应用于远程教育的在线协同作图平台

陈喆　吴志军（清华大学 精密仪器与机械学系 100084）

Online Cooperative Drawing Platform Designed for Long-distance Education

摘要：本文就应用于远程教育的在线协同作图平台开发过程中的一些主要方面做了探讨与实践。针对要实现的功能和相应的系统体系结构，重点讨论了作图平台设计中的一些问题，提出了解决问题的方法和思路，主要论题包括：图形的数据结构描述及算法库、网络连接及通信、多线程技术的应用。

关键词：在线协同作图平台　Java　网络通信

1 引言

基于internet的远程教育是利用网络通信技术、计算机技术、多媒体技术和现代教学方法进行教学活动的一个系统。远程教育提供了生动、友好、多样化的人机交互界面以及方便、灵活的人机交互方式，给学生一个身临其境的环境，并且通过提供诸如答疑教室、讨论区等功能，实现了人与人之间的异地交互、信息共享，从而提高了学生的学习热情和学习效率。

为了满足课程中部分作图练习的需要，弥补现在网络教学过程中缺少图形信息交流的不足，提供一个简单易用的在线作图工具，使教师和学生通过文本信息和矢量图信息实时交流，用Java语言开发了在线协同作图系统。

2 在线协同作图平台功能设计

在线绘图平台的目的是提供一个简单易用的作图工具，有实用的绘制二维图形的功能，满足网络课程的在线作图的要求。参考常见绘图软件，结合平台“在线作图”的特点，确定了平台的绘图功能。

协作性是网络课程的特点之一，单纯的文字交流形式的聊天室缺乏对图形的支持，对于“以图说话”的课程来说不能很好的体现课程特色，更为复杂的视频会议形式成本昂贵，需要高条件的硬件支持，且目前应用效果并不理想。在线协同作图平台的开发是为了实现信息交流的从非实时的文本信息交流扩展到非实时的图形信息交流，并在此基础上，将网络教学过程中的信息交流从非实时的图形信息交流，提高到实时的图形信息交流的水平。

作图平台客户端进行图素绘制、图形编辑、视图的放大、缩小、平移等数据处理工作，服务器则除此之外还负责客户端协作管理以及通信调度等功能。服务器是开启协同作图服务的特殊用户，协作服务包括文本信息的交流服务和图形信息的交流服务。他可以将本机或接收的用户图形数据发送到连接进入协作空间的每一个用户，也可以通过文本信息的发送和接收功能，与别人进行文本信息的交流。客户端在向服务器申请了作图权限后将本地图形发送至服务器，由服务器再分发出去。平台的在线作图及协同功能如图1：

系统采用Java Applet作为在线绘图平台的开发方案，使得绘图平台基于浏览器运行并具有跨平台的特点。Applet在运行时可以有自己的GUI界面，可以处理鼠标事件，并且可以使用Java中的有关图形处理包实现基本图形元素的绘制，基本可以满足在线绘图平台功能的需要。通过签名的方法可以解决Applet的安全性限制问题，如缺少本地文件访问能力和进行打印操作。

3 在线协同作图平台体系结构

采用面向对象的分析和设计方法，在设计阶段的目的之一是要确定系统中的类（Class）和对象（Object）以及对象之间的关系，在线绘图平台系统中包含的类和对象的组成构成了系统的体系结构，用Class框图表示如图2：

图2是一个简化的Class框图，这里忽略了每个对象的细节，每个对象只给出了对象的名字，而没有指出具体的属性和方法，重点在于描述系统的体系结构，其中以“用户界面类Applet”为核心，该类主要任务是处理与用户的交互，使用工具栏和其他GUI元素获取用户的命令输入，

