治出现不必要的混乱并且后续的开发能方便地兼容旧的接口。

基于插件 MATLAB 仿真集成平台的插件加载流程如图 4 所示，其具体步骤如下：

Step1：启动主程序框架，实现主程序框架和插件的初始化，转到 Step2；

Step2：在指定文件目录 Plugin 下遍历 dll 插件，获取第 *i* 个插件 dll，转到 Step3；

Step3：判断该插件是否满足插件端口标准。是，加载其插件，转到 Step4；否，转到 Step4。

Step4：判断是否存在下一个插件：是，*i*=*i*+1，获取下一个插件，转到 Step2；否，转到 Step5。

Step5：退出。

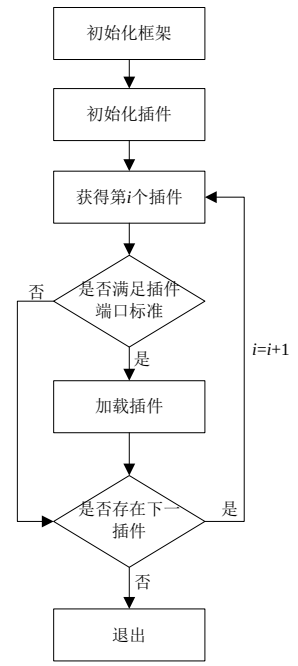

图 4 插件加载流程

3.3 插件消息通信

插件的通讯功能包括两个方面：一方面各个插件与遥测数据库之间的通讯，即各个插件模块中的参数输入，需要通过从遥测数据库中获得，以便能够根据

实际情况进行实时仿真；另一方面，是各个插件之间的数据通讯，即插件 *i* 的输出参数是插件 *j* 的输入，为了降低插件与平台、插件与插件之间的耦合性，实现插件的“即插即用”，本文提出采用一种通讯组件来实现，该组件提供统一的、简单易用的 API 接口，遥测数据库、各个插件之间通过统一的接口标准实现数据之间的通讯，其通讯机制如图 5 所示。

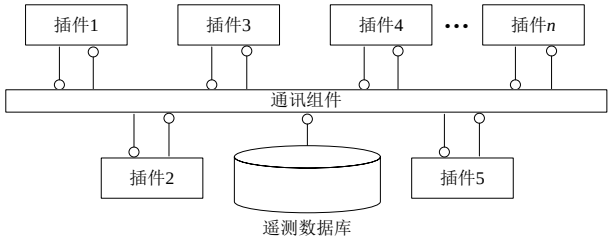

图 5　插件之间的通讯机制

作为一个通用的通讯组件而言，其自身不但要实现强大的通信功能，同时还要有一个简单易用的使用接口，在接口设计中，遵循简单易用的原则，接口设计列表如表 1 所示。

表 1　接口设计列表

序号	接口方法名	功能说明
1	Pl_LoadLibrary	初始化通讯组件
2	Pl_FreeLibrary	释放通讯组件
3	IrecvPlugID	接收插件 ID
4	IsendPlugID	发送插件 ID
5	IdataExange	交换数据包

交换数据包 IdataExange 由一系列参数列表组成，包括三个部分，可表示为 IdataExange={ParaName, ParaType, ParaValue}，其中 ParaName 表示参数名称，ParaType 表示参数的类型（包括 int, char, double, bool,…等数据类型），ParaValue 表示参数的值。

4　实例验证

针对导航星座卫星运行管理需求，开发了一个在轨维护核心功能平台。工具栏中每一个命令图标表示一个插件，用来实现某一个功能，该工具栏的命令通过搜索指定路径下的插件而动态加载。

从图 6 中可以看出，该平台集成了 8 个插件，即对应 8 个算法，这些插件之间相互独立。每个界面算法中的参数可以通过交互界面直接输入，也可以通过选择不同的卫星号而直接从遥测数据库中获取，设计人员可以根据不同的情景需求，选择不同的输入方式。图 6 中的“南极干扰算法”为直接从交互界面中输入而进行计算。

图 6　基于插件的在轨维护核心功能平台界面

5　结束语

随着越来越多功能模块的开发和使用，传统的软件开发模式已无法满足需求，而基于插件的集成平台软件开发模式较好地解决了这个问题。该模式可使多个功能模块的开发并行进行，且各个模块之间相互独立，能够有效减少平台维护和升级的工作量。通过定义统一的数据交换接口，便于不同功能模块之间集成和后续扩展，具有较好的适用性。

参考文献

1　申启杰，凌捷. 基于 C#的插件框架设计和实现[J]. 计算机应用于软件，2010，27(1)：148～149，164

2　李俊娥，周洞汝.“平台/插件”软件体系结构风格[J]. 小型微型计算机系统，2007，5(5)：876～881

3　吴良波. ISV 插件平台系统设计与实现[D]. 浙江大学硕士学位论文，2010

4　吴亮，杨凌云，尹艳斌. 基于插件技术的 GIS 应用框架的研究与实现[J]. 地球科学-中国地质大学学报，2006，31(5)：609～614

5　陈方明，陈奇. 基于插件思想的可重用软件设计与实现[J]. 计算机工程与设计，2005，26(1)：172～173