CTC控制台系统的设计与实现方法

张德明 董金明 黄 康 U28 A

摘 要：给出了一种CTC控制台系统的设计和实现方法，包括系统的设计分析，结构设计，行为设计以及系统的实现过程，并对系统开发过程的实践进行了总结。

关键词：CTC 控制台系统 面向对象 系统分解 概要设计

A Method for Design and Implementation of the CTC Control System

Zhang Deming Dong Jinming Huang Kang

(Beijing University of Aeronautics and Astronautics, Beijing 100083)

Abstract: This paper introduces a method for design and implementation of the CTC control system. It includes the design analysis, structure design, behavior design and the process of the system implementation. Some ideas that come out at the development process are also summed up here.

Key words: CTC ,control system ,object oriented ,system partitioning ,high-level design

1 引言

CTC(Centralized Traffic Control)系统是铁路通信信号中的一种调度集中控制系统，在英文中它还可以解释为Relay Interlocking，也就是一种间接联锁的概念。它的基本思想就是采用机械或者电子的方式，将一些单独的铁路线段的控制作业集中起来，以提高控制管理水平，节省劳力资源，增加铁路运输效率。在国外，自1927年开始，这种系统已经逐步广为应用。然而，由于某些历史的原因，以及中国铁路的复杂性，这种系统在国内开始着手研制和应用在最近几年才开始。

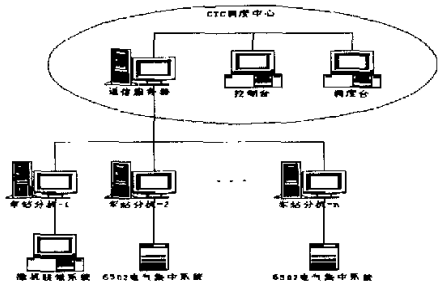

图1 CTC控制台系统相关系统框图

图1给出了我们研制的CTC系统的主要部分框图。从该图中可以看出，CTC系统分两大部分：车站部分和调度中心部分。对于每个CTC系统，有一个调度中心，这个调度中心对应所管辖的调度区段的车站。各车站可以有不同的控制方式，但都按照统一的通信格式将列车运行相关信息上传到调度中心，调度中心根据这些信息，作出判断，然后由控制台发送控制码给各车站，各车站接收控制码并执行控制作业。

从上面的叙述中可以看出，控制台在其中扮演着核心的角色。根据需求，控制台系统需要具有两大功能：站场显示功能和站场控制功能。而其中，站场控制功能又分为人工，存储，自动3种，而且能够对每一个车站在这三种控制状态之间进行切换。

张德明 北京航空航天大学电子工程系 硕士研究生 100081 北京市
董金明 北京航空航天大学电子工程系 教 授 100081 北京市
黄 康 郑州铁路局 工程师 450052 郑州市

2 设计分析

设计分析就是指在系统设计之前，对系统的设计方法的选择。在控制台设计中，我们选取了面向对象的方法，并从结构和行为两个视角来看待系统。

2.1 面向对象方法的选取

我们选择面向对象的方法来设计控制台系统，基于如下3方面的考虑：

（1）市场方面的考虑。目前，面向对象的开发方法已经成为市场上开发方法的主流，而且就将来的发展趋势来看，这种开发方法发展势头仍然相当强劲。随着UML标准的建立，面向对象的方法已经形成了一套基本的科学体系。另外，各种CASE工具也都主要集中在面向对象开发的功能特性上。

（2）面向对象方法本身特性的考虑。面向对象方法之所以流行，决不仅仅是因为众多的开发人员爱赶时髦，而是因为这种方法本身确实具有一些内在的良好特性。由于面向对象所具有的封装，继承和多态的特性，相较于另外两种常用的设计方法：即面向数据的方法和面向行为的方法，面向

对象的方法在系统的易建模性，易维护性和易重用性方面均具有开拓性的优点。

（3）控制台系统特性的考虑。尽管面向对象方法具有广泛的适应性，但这并不表示所有的系统采用面向对象方法是最优选择。一般来说，当系统具有一定的复杂性，适合于采用对象建模，并对系统的重用性和维护性具有要求时，就比较适合于采用面向对象方法。在这里，控制台系统需要对站场对象建模，这些站场对象模型已经在相关项目中存在，另外，与控制台系统相关的模拟车站系统，在功能上与控制台系统具有很大的相似性，需要控制台系统的设计便于重用，而且，对于这种实用系统的维护性需求也是必然的。因此，从控制台系统本身的特性上来看，选择面向对象方法也是必然的。

