doi:10.3969/j.issn.1671-637X.2014.12.021

插件式无人机任务规划软件框架设计

肖 强, 朱玉祜, 杨丙泉
(中国航空工业集团公司洛阳电光设备研究所,河南 洛阳 471000)

摘 要:在复杂的无人机地面站系统软件开发过程中,如何实现高效的并行开发和应对不断变化的需求显得至关重要。首先介绍了插件框架的工作原理和实现途径,在此基础上提出了插件式无人机任务规划软件框架设计方案,详细描述了框架的设计思路,对插件管理器、人机界面管理和框架内部消息通信机制等一些关键技术进行了探讨。最后将这一理论成果运用到了实际项目中,实现了全插件式的任务规划软件框架,实践证明该框架可以显著提高软件开发效率,提升软件产品质量。

关键词:无人机;插件;软件框架;任务规划;插件通信机制
中图分类号:V279 **文献标志码**:A **文章编号**:1671-637X(2014)12-0094-04

Design of a Plug-in UAV Mission Planning Software Framework

XIAO Qiang, ZHU Yu-hu, YANG Bing-quan
(Luoyang Institute of Electro-Optical Equipment, AVIC, Luoyang 471000, China)

Abstract: In development of system software for UAV Ground Control Station (GCS), it is essential to realize parallel development effectively and deal with the constantly changing demand. The working principles and realization ways of plug-in framework are presented, and the design scheme of plug-in UAV mission planning software framework is given. The design idea of the application framework is described in detail, and some key technologies of the plug-in manager, user interface management and the internal message communication mechanism are discussed. The design scheme was applied to an actual project and realized all-plug-in mission planning software framework, which proved that the framework can significantly improve the efficiency of software development, and improve the quality of software product.

Key words: UAV; plug-in unit; software framework; mission planning; plug-in communication mechanism

0 引言

随着军用计算机技术的发展,用户对软件的专业性和扩展性提出了更高的要求。目前机载、舰载等软件大多采用面向过程的软件开发方法,其明显缺点是:1)在设计开始资料比较缺乏的情况下,缺少模块划分和接口约定,不能快速建立并行开发模式,开发速度过于缓慢;2)随着代码规模的增加,模块间的耦合关系已经形成,原先的设计理念和架构很难适应需求的变化,必须寻求一种新的软件体系结构和开发方法来解决上述问题。本文结合面向组件开发技术的特点引出了针对无人机任务规划应用的插件式软件架构,在解决当前软件危机、增强软件健壮性和扩展性、提高软件质量等方面提供了高效的解决方案,它能在不修改程序主体的情况下对软件功能进行修改和加强,真正实现软件的"即插即用"[1]。

收稿日期:2014-02-08 **修回日期**:2014-03-11
作者简介:肖 强(1984—),男,河南洛阳人,硕士,研究方向为火控系统仿真。

1 插件框架分析

1.1 插件框架的工作原理

插件技术就是在程序设计过程中把应用程序分成平台和插件两个部分。平台作为程序的主体,实现基础架构和业务核心,插件用来实现主要逻辑和扩展功能,插件管理器作为平台和插件之间的桥梁,负责插件的管理和维护。平台和插件之间通过统一接口和约定调用,在程序结构基本不变的情况下通过增减或修改插件来调整应用程序的功能。在插件框架下,插件可以方便地进行配置和整合,需求变更只会引起一个局

部性的修改，通过设计新的插件或者修改现有插件就可以满足要求，不需要对整个软件进行编译[2]。

1.2　插件框架的实现途径

综合插件框架的应用场景和运行平台，可以考虑采用以下两种插件框架。

1）基于 COM 组件的插件框架。COM 即组件对象模型，提供了组件之间的交互规范和环境，插件对于主程序来说是完全透明的，主程序不需要知道插件怎样实现预定的功能，它只需要通过接口访问插件，主程序与各插件之间的信息交流变得非常简单[3]。这种方式虽满足设计需求，但缺点在于设计过程中涉及到大量的 COM 技术原理，增加了开发的门槛和过程中的不确定性。

2）基于动态链接库（DLL）的插件框架。DLL 是一种具有一定功能的可执行软件模块，虽然它本身不能独立运行，但是可以通过其他能独立的程序调用它的内部功能[4]。此类插件可以通过接口契约获得插件 DLL 中的函数签名，然后由主程序在合适的地方调用它们，同时对插件的合法性做出判断。

综合考虑适用性和灵活性，选择基于 DLL 的插件框架作为任务规划软件的应用框架。通过重新组织架构，定义 DLL 开发的统一接口和调用原则，使得开发出的插件符合组件式开发特点，满足实际工程需要。