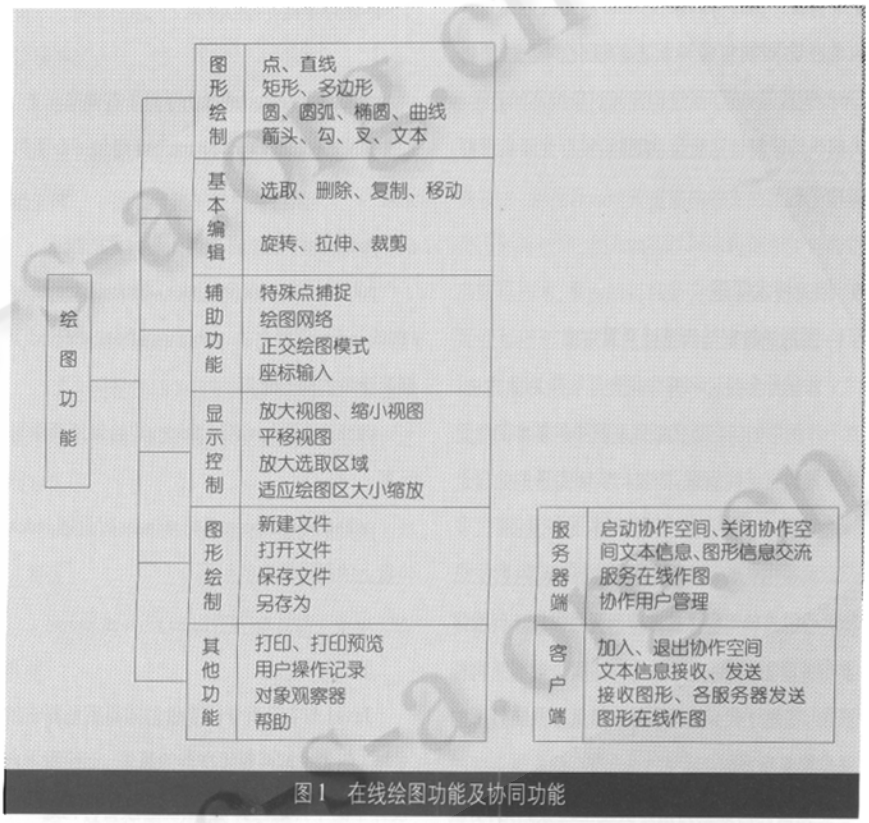

图1 在线绘图功能及协同功能

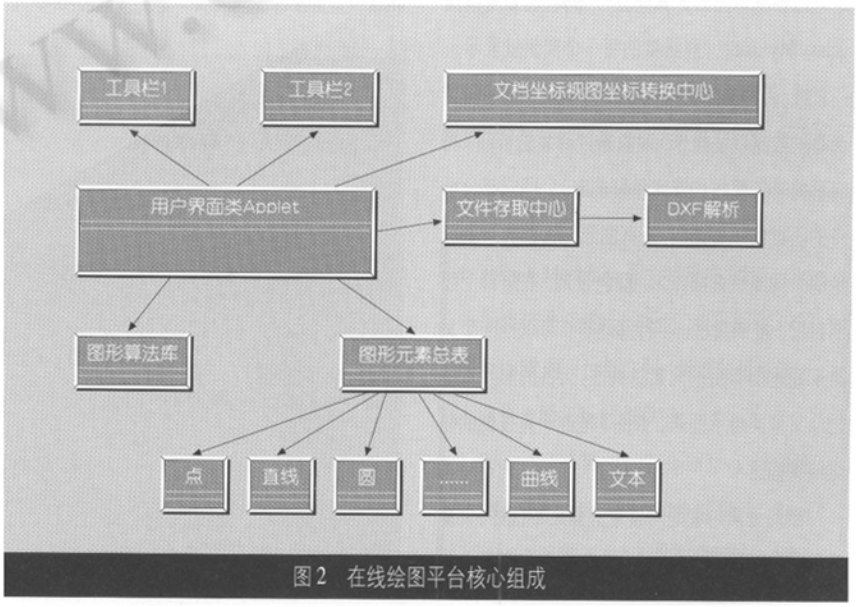

图2 在线绘图平台核心组成

并进行相应的处理。同时还要维护一份"图形元素总表"，其中存放所有的图形元素。其他的类的功能也都一目了然。

详细设计阶段的工作是对这个Class框图的进一步细化，明确每个类和对象的属性、方法，在需要的情况下，采用其他的框图来描述类和对象的特性，例如可以采用State框图描述某个对象的生命周期中各种状态的变化。通过细化Class框图，形成了系统构建和对象编写的蓝图，为编码阶段制订了框架，编码阶段专注于实现框架中的细节。

4 相关技术实现

4.1 图形的数据结构描述及算法库

作图平台设计和开发采用了面向对象方法。对于作图中如何描述图纸以及其中的基本图形元素等是数据结构问题，而如何实现图形的处理是算法问题。绘图平台处理的对象是矢量图，

它是由各种基本图形元素组合而成，数据结构关心的是如何描述基本图形元素以及如何描述整个矢量图。用面向对象的方法实现数据结构和算法，提高了代码重用程度，降低了系统分析的难度和复杂度。

