ISSN 1009-3044
Computer Knowledge and Technology 电脑知识与技术
Vol.7, No.10, April 2011,pp.2281-2284,2300

E-mail: xsjl@cccc.net.cn
http://www.dnzs.net.cn
Tel:+86-551-5690963 5690964

软件构架中接口设计方法

巩文化[1],张静[2],唐世庆[1],党朝发[1],陈波[1]

(1.装甲兵工程学院,北京 100072;2.酒泉卫星发射中心 东风航天城,甘肃 兰州 732750)

摘要:该文强调了软件构架中接口设计的必要性,介绍了5个方面的设计原则;给出了接口描述规范,介绍了5个指导方针;描绘了接口设计的标准结构(包含9个部分);罗列了接口可能的涉众;阐述了接口的表示方法;用具体的实例展示了这些方法在实践中的应用。

关键词:软件构架;接口设计;接口描述规范

中图分类号:TP311 文献标识码:A 文章编号:1009-3044(2011)10-2281-03

Interface Design Method in Software Architecture

GONG Wen-hua[1], ZHANG Jing[2], TANG Shi-qing[1], DANG Chao-fa[1], CHEN Bo[1]

(1.Academy of Armored Force Engineering, Beijing 100072, China; 2.Jiuquan Satellite Launch Center, Dongfeng Spacecity, Lanzhou 732750, China)

Abstract: This paper emphasizes interface design in software architecture, describes five aspects of the design principles, gives interface specification description, introduced 5 guidelines, describes the standard structure of interface design (including 9 parts), lists possible stakeholders of the interface, describes representation of the interface, using specific examples show the application of these methods in practice.

Key words: software architecture; interface design; description specification

1 概述

在软件构架研究的早期,人们对系统元素及其相互作用给予了殷切的关注,但总是忽略元素的接口,好像接口并非构架的组成部分。然而,接口完全属于构架范畴,没有接口,人们就无法进行系统分析或系统构建。因此,软件构架视图编档中一项关键工作是为视图中所展示的元素接口编档。元素与其环境的交互可以采用多种形式,大多数交互涉及控制和(或)数据传送,这就需要构架师设计接口并规范地描述接口。

设计接口一般遵循以下一些原则:

1)所有元素都拥有接口

所有元素都会与其环境进行交互。

2)元素接口包含视图特有的信息

同一个元素可能出现在多幅视图中,因此在软件构架的设计视图中必须利用视图特有的术语(符号)为元素的接口编档。例如,在使用视图中,模块的接口可能会包含接口所提供的方法;但在工作任务视图中,该模块的接口就不会包含其所提供的方法。因为不同的视图要透露接口的信息的详细程度不同。

3)接口是双向的

在考虑接口时,大多数软件工程师首先会想到元素提供什么,但是,元素与其环境交互时,也需要使用资源或对其环境的运行方式有断言。因此,接口不仅包含元素"提供"什么,还包括元素"需要"什么。

4)一个元素可以拥有多个接口

每个接口都拥有一套逻辑相关的独立资源,服务于特定类别的元素。但有时候元素的用户可能仅需要元素所提供功能的一个子集。如果元素拥有多重接口,有可能其中一个接口就完全满足开发人员的需求,那么,开发人员仅需了解与自己相关的接口,而无需了解元素提供的全部资源。另外,多重接口还支持不同权限的访问和系统的不断演进。

5)元素通过一个接口可以与多个参与者交互

通过一个接口与元素交互的参与者的数量应该编档。例如,Web服务器通常会限制同时建立的HTTP连接的数量。

2 接口设计规范

为了便于其它实体与一个元素交互,构架师认为需要公开的该元素的相关信息就是接口信息。构架师要做出决策,决定应该公开哪些信息,这就是接口设计。将接口设计记录在文档中就是接口描述或接口说明。

编档接口就是要编写接口说明。虽然元素的接口包含了元素与其环境间的所有交互,但构架师决定透露的接口信息(即在接口说明中编档的信息)较为有限。编档交互的所有信息既不实际也无必要。相反,构架师只应该公开那些元素的使用者必需了解的信息。设计接口需要在透露信息多与少之间进行权衡。透露的信息过少会妨碍开发人员有效地使用元素。透露的信息过多则会使得