2.2 从结构和行为两个视角看待系统

选择了面向对象方法，在系统设计上已经把握了一个重要的基础方向，但就具体如何应用这种方法，却尚不知晓。在结构化程序设计（包括面向数据和面向行为的设计）时期，有一个著名的等式，即：程序 = 数据结构 + 算法。实际上这里所指的主要是面向数据的程序设计的思路。怎样叫做面向数据呢，就是建立一堆数据结构，并设计对这些数据结构进行处理的算法。什么是面向行为呢，就更简单了，系统就是一系列大大小小的过程调用。其实，对于面向对象的设计，很早以前也有一个思路：程序 = 对象 + 消息。

前面对于各种方法下程序本身的认识，都具有相当的抽象性和准确性，在一定时期对于指导软件开发起过很重要的作用。但是，只要认真一看，就会发现，前面的各种定义，都太朴素，对软件开发的认识还停留在设计和编程揉和在一块的概念上。随着软件系统本身越来越复杂，软件开发过程的分工越来越细，我们对待程序的认识也要跟着改变。程序设计成为软件开发，程序也不再仅仅是原来概念上的死的“程序”，而用“软件”似乎更加准确了。

在对控制台系统设计中，我们采用了“软件 ≈ 结构 + 行为”的概念。这实际上是从前面的“程序 = 对象 + 消息”演化而来的。对于简单的程序，建立几个对象，然后根据系统的功能添加这些对象之间的消息，系统就构建起来了。然而，对于复杂的系统，有许多的对象（如上百个），要想简单的在这些对象之间建立消息，几乎是不可能的。通常我们采用的办法就是按照强内聚，松耦合的原则将这些对象分组，然后在各组内和组间建立消息关系。这种分组所产生的结构，就是软件的结构。而组与组之间对象的消息传递过程，以及组内对象之间的消息传递过程，就是软件的行为。

从表面上看，这里的“结构”和原来的“对象”，这里的“行为”和原来的“消息”，只不过是一些概念的轻微演变而已。但这种演变实际上是一种认识上的重要突破。这里的结构不再是单个的对象，而是层次化的对象，是一种有组织的系统结构。这里的行为更不是简单的消息，而是系统为完成其功能所具有的一系列相关对象和消息所构成的行为过程。从结构和行为两个方面，也就是从空间和时间的角度，反映了系统的静态和动态特性，从而完整的描述了一个系统。

这种认识角度对于处理各种问题，无论简单的，复杂的，都具有很好的解决问题的能力。下面两节就分别从结构和行为两方面，给出了控制台系统的设计思路。读者可以从中看到我们是如何利用这种视角来构建系统的。

3 结构设计

从两个层次给出控制台系统的结构设计。第 1 个层次是系统的结构图，也就是系统的模块划分。第 2 个层次是系统的类关系图，反映了系统的类在系统模块中的分组情况。第 2 层次是第 1 个层次的细化。两个层次一起形成了系统的结构设计。

3.1 系统结构图

图 2 给出了控制台系统的基本结构图。图中把控制台系统主要划分成 4 个部分，即界面部分、显示部分、控制部分和通信部分。其中，显示部分和控制部分对应于系统的两个主要业务功能需求。而界面部分和通信部分则是为实现上面两个需求而产生的实现上的功能部分。界面部分提供人机接口，即从用户获取操作命令，或者向用户输出状态和结果；通信部分则负责接收和发送信息。另外，该图还给出了系统各模块之间的主要交互关系。