2　任务规划软件框架设计

2.1　软件框架模型

根据以上对插件框架的分析，综合任务规划软件的需求和平台插件结构的设计要点，得出了任务规划软件的框架模型，如图 1 所示。

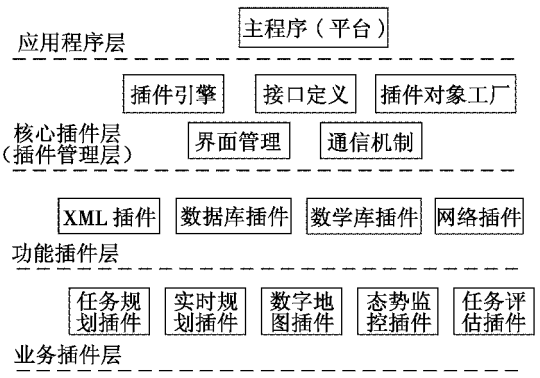

图 1　任务规划软件应用框架模型示意图

Fig. 1　Schematic of mission planning software application framework

该框架模型中，主程序只负责构建软件基本运行环境和激活核心插件——插件管理器，它作为框架的核心，负责插件的集中管理，包括动态加载、对象生命周期管理、动态生成界面、插件间通信等重要功能[5]。

功能插件层是最能够体现插件思想的，通过将开发用到的基本功能，如数据库、网络通信等插件化，不仅方便了业务层调用，而且将那些分散的、不易重用的、容易在调试中出现问题的部分以标准接口的形式进行提取，业务插件层可以利用上层插件提供的接口来展开任务规划软件的各项功能详细设计。

2.2　插件管理器的设计

插件管理器属于核心层插件，作为平台与插件交互的媒介，插件管理器是其他插件运行的基础。插件管理器按其功能主要包括以下几个重要类和对象。

1）基础插件接口类 IObject。IObject 类是插件开发的重要基类，任何需要被直接识别的插件必须继承该类。IObject 类的声明如下所述。

```
class IObject{
virtual long retainObject( ) const = 0;
virtual long releaseObject( ) const = 0;
virtual const char * getClassID( ) const = 0;
virtual const char * getClassName( ) const = 0;
virtual bool queryObject(long iid,IObject * * p) const = 0;};
```

IObeject 类似于 COM 组件中的 IUnknown 类，继承自 IObject 类的接口具有引用计数，获取属性，查询接口的能力。接口对象以不同的派生形式为平台或其他插件提供服务，这样就使得用户只需关心接口定义，而不必去关注接口是如何实现的，从而实现接口和实现的分离[6]。

2）插件类型库 plugins_info。插件类型库是为了记录插件类型信息而定义的一个 hash_map 对象，该对象定义如下所述。

```
typedef bool ( * Creator)(const char * ,long,IObject * * );
hash_map < std::string,Creator > plugins_info;
```

主程序加载某个动态链接库插件时会自动向插件类型库进行注册，plugins_info 对象完成一次赋值操作。注册内容包括插件的标识和插件构造函数。平台每次启动时首先会通过类型库管理器来进行加载，平台退出时会自动卸载所有已注册的插件。

3）插件对象工厂类 PluginFactory。插件类型库记录了所有已注册的插件类型信息，当平台或其他插件需要调用注册插件的接口函数时，就需要插件对象工厂类根据插件的标识创建出接口对象实例。插件对象工厂类是平台或插件运行过程中获取其他插件接口对象的唯一途径。插件对象工厂类 PluginFactory 的声明如下所述。

```
template < class Interface > class PluginFactory{
public:
    PluginFactory (const char * clsid):_p(0){
        createObject(clsid,Interface::getIID( ),address( ));}
private:
    Interface * _p;
    …};
```

模板类的使用出于实例化不同接口时的考虑，要

使用插件提供的接口函数时,只需如下的方式:

```
PluginFactory < your_interface >  pInstance( your_interface_id);
```

就可以方便地获取到接口对象的智能指针,关于查询接口是在哪个插件中实现,创建接口对象时的内存申请和释放等底层工作都由插件对象工厂类在后台完成了[7]。

2.3 人机界面管理

任务规划插件、数字地图插件、任务评估插件等很多需要人机交互的功能插件都以人机界面的形式在软件中呈现,因此人机界面管理在整个框架中处于非常重要的地位[8]。

人机界面接收用户输入,对系统消息进行处理并显示输出结果,作为一个完整的界面线程存在于进程地址空间中。它要实现插件化,除了需要继承基本插件接口类 IObject,还需要增加新的接口方法。其中 GetWindow()函数获得窗口句柄,CreateWnd()函数负责创建窗口,DestroyWnd()函数负责销毁窗口,Refresh()函数来刷新渲染窗口,OnCommand()函数和 OnUpdateUI()函数用来对窗口的消息进行处理。这样就把界面的通用属性提取了出来。