收稿日期:2011-02-21

作者简介:巩文化(1965-),男,安徽宿州人,装甲兵工程学院副教授,主要研究方向为C3I软件开发、测试,风险管理。

将来更改系统困难。不同的人需要了解元素的不同信息(参见第 4 部分),构架师可能需要在相关接口文档中描述接口不同的信息,以适应元素不同涉众的需要。

下面几条原则是业界通用的接口设计指导方针:

1)要关注元素如何与其环境进行相互作用,而不是关注元素如何实现。要使文档的内容是外部可见的特性。

2)仅可公开元素参与者需了解的信息。将一条信息写进文档就是隐式地向元素的涉众保证这一信息的可靠性和稳定性了。信息一旦公开,其它元素就可能会依赖这一信息,因此,更改信息也将产生更为广泛的影响。

3)如果不想让别人依赖某条信息,就不要将其写进接口文档。应该明白只有在参与者面临困境和整个系统面临危险时才能使用接口文档中不存在的,而从接口"泄漏"的信息。

4)要牢记谁会使用接口文档,他们又需要什么类型的信息。避免编档中存在不必要的信息。

5)描述信息尽可能具体和精确。如果两个参与者对一个接口说明有不同的解释,则该接口说明就可能会导致混乱。

元素接口规范

第 1 部分	接口身份
第 2 部分	接口资源
	a. 资源语法
	b. 资源语义
	c. 资源使用限制
第 3 部分	本地定义的数据类型
第 4 部分	错误处理
第 5 部分	接口可变性
第 6 部分	质量属性特征
第 7 部分	元素的需要
第 8 部分	基本原理和设计问题
第 9 部分	使用指南

图 1　接口文档包含 9 部分

3 接口文档的标准结构

构架师应该选择能够正确展现元素接口外部可见交互的结构。一般地,可以参考并剪裁下面的标准结构。如图 1 所示。

1)接口标识:当一个元素拥有多个接口时,应该分别对这些接口进行标识,以将它们彼此区别开来。最常见的标识方法是为接口命名。有时,仅仅命名接口还不能满足需要,还必须标识接口的版本。

2)所提供的资源:接口文档的核心内容是说明元素向其参与者所提供的资源。通过说明使用资源的语法、使用资源的语义(即使用他们时会发生什么)和使用资源的其它限制来定义资源。比如,下面是使用资源的语法示例:

afx_msg void OnSysCommand(UINT nID, LPARAM lParam);

3)局部数据类型:如果接口资源使用的数据类型不是编程语言提供的数据类型,构架师就需要定义该数据类型。使用这种资源的程序员必须知道:

① 如何声明自定义数据类型的变量和常量;

② 如何写这种数据类型中的文字值;

③ 这种数据类型的数据可能执行什么操作或比较;

④ 如何将这种数据类型的值准确地转换成其它数据类型的值。

4)错误处理:应该说明资源接口引发的错误。多个资源可能会引发相同的错误,为方便起见,每种资源都要列出自己引发的错误,并在字典内分别定义它们。也可以在字典内定义常见的错误处理方法。

5)接口提供的可变性:接口是否允许以某种方式对元素进行配置?要编档"配置参数"是什么,其如何影响接口交互的语义。这种可变性包含易修改性,如可见数据结构。应该为每个配置参数起一个名称并提供一个值域,并说明其实际值的有效时间。

6)接口质量属性:构架师必须编档向使用者公开的元素接口的质量属性(如性能和可靠性)。这一信息可以对接口的实现限制。

7)元素的需求:元素的需求可以是其它元素提供的指定资源。

8)基本原理和设计问题:像构架的基本原理或构架视图的基本原理一样,构架师还应该说明设计某个元素接口的理由。应该在基本原理中说明设计的动机、限制和折衷情况、被选择的设计、被拒绝的设计、选择和拒绝的原因和构架师对未来如何更改接口的所有见解。