用面向对象的方法来实现上述数据结构，就是对每种基本图形元素建立一个类，例如直线类(class MyLine)，直线类的每一个实例就表示一条直线，每种基本图形元素作为一个对象包含了自身的数据以及操作，对象通过封装的机制，将自己特有的数据和操作包装起来，只留出部分公开的函数供外界调用，所有图形元素都具有的其他的共同属性或操作如：颜色设定、在屏幕上绘制自己、复制自己、平行移动等，可以用一个父类来描述所有图形元素的共性，而各图形元素类则从父亲类继承而来。如图3是对于直线类的数据结构描述：

图形元素总表是对整个矢量图的描述，矢量图是由各图形元素组合而成，每个图形元素之间相互是独立的，一幅矢量图中有多少图形元素是不确定的，其数量在运行中会增加或者减少。基于上述考虑，采用Java提供的Vector来存放所有图形元素，Vector可以当一个链表来使用，以此为核心，构造图形元素总表类ShapeTable，其定义如下：

```
public class ShapeTable
{
protected Vector cVct;// 存储所有图形元素
public void add(MyShape p)// 增加一个图形元素
public int count()// 取得图形元素的数目
public void delAt(int index) throws Exception// 删除 index 位置上的 public void delSelected()// 删除选中的图形元素
public void draw(Graphics g)//绘制所有图形元素
public MyShape getAt(int index)// 返回 index 位置上的图形元素的引用
……
}
```

Java1.1中没有专门提供对常见图形算法的支持，例如求解两直线交点的算法，而绘图平台设计中离不开这些算法，因此必须自行构建一个算法库，将这些算法包装到一个类MyMath中，而每种算法都作为该类的一个成员函数，把该函数声明为静态(static)，这样就成为一个类的函数，不需要实例化MyMath类就可以使用该函数。具体算法可参考已有的一些关于计算机图象学的算法理论，并通过Java编程实现。

4.2 网络连接及通信

协同作图系统要解决两种信息的共享问题，一种是图形信息，一种是讨论的文本信息。图形信息是可编辑的，例如可以增加一条线，也可以删除一条线。讨论的文本信息的特点是不可编辑，说出来的话就不能收回了，文本信息只有不断的增加，而没有将说过的话进行编辑的操作。两种信息的特点决定了两种不同的信息共享模型。

使用系统的有两种角色(Actor)，主席和听众，即系统的服务器端和客户端，每个协同作图空间内，主席只有一个，而听众可以有多个。图形、文本信息的实时交流在这两种角色之间进行。通过定义两种角色的权限，保证了共享数据的一致性。

图形信息的共享是通过将主席的图形数据发送给每个听众来实现的，主席拥有对图形数据的完全控制，可以编辑修改。图形数据对听众来说是只读的，主席可以决定在适当的时候进行数据更新，将自己的图形数据发送给每个听众，保持

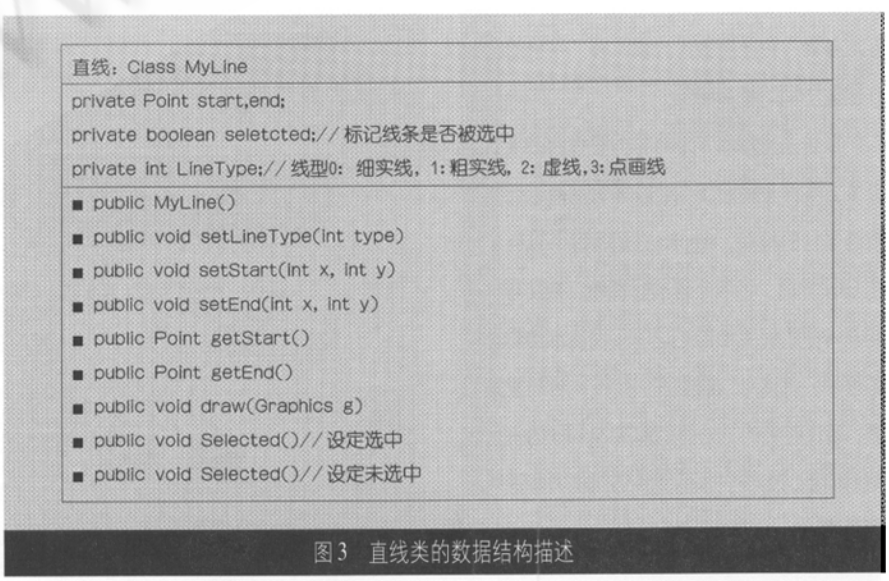

图3 直线类的数据结构描述

数据的一致。

讨论的文本信息的特点是只增加不减少，每个人随时可以发言，并且发言每个人都能够看到。在这种模型中，主席同听众是对等的，每个人的发言首先进入讨论缓冲区，然后再从讨论缓冲区中获得所有人的发言。讨论缓冲区的位置在讨论服务器上，通常也就在主席的系统中。