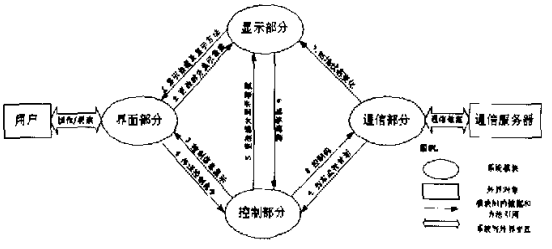

图 2 系统基本结构图

作为控制台系统的结构设计，图 2 是整个系统设计的基础和基本蓝图。在进一步的设计中，则围绕此图进行扩展，可在此基础上补充全局访问数据和 LOG 信息记录部分。另外，我们还在此图基础上补充了各模块对各种文件，如站场图形文件、进路表文件的访问关系。从而形成一张完整的系统结构图。

3.2 类及关系

形成了系统的结构图后，对系统进行设计的进一步任务就是抽象出各模块内部的类，并建立类之间的关系。

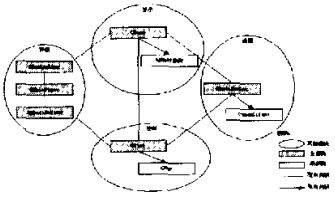

图 3 控制台系统主要类关系图

图 3 给出了控制台系统的主要类关系图。图中界面部分有三个主要类：CControlApp，CMainFrame 和 CStationView。这三个类都是从 Microsoft Foundation Class 继承来的，共同构成一种单文档界面。这 3 个类彼此交互，并且都与系统其它部分交互，构成系统与用户接口的基础。显示，控制和通信部分均只有一个主要类。显示部分的主要类是 CDisp，它负责与站场对象组织和显示相关的任务。一系列根据站场对象抽象出来的类作为显示部分的局部类，由 CDisp 类来协调和组织。控制部分的主要类是 CCtrl 类，它负责与控制业务相关的各种功能。CWay 类是对站场进路的抽象，只在控制部分使用，作为局部类由 CCtrl 调用。通信部分的主要类是 CDataEngine，它封装了通信部分与其它部分的接口。CSockClient 则负责实际的通信交互，由 CDataEngine 调用。

当系统的类关系图建立后，系统的结构就比较清楚的建立了。

4 行为设计

同结构设计一样，对系统的行为，按照特征不同，也需要进行分解。对控制台系统，我们对其行为分解为两部分。一部分给出系统建立和退出过程，也即系统的生命线过程；另一部分则给出系统运行过程中的各个主要功能的实现过程。

4.1 系统生命过程

图 4 给出了控制台系统的生命过程：系统启动时，构造主要对象，退出时，析构主要对象，在系统登录后，系统作一些必要的登录处理，然后，系统进入正常运转，根据各种输入完成对应功能，系统注销时，作一些注销处理，然后退回到登录前的状态。

系统生命过程是对系统行为的最初略描述，但它是认识系统主要功能处理过程一个必要前提。图 4 中灰色矩形框是除系统主要功能过程外的一些系统行为。在设计中，将这些行为是放在系统生命过程部分进行一些设计描述的，而将系统主要功能处理部分单独提出来进行设计描述。

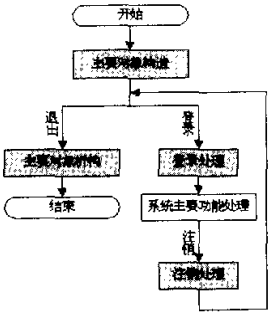

图 4 控制台系统的生命过程

4.2 系统主要功能过程

系统主要功能过程是系统实现中与系统功能需求完全对应的部分，系统设计的其它部分都是基础，是为主要功能过程实现服务的。

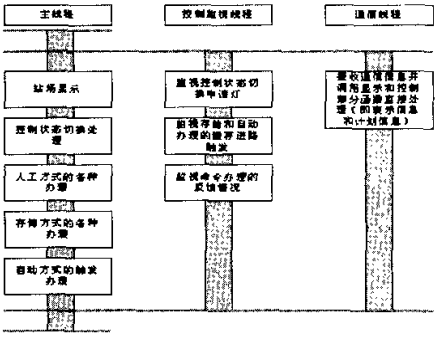

图 5 系统多线程关系图

图 5 给出了控制台系统的多线程关系图。图中灰色的柱形表示各个线程的生命线。在登录与注销之间，各生命线上有一些矩形文字，就是说明在系统登录期间各线程完成的各种功能。

