

一致好评,广大医护人员纷纷对系统的内容和操作方式等给药剂科提出了许多宝贵建议。根据这些建议,药剂科及时对部分界面和内容进行了调整,使其更贴近临床,更具有可操作性。药品信息查询管理系统的开发与应用,对提高我院的药事管理水平具有积极意义。

5 结语

在现阶段,医院所使用的 HIS 系统大多只停留在以经济管理为主线、面向医院内部资金流和物流管理的阶

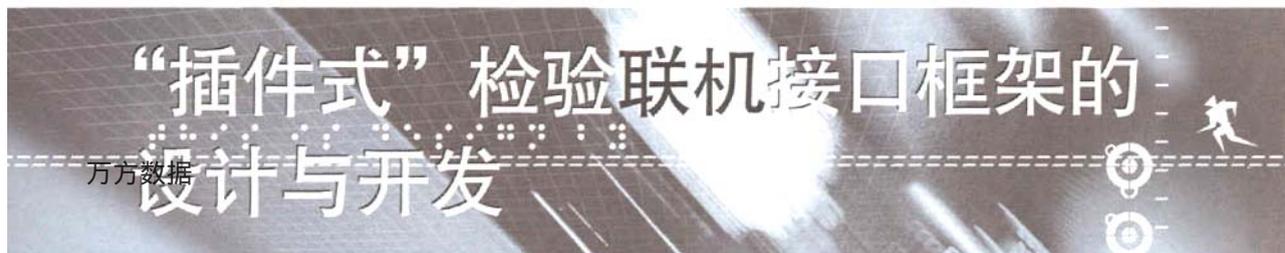
段^[1],仅仅体现了对物的管理,未能满足现代医院药学发展和医疗质量不断提高的要求。而我院自主开发的药品信息查询管理系统,在医院 HIS 系统不断发展与完善的基础上,配合用药安全防火墙的推广与使用,二者相互补充,以网络形式向医院广大医务人员提供药学服务保障平台,在线开展信息交流、科研教学、处方分析、新药审评、不良反应、科室管理、合理用药等全方位药学服务,增加了医务人员间的交流和互动,功能覆盖了医院药

事管理工作的各个方面,改变了医院药事管理头绪繁多、疲于应付的现状,推动了医院药事管理事业的发展,提高了药品治疗质量和合理用药水平,具有良好的社会效益和推广价值。CDM

参考文献

- [1] 李丹.浅谈医药行业信息化的发展趋势[J].山西高等学校社会科学学报,2006,18(4):48-49.

【收稿日期:2007年9月28日】



严睿^① 符波^① 陈军龙^①

关键词 检验信息系统 框架开发 插件

摘要 临床检验信息系统(LIS)在医院信息系统中占有较大的比重,然而由于各种检验仪器的输出协议各不相同,很难有“包罗万器”的LIS系统能够采集所有的仪器数据。开发仪器接口框架,利用“插件”机制建立可扩展的解决方案,使之实现“包罗万器”的应用目的。

Design and Development of Interface Framework of "Plug-in" Laboratory Online System / YAN Rui, FU Bo, CHEN Jun-long // China Digital Medicine. -2008 3 (4): 42 to 43

Keywords laboratory information system, framework development, plug-in

Abstract Laboratory Information System (LIS) is a main part of hospital information system, but the difference of output protocol in laboratory instruments leads the general LIS system can't collect all the instrument data. The instrument interface framework, relying on the "plug-in" mechanism, can form an extensive solution and make the LIS become compatible for all the laboratory instruments.

Corresponding author The 478th Hospital of PLA, Kunming 650200, Yunnan Province, P.R.C.