9)使用指南:图 1 中的第 2b 部分和第 7 部分对元素的每种资源的语义信息进行了编档。有时候,这样做并不能满足需求。在某些情况下,必须根据大量交互操作相互联系的方式推测语义。实质上,同时考虑多个交互又涉及对交互"协议"的编档。这些协议能够说明元素设计者期望得到反复使用的、交互的所有行为和使用模式。

4 接口文档的涉众

不同的涉众关注接口的信息不同。设计者描述接口时须选择性地描述接口。

1)元素构建人员。元素构建人员需要了解其它涉众将要了解并可能会依赖的所有接口断言,以便实现这些断言。

2)元素测试人员。元素测试人员需要接口所提供的所有资源和功能的详细情况,这些资源和功能通常是测试的对象。

3)使用元素的开发人员。这类开发人员需要元素所提供资源的详细情况,包括语义信息。

4)分析人员。分析人员所需求的信息取决于所执行的分析类型。例如,对性能分析员来说,接口文档应该提供性能模型所需要的信息,如资源所需要的计算时间。分析人员是接口文档中的所有质量属性信息的最初使用者。

5)系统构建人员。系统构建人员在构建系统的同时,关注如何为元素接口中每项"需求"找到"提供"。通常,这种关注更多的是关注满足需求的语法,而不是满足需求的语义。

6)集成人员。集成人员用系统的构成元素组装系统,但对组装体的行为更感兴趣。因此,集成人员关注的更有可能是元素接口之间"需求"和"提供"的语义匹配,而不是语法匹配。

7)寻找在新系统中可重用品的构架师。构架师通常会先检查以前系统中的元素接口,也可能会考察商业市场,以期发现做预期任务的商用已上架元素而购买之。

8)管理人员。管理人员可能需要接口大小信息和接口所包含功能的信息,但无需过多细节。

5 接口文档的表示法

5.1 展示接口的存在

可以用展现构架的图形符号来展现接口的存在。图 2 展示了利用非正式表示法的一个示例。

如图 2 所示。接口的图形表示法通常在元素图标的边界展示一个符号。连接接口符号的线条表示在相连的元素之间存在接口。这样的图形表示法只能展现接口的存在,不能展现接口的定义。(a)表示一个含多个接口的元素。对于只有一个接口的元素,接口符号通常被省略。(b)展现一个接口有多个参与者的情况。客户 1 和客户 2 通过同一接口与主服务器交互。

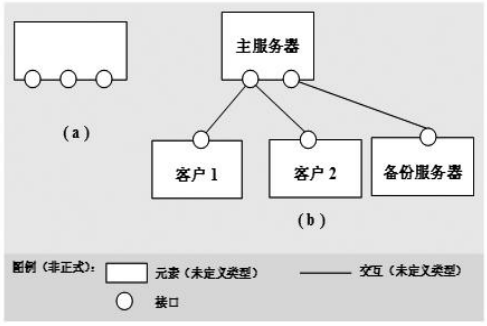

图 2 接口的图形表示法

图 3 说明了以 UML 展示接口的方式。图 3 尽管展示了接口的存在,但几乎没有揭示一点接口的定义:接口提供什么资源、接口需要什么资源,或接口交互的特性是什么。必须在相关支持文档中提供接口的定义。

如图 3 所示。用 UML 展现接口的语法信息。UML 用“棒棒糖”符号表示接口,此符号可以附着在类或子系统上。UML 还允许用一个类符号、一个方框被定格成一个版型来表示接口。带三角的虚线箭头表示一个元素实现一个接口。类符号的底部可以放置接口的说明信息:方法名、参数及参数类型等等。“棒棒糖”符号正常情况下用于表示元素对接口的依赖性;方框符号允许有更详细的接口描述,如接口所提供的操作。