在控制台系统的行为设计中，引入多线程是由于系统本身的复杂性要求。但是，为避免多线程带来的系统行为的混乱，采用以一个线程为主，其它线程为辅的方式，即将只有必要由单独线程处理的任务才交由另外线程处理。

在图 5 中，只是给出了各功能在各线程之间的分工，在进一步的设计中，对于系统要实现的每个功能，给出一个由各结构体在三个线程之间如何协作完成的详细过程，从而形成一个完整的系统行为。

计算机用于车站接发列车作业的开发与实践

钱国伟

U29 A

摘 要：应用计算机控制技术，开发并建立车站接发列车安全联控系统装置。论述了系统开发的作业过程、目标和功能、原则和标准以及探索和实践。

主题词：接发列车 安全联控新技术 开发与实践

1 引言

铁路车站接发列车作业始终是铁路行车作业安全的一个重点。目前，部分三等站车站值班员的工作中所使用的工具仍然靠笔、电话、行车日志，行车作业过程的信息传递，仍以作业人员按作业标准进行电话传递，主要靠相关作业人员人工连锁控制，由于人的生理因素、通话质量等原因，易构成差错，造成列车运行信息的错误传递或漏传，导致危胁行车安全，甚至造成严重的行车事故。

铁路列车已全面提速，以某三等站为例，每办理一趟列车作业，需通知助理值班员、道口看守员、客运员、行包员、售票员、广播员、调度联系（有的车站还有车辆部门、机车部门、邮政部门）等十几个作业岗位，通知内容为前站预告、前站报开、列车接近、列车到达、向后站预告、向后站报开、向前站报到、后站报到等8个电话，一趟列车需办理各种电话近40次，如该站图定列车230列，24 h则需办理电话18000余次，平均每5s种一次电话.目前，正在应用的新技术中，如列车调度监控系统、车号识别系统、车站信息管理系统、列车到发通知系统等系统由于其各自的应用对标标禾和平共处象的不同，尚没有涉及到车站内部的接发列车的信息共享和作业联控。

为此，探索车站接发列车作业过程的技术创新，应用计算机控制技术，开发安全联控新系统，以确保列车安全运行迫在眉睫。

钱国伟 杭州铁路分局乔司站 工程师 311101 余杭市

5 实现过程

这里所论述实现过程，是指将设计转换为编码的过程，还不是仅指的通常意义上的编码。前面两节给出的只是系统的概要设计，有许多具体问题没有考虑，如类内部的数据和操作的设计，类对象本身的状态变化设计等。在控制台系统的实现过程中，我们分别针对一个或者一部分细节进行设计，然后进行编码，构成可运行系统，然后进行下一个细节设计，下一步编码，按照这样一个过程把系统构建完毕的。在图6控制台系统开发过程中，清晰的表示了设计和编码过程中采用的迭代式的开发过程。

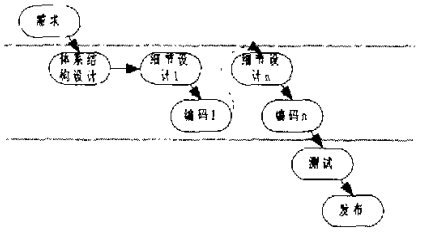

图6 控制台系统开发过程

6 结束语

在CTC控制台系统的开发中，结合实践，对一些方法进行了有意的探讨和尝试，获得了很好的效果。总结主要有3点经验或结论：1.选择面向对象的方法；2.从结构和行为视角来构造系统；3.先进行概要设计，然后在细节设计和编码过程中采用迭代方法。另外，在一些方面也值得继续改进，如在设计文档的图形表示方法上的标准化，在系统和模块的状态机行为设计和描述上，以及在迭代过程中设计文档的编制和评审安排上。

7 参考文献

1 Roger S. Pressman, Software Engineering: A Practitioner's Approach, 4 th ed, McGraw-Hill, 1997

2 Jeffrey L. Whitten, Lonnie D. Bentley and Kevin C. Dittman, Systems Analysis and Design Methods, 5th ed, McGraw-Hill, 2001

3 Grady Booch, James Rumbaugh, Ivar Jacobson, UML 用户指南.邵维忠.北京：机械工业出版社，2001

（收稿日期：2001-11-20）