协同的实现建立在因特网的TCP/IP传输协议基础上，用户和服务器之间实现通信首先是建立TCP连接，连接通过套接字来实现。Java提供的Socket类可以创建套接字端口，服务器端可以用ServerSocket类来创建的监听套接字。协同作图系统中建立一个TCP连接的过程大致如下：

服务端构造ServerSocket，并调用其accept()方法监听客户端的连接：

```
server = new ServerSocket(port,backlog,
myHost);
while(islistening){
    Socket socket;
    socket = server.accept();
    clientSockets.addElement(socket);// 连接
建立后存放到连接列表
}
```

客户端构造Socket，构造时指定了服务器，在构造的同时就开始连接服务器，连接成功后客户端就获得了套接字端口，可以用Socket的getInputStream()方法和getOutpuStream()方法，获得输入输出流，输出流将数据发送到服务器的套接字，输入流则获得服务器送来的数据；客户端连接成功的同时，服务器端accept()方法也成功返回一个服务端的套接字端口，同样利用该套接字端口可以获得输入输出流，实现与客户端的通信。

4.3 多线程技术的应用

在协同作图平台中不论是服务器还是客户端，都可能会在同一时间执行多个任务，例如一边要接收其他人送来的讨论信息，一边要将自己说的话发出去给其他人，程序中如果只有一个执行流程，这两个任务就只有先后进行；如果有两个不同的执行流程，一个接收数据，一个发送数据，这两个任务就可以并发执行，提高了实时性，这就利用了Java的多线程技术。

服务器要处理多个客户端之间的数据传输的请求，例如在用户交谈时，服务器要创建一个讨论缓冲区：

```
dataPool = new ChatDataPool();
```

在每一个新的Socket连接建立之后，服务器就生成一个接收数据的线程并启动它：

```
MyChatListenThread lsnThread = new
MyChatListenThread(this,socket);
lsnThread.start();
```

这个线程的任务是随时等候来自网络(也就是来自客户端)的数据，收到之后将其放到讨论缓冲区中：

```
while(parent.islistening){ // 服务器关闭则停
止接收
String str = dis.readUTF();// 从输入流中读取
数据
pool.pushData(str);
}
```

这一过程如图4所示：

同时为了避免多个用户向服务器发送引起冲突，必须使向讨论缓冲区写入的方法为同步，即每次只能有一个线程添加信息：

```
synchronized public void pushData(String data)
```

发送信息给所有用户的过程原理类似，在此不再赘述。服务器通过使用讨论信息的缓冲区和大量使用线程并发执行各种任务，利用多线程实现了发送数据、接收数据等多项任务的并发执行。

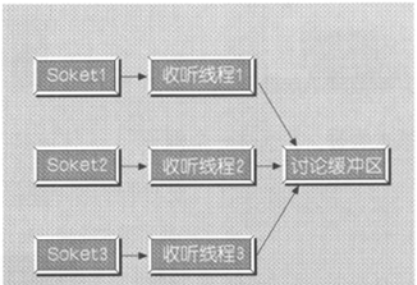

图4 Java多线程的应用

5 结束语

本文研究和论述了应用于远程教育的在线协同作图平台的设计和技术实现方案。文章首先提出了作图平台要实现的功能设计；然后通过面向对象的设计和开发方法讨论了系统的体系结构；在此基础上，结合绘图平台开发讨论了其中的关键技术，包括Java对数据结构的支持以及自行构造图形算法库、用Java实现网络通信和如何利用多线程技术。平台的特点是突出其在应用中的交互性和图形文字的信息共享性，因此如何利用Java的相关技术实现和完善平台的功能是本文的重点，也是今后深入研究的重点。

由于开发时间、技术实现能力以及教学理论等方面的不足，随着实践的深入和技术的发展，这一课题仍有继续探讨的价值，网络技术和Java技术日新月异的进步也将使更多更好的设计方案成为可能，如Java2的成熟将使在线协同作图平台从数据结构描述和图形算法上更优化，可实现的功能更复杂，进一步扩展和完善。网络教学将出现更多的形式，远程教育的建设也会出现更多新的变化。

参考文献

1 Joshua Marketos 著，杨武杰 等译，Java编程技术与技巧，电子工业出版社，1997。

2 周培德 著，计算几何--算法分析与设计，清华大学出版社，2000,3。

3 Bruce Eckel 著，Thinking in Java，电子版。

4 John Zukowski 著，邱仲潘 等译，Java2从入门到精通，电子工业出版社，1999。