检验工作是临床业务中不可或缺的一部分。临床检验信息系统(LIS)在医院信息系统中占据比重大,然而由于各种检验仪器的输出协议各不相同,市场上很难有“包罗万器”的LIS系统能够采集所有的仪器数据。开发一个能够适合各类检验

① 解放军第四七八医院, 650200, 云南省昆明市关上双桥路299号

仪器接口的框架,使之能够“包罗万象”,其软件体系结构见图1。

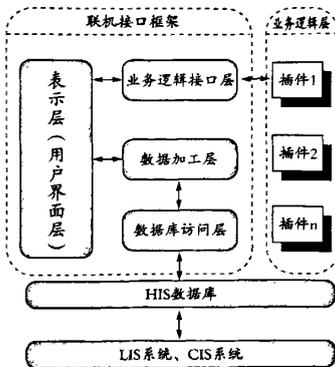


图1 LIS系统软件体系结构图

该体系结构设计思想是:仪器接口框架主要完成对仪器原始数据的自动采集,并把采集到的原始数据交给插件(业务逻辑层),插件则根据仪器的接口说明书,提炼出各检验项目的正确结果,传回接口框架,由框架把结果保存至HIS数据库。此设计思想的核心是:利用插件(plug-ins)机制建立可扩展的解决方案,把业务逻辑接口展示给用户(各医院的信息专家),由用户来完成业务逻辑层的编写(把框架传过来的字符串,按一定的规则截取相应位置上的子串而已),并把业务逻辑层编译为插件,插入框架中后,一个完整的接口应用程序就可付诸实用了。

按照这种构想,框架是不用关心插件是如何完成对数据的解析的,插件也不用管框架是如何完成对串口的读写,不用管解析后的项目结果,框架是如何处理的。二者只要严格按照业务逻辑接口层的协定进行通信,就可实现设计的目的,二者的通信采用C#和.NET作为开发平台。

1 建立接口(定义在业务逻辑接口层)

在C#程序中,接口是用来定义一

个类的功能的。接口定义了预期的方法、属性、事件信息,为了使用接口,每个实现接口的类必须严格按照接口的定义完成所描述的功能。笔者把这个接口定义为: IPlug。这个接口定义了以下成员: void ParsingData(byte [] data),该方法用于解析框架输入的原始数据;event DataReadiedEvent-Handler DataReadied,每解析好一个项目结果后触发此事件,通知框架显示。event DeleteOldDataEvent-Handler DeleteOldData,每收到新样本数据时触发此事件,通知框架删除数据库中的老数据;event EventHandler InsertToDB,每样本的数据全部解析完成后触发此事件,通知框架把数据保存至数据库。

2 定制属性(定义在业务逻辑接口层)

自定义属性(attribute)的作用是用来描述插件的信息,它可以定义自描述的元数据,比如说插件名称,插件用途等。这定义了两个定制的属性: PlugDisplayNameAttribute和PlugDescriptionAttribute,所有的插件内部的类必须支持这两个属性。在程序运行的时候,主程序将可以利用反射(reflection)来取得属性值。

3 编写插件(Plug-Ins,定义在业务逻辑层)

插件的编写由用户来完成,但必须遵循以下约束:一是必须实现IPlug接口。由于框架根本不会知道插件内部的类是如何定义的,这非常重要,框架需要使用IPlug接口和各个插件通信。二是必须被上述两个属性标识,这

样使用者才知道这个插件的来源与用途。

4 加载插件

在一个插件定义好之后,下一步要做的就是框架是如何加载插件。为了实现这个目标,框架使用了反射机制。反射是.NET中用于运行时查看类型信息的。在反射机制的帮助下,类型信息将被加载和查看。这样就可以通过检查这个类型以判断插件是否有效,如果类型通过了检查,那么插件就可以被实例化,接口的成员就可以被框架调用。

至此整个设计构想完成。接下来需要用户自己完成插件的编写工作,才能部署使用。如果我们把接口框架比作一个有牢固地基和框架的建筑,而各个房间就是没有装修的“插件”,房产商交付房子后,用户应该自己装修,以满足不同的功能需求。框架完成后的运行界面见图2。CDM



图2 运行界面

参考文献

- [1] Jacquie Barker and Grant Palmer. Beginning C# Objects.
- [2] (美)里伯提(Liberty J.). Programming C# 中文版. 刘基诚,李愈胜,刘卫卫译.
- [3] (美)罗宾逊, (美)内格尔. C# 高级编程. 李敏波译. 3版.

【收稿日期: 2007年10月13日】