在窗口实例化的时候,为了兼容不同的窗口类型,提出了窗口适配器的概念。窗口适配器即创建一个新的窗口作为功能窗口的父窗口,不同类型的窗口只要具有 HWND 数据成员,都可以作为子窗口嵌入到父窗口中,系统消息由父窗口利用反射机制路由给子窗口来处理,这样不仅考虑到了消息处理效率,而且提供给用户较好的交互感受。

2.4 插件通信机制

任务规划应用中,需要各插件之间灵活互动,比如移动鼠标时,数字地图插件需要将当前的地理位置信息发送给任务规划插件,因此必须建立一套有效机制,既保持插件相互独立,又实现插件间的正常通信。

应用框架插件间的通信机制采用观察者设计模式[9],由插件管理器集中处理插件间各类事件的触发与响应,其工作机制如图 2 所示。

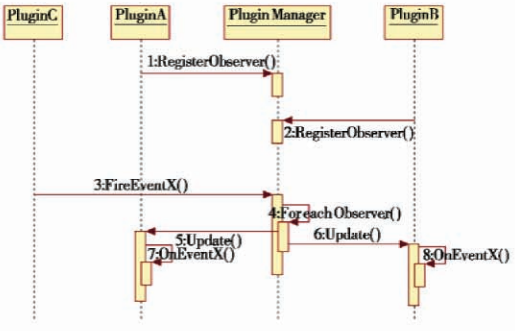

图 2 插件通信机制序列图

Fig. 2 Sequence diagram of communication mechanisms

首先需要对事件 EventX 的事件类型进行预定义。事件类型包括事件名称和形参类型。事件名称是为了让插件管理器能够动态对事件进行识别,形参是使得事件发生时能够进行参数传递[10]。PluginA 和 PluginB 向插件管理器注册事件 EventX 的响应函数,为了能够正确地响应事件和传递参数,注册函数的返回值和形参类型必须与前面事件类型的定义一致。事件 EventX 由 PluginC 触发,使用辅助函数 FireEventX()来触发事件。插件管理器发现事件 EventX 满足触发条件就会调用 Update()依次通知已经向它注册过的 PluginA 和 PluginB,然后按照注册顺序进入插件内部去调用各自的消息响应函数 OnEventX()。观察者设计模式使得插件之间实现松耦合,通信变得简单而高效。

3 应用示例

插件式任务规划软件框架在实际项目中得到了应用,图 3 为某任务规划软件运行界面,图 4 为软件的工作序列示意图。

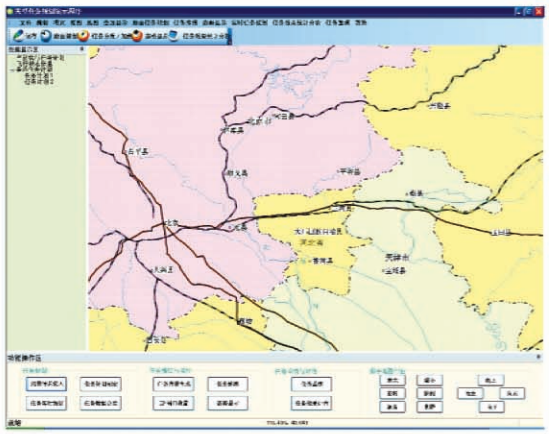

图 3 任务规划软件界面

Fig. 3 Mission planning software UI

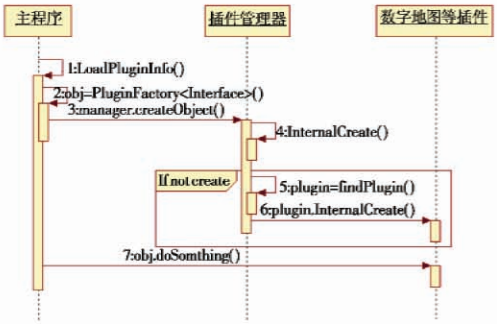

图 4 任务规划软件序列示意图

Fig. 4 Schematic sequence of mission planning software

该软件在 Microsoft Visual Studio 2005 环境下使用标准 C ++ 开发。软件中所有功能模块都以插件形式实现,软件启动后读取配置文件中的插件目录,然后通过插件管理器完成插件的注册和加载。实例化插件对象时,插件管理器首先试图通过内建函数创建,如果创