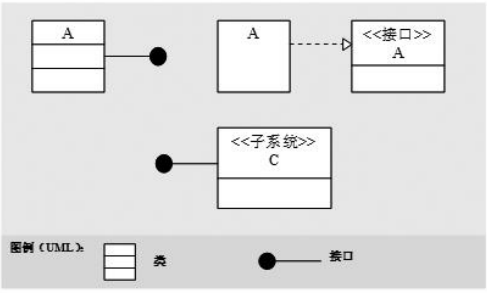

图 3 用 UML 展现接口的信息

5.2 语法信息的传达

对象管理组织(OMG)的接口定义语言(IDL)应用于 CORBA 领域,来说明接口的语法信息。IDL 提供了描述数据类型、操作、属性和异常的语言概念原子。但是,语义信息的唯一语言支持是注释机制。大多数编程语言都拥有说明元素特征的内置方法。C 语言的头文件(.h)和 Ada 包说明就是两个例子。例如:

Void getSystemTime(&systime);

5.3 语义信息的传达

自然语言是表达语义信息的最为普遍的表示法。布尔代数通常用来记录前置条件和后置条件,它们为语义的表达提供了相对简单和有效的方法。通过记录下元素响应特定使用的活动和交互顺序, 足迹也能用来表达语义信息。语义信息通常包含一个元素的行为或其一个或多个资源。在此情况下,任何描述行为的表示法都能发挥作用。

```
interface Account
{
      Readonly attribute string owner;
      Readonly attribute float balance;
      Void deposit(float amount);
      Void withdraw(float amount);
};
interface CheckingAccount:Account
{
      Readonly attribute float overdraft_limit;
      Void order_new_checks();
};
interface SavingAccount:Account
{
      Float annual_interest();
};
interface Bank
{
      CheckingAccount open_checking(string name,float starting_balance);
      SavingsAccount open_savings(string name,float starting_banace);
};
```

图 4 一个银行应用系统中元素的 IDL 范例

6 接口文档范例

以下是 2 个接口文档范例。每个范例,均指出了每个范例所展示的内容和未展示的内容。

6.1 IDL

图 4 展示的是用对象管理组织(OMG)接口定义语言(IDL)说明的一个小型接口范例。该接口是管理银行账户的一个元素的接口。

如图 4 所示。一个银行应用系统中元素的 IDL 范例(Bass, Clements, and Kazman 1998, p. 177)。元素提供管理金融账户的资源。账户的属性有:余额和账户名。所提供的操作有存款和取款。

尽管在这一类型的文档中,语法说明并不模糊,但还是缺少大量的语义信息。例如,用户是否能随意取款?取款数额上限是否是账户当前余额?每日取款是否有最大限额?取款后是否必须保留最低余额?如果所有这些限制都是真实的,那么,当该限制被违反时,会出现什么情况呢?是取走最高允许金额,还是取消整个交易?IDL 本身并不足以充分编档接口,原因主要是,IDL 没有提供讨论接口语义的语言概念原子;没有语义,就会产生大量的歧义和误解。

6.2 某项目接口表示法

表 1 所示的是装工院目前正在研制的某项目中的一部分接口描述。从表中可以看出,接口描述的信息既包括语法信息、语义信息,又包括使用接口的限制。这一种接口描述是面向编程人员的。当然,项目中还有面向其他涉众的接口描述,不在此赘述。 (下转第 2300 页)

表 1 时统接口函数使用简表

序号	各项描述	
1	函数名	TS_SendContinueCommandPack
	功能简述	发送作战时继续命令。在当前作战时基础上,作战时继续运行。
	参数类型	DWORD DownIP 目的 IP 地址
	返回值	int 型,通过 TS_RecvBackPackage()获取,如果命令发送成功,等于 5,表示发送的字节数。否则返回一个 SOCKET_ERROR 错误。
	所属动态库	UdpProcess.dll
2	函数名	TS_SendNextZTKZPackage
	功能简述	发送下一阶段状态控制命令。用于设置演习阶段。
	参数类型	DWORD DownIP 目的 IP 地址
	返回值	int 型,通过 TS_RecvBackPackage()获取,如果命令发送成功,等于 5,表示发送的字节数。否则返回一个 SOCKET_ERROR 错误。
	所属动态库	UdpProcess.dll