建不成功说明该对象没有在插件管理器中实现，需要根据插件 ID 从已注册的插件类型库中查找对应的插件，由该插件完成具体的对象创建工作，此后插件提供的接口方法可被主程序和其他插件识别并调用。以此类推，所有的业务插件在主程序的合理调度下“有条不紊”地完成任务规划的各项功能。

结果表明，采用插件框架的软件系统具有代码简洁、执行稳定的特点，其强大的扩展特性为软件系统的后续升级和日常维护提供了有力的技术支撑。插件架构下的开发模式不仅解放了软件工程人员，也使用户的使用感受得到很大提升。

4　结论

插件式开发降低了系统的耦合性，规范了软件开发过程，提高了开发的效率、并行性和灵活性，大大缩短了开发周期[11]。利用该框架开发应用最大的好处在于提供了定义规范、面向接口的功能插件集，在此基础上能够实现强大的业务逻辑和定制功能。框架设计时充分考虑了框架的通用性，其中包括了基础接口的定义准则，人机界面管理的适配器模式和完整的插件内部通信机制，使得该框架为实现无人机地面站软件的通用化起到了积极的推动作用。

参 考 文 献

[1] 向慧，赵恒，唐素芬，等. 基于插件技术的舰载指控系统应用框架[J]. 火力与指挥控制，2010，35(7)：120-122.
XIANG H, ZHAO H, TANG S F, et al. Research on application framework of warship command system based on plug-in[J]. Fire Control and Command Control, 2010, 35(7): 120-122.

[2] 姜川. 计算机软件中的插件技术及应用研究[J]. 数字技术与应用，2013(1)：94.
JIANG C. The plug-in technology and application in computer software [J]. Digital Technology & Application, 2013(1): 94.

[3] 潘爱民. COM 原理与应用[M]. 北京：清华大学出版社，2000.
PAN A M. The principle and application of COM[M]. Beijing: Tsinghua University Press, 2000.

[4] 侯锐，田泽. 基于 QT 构建算法类库插件应用框架的研究与实现[J]. 信息与电脑，2009，11：119-120.
HOU R, TIAN Z. Algorithm based on QT library plug-in application framework to build the research and realization [J]. China Computer & Communication, 2009, 11: 119-120.

[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.
CHEN F M, CHEN Q. Design and implementation of reusable software based on thought of plug-in [J]. Computer Engineering & Design, 2005, 26(1): 172-173.

[6] 梅珊，赵雯，朱一凡，等. 插件式可重配导弹总体设计集成优化框架研究[J]. 系统仿真学报，2006，18(12)：3564-3568.
MEI S, ZHAO W, ZHU Y F, et al. Research of plug-in based configurable missile design, integration and optimization framework[J]. Journal of System Simulation, 2006, 18(12): 3564-3568.

[7] 陈志飚，王娜，张净宙，等. 基于. NET 平台的插件式船舶工程计算系统应用框架开发[J]. 计算机应用，2010，30(s1)：225-229.
CHEN Z B, WANG N, ZHANG J Z, et al. Development of plug-in ship assess application framework based on. NET [J]. Journal of Computer Applications, 2010, 30(s1): 225-229.

[8] 周焱. 无人机地面站发展综述[J]. 航空电子技术，2010，41(1)：1-6.
ZHOU Y. A review of UAV GCS development[J]. Avionics Technology, 2010, 41(1): 1-6.

[9] GAMMA E, HELM R, JOHNSON R, 等. 设计模式[M]. 李英军，马晓星，蔡敏，等译. 北京：机械工业出版社，2000.
GAMMA E, HELM R, JOHNSON R, et al. Design pattern [M]. Translated by LI Y J, MA X X, CAI M, et al. Beijing: China Machine Press, 2000.

[10] 张毅，李国卿，赵军喜，等. 插件式 GIS 应用框架关键技术研究[J]. 测绘科学技术学报，2010，27(4)：298-301.
ZHANG Y, LI G Q, ZHAO J X, et al. Key technology of plugin-based GIS application framework [J]. Journal of Science and Technology, 2010, 27(4): 298-301.

[11] 王宏强，张航峰，李继进，等. 战术指挥情报终端的插件式软件框架设计[J]. 指挥控制与仿真，2012，34(6)：115-117.
WANG H Q, ZHANG H F, LI J J, et al. Design of plug-in based software framwork for tactial command & intelligence terminal [J]. Command Conrtol & Simulation, 2012, 34(6): 115-117.