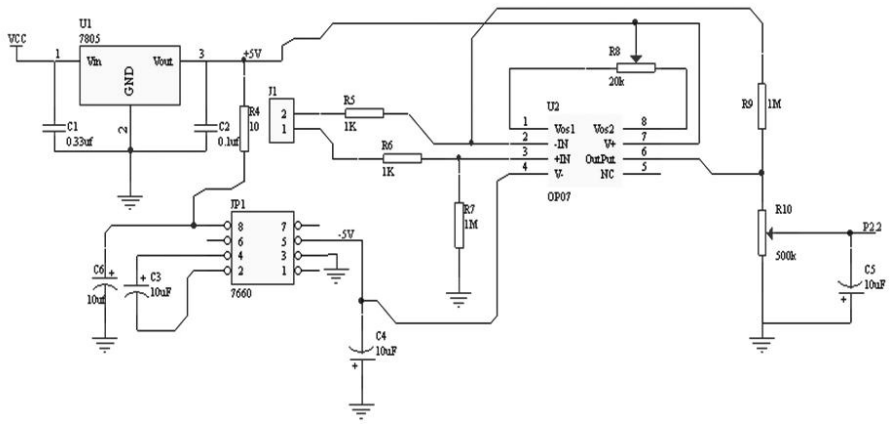

图 2　放大电路图

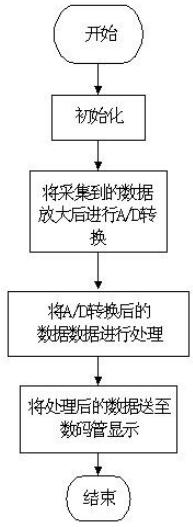

图 3　程序流程图

P2.2 管脚,利用单片机的 A/D 转换功能,将模拟量转成数字量。

4 软件设计

本系统采用 C 语言进行程序设计,并在编程中对数据进行处理,提高了开发调试的工作效率。程序语言编辑环境用 Keil C51 进行编写。系统流程图 3 所示。

5 结束语

设计主要利用 C8051F 系列单片机与 8051 指令值完全兼容的 CIP-51 内核，它在单片机内集成了一个构成单片机数据采集或控制系统所需要的几乎所有模拟和数字外设及其功能部件,使电路简单,成本降低。

参考文献:

[1] 宋文绪,扬帆.传感器与检测技术[M].北京:高等教育出版社,2004.
[2] 童长飞.C8051F 系列单片机开发与 C 语言编程[M].北京:北京航空航天大学出版社,2005.
[3] 马忠梅,籍顺心,张凯,等.单片机的 C 语言应用程序设计[M].北京:北京航空航天大学出版社,2007.
[4] 潘琢金.C8051F350/1/2/3 混合信号 ISP FLASH 微控制器数据手册[Z].沈阳:新华龙电子有限公司,2005.
[5] 鲍可进.C8051F 单片机原理及应用[M].北京:中国电力出版社,2006.

(上接第 2283 页)

7 结束语

任何一个表示法都不能充分编档接口;专业人员必须采用多种表示法,组合使用。在视图主表示中展现接口存在时,应该选用图形符号。应该采用一种语法表示法为接口说明的语法部分编档。应该采用自然语言、记录前置和后置条件的布尔代数或任何描述行为的语言来表达语义信息。

参考文献:

[1] Clenments P,Bachmann F,Bass L,et al.软件构架编档[M].朱崇高,译.北京:清华大学出版社,2002.
[2] Bass L,Clements P,Kazman R.软件构架实践[M].2 版.车立红,译.北京:清华大学出版社,2004.
[3] 张友生.软件体系结构[M].2 版.北京:清华大学出版社,2006.
[4] 王映辉.软件构件与体系结构——原理、方法与技术[M].北京:机械工业出版社,2009.
[5] 王广昌.软件产品线关键方法与技术研究[D].杭州:浙江大学博士学位论文,2001.
[6] Bass L,Clements P,Kazman R.软件构架实践[M].2 版.车立红,译.北京:清华大学出版社,2004.
[7] Kuusela J,Savolainen J.Requirements engineering for product families[C].Proceedings of the International Conference on Software Engineering.Limerick,Ireland,2000.
[8] Griss M.Domain engineering and reuse[J].IEEE Computer, Roundtable on Software Development Trends,1999(5).