

基于 UML 的数据流图可视化编辑工具的设计

陈荟慧, 王伟静

(洛阳理工学院 计算机与信息工程系, 河南 洛阳 471023)

摘要:数据流图可视化编辑工具 SECAI 是适用于传统的面向数据流需求分析的一种工具软件。SECAI 是为了满足需求分析时绘制数据流图和编写数据字典的需求, 因此它的主要功能包括: 绘制分层数据流图、保证数据流的一致性以及记录外部实体的信息、记录加工的信息、记录数据流的信息、记录数据存储(文件)的信息并根据以上信息自动编写数据字典。文中使用 UML 描述了数据流图可视化编辑工具的设计方法, 设计了数据流一致性保持算法并给出了运行结果。

关键词:数据流图; UML; 可视化编辑; 分层; 一致性

中图分类号: TP311.56

文献标识码: A

文章编号: 1673-629X(2012)05-0145-05

Design of Visual Edit Tool for DFD Based on UML

CHEN Hui-hui, WANG Wei-jing

(Dept. of Computer and Information Engineering, Luoyang Institute of Science and Technology,
Luoyang 471023, China)

Abstract: DFD visual editing tool SECAI is a software for traditional data-flow oriented requirement analysis. Because SECAI is designed for drawing DFD and making data dictionary as requirement analysis result, the main functions of SECAI include drawing hierarchical data flow diagram, remaining the consistency of data flow, recording external entities, recording processes, recording data flows, recording data stores and building data dictionaries according to information recorded automatically. In this article, describe the procedure of system design by UML notations, and give a running image of this system.

Key words: data flow diagram; UML; visual editing; hierarchy; consistency

0 引言

面向数据流的需求分析方法是传统的软件工程需求分析方法, 尽管目前需求分析大多采用面向对象的需求分析方法, 但结构化分析在把握软件整体设计思路中显示出来的优势是不容忽视的, 因此, 在面向对象分析广泛应用的今天, 数据流图以其提供强大的信息量这个特点, 在软件的开发中仍占有一席之地^[1]。数据流图编辑工具的研究在没有 Windows 系统时就已经开始, 文献[2]中介绍了在没有可视化编程环境时, 数据流图的编辑和绘制系统的开发, 以及数据流图一致性维护的方法。文献[3]提出类图的一致性检查方法。文献[4]提出面向对象的集成化 CASE 平台。文献[5, 6]提出了面向数据流分析方法与面向对象分析方法的结合。由此可见, 在软件系统分析时, 把面向数据流分析方法与面向对象方法有机结合才能取得更好的效果。

采用面向数据流的需求分析方法^[7]可以得到系统的数据流图、数据字典、E-R 图。数据流图 DFD (Data Flow Diagram) 是描绘信息流和数据从输入移动到输出的过程中所经受的变换, 从数据传递和加工的角度, 以图形的方式刻画数据流从输入到输出的移动变换过程^[8]。数据字典是关于数据的信息的集合, 也就是对数据流图中包含的所有元素的定义的集合。数据流图也为设计 E-R 图提供了依据。

现有的绘图软件功能虽然强大, 但并不适用于面向数据流的需求分析, 如 Visio 仅能够绘制数据流图, 不能够检查分层数据流图的平衡性和一致性。文中介绍的数据流图可视化编辑工具的主要功能除了能够利用可视化环境绘制分层数据流图, 还要保证数据流图的一致性和完整性, 并自动生成数据字典。文中采用 UML 描述了数据流图可视化编辑工具的设计过程。

1 用例模型

数据流图可视化编辑工具是为了满足在采用面向数据流的需求分析方法时, 绘制数据流图和编写数据字典的需要, 因此, 软件的主要功能概括起来即: 记录

收稿日期: 2011-10-24; 修回日期: 2012-02-01

基金项目: 河南省教育自然科学基金项目(2008B520025); 洛阳理工学院教研项目(10-JY057)

作者简介: 陈荟慧(1978-), 女, 硕士, 讲师, 研究方向为并行计算。

外部实体的信息、记录加工的信息、记录数据流的信息、记录数据存储的信息、显示分层数据流图和自动编写数据字典。

数据流图可视化编辑工具的主要特性包括:

1. 用户可编辑数据流图中的各个元素的属性。
2. 根据数据流图元素的属性,自动绘制数据流图。
3. 自动分析父子图的平衡关系,并自动为外部实体、加工、数据流、数据存储编号。
4. 自动保持父子图的平衡性和一致性。
5. 自动生成数据字典。

1.1 系统活动图

活动图用于研究实现数据流图可视化编辑工具时所执行的各项任务或活动的顺序安排。图 1 所示的活动图描述了系统从创建数据流图至保存数据流图的一系列活动,图中大部分活动都是参与者的活动,因此,图 1 所示的活动图也可以视为系统的操作流程。

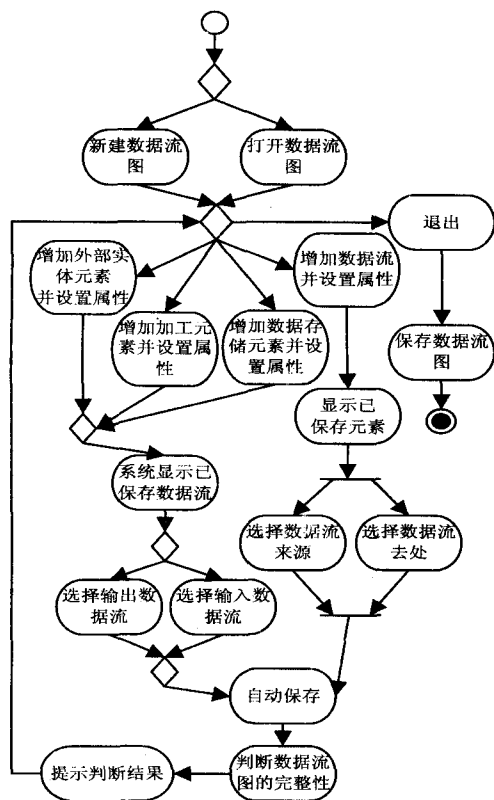


图 1 系统活动图

说明:选择输出数据流活动结点和选择输入数据流活动结点为选择关系,原因是外部实体和数据存储可以是只有输入数据流或输出数据流,而加工则需两者都有。因版面限制,图 1 只给出了新建数据流图和增加元素时的活动流程,未给出修改和删除等活动。

1.2 系统用例图

根据数据流图编辑工具的功能需求,得到系统的用例图如图 2 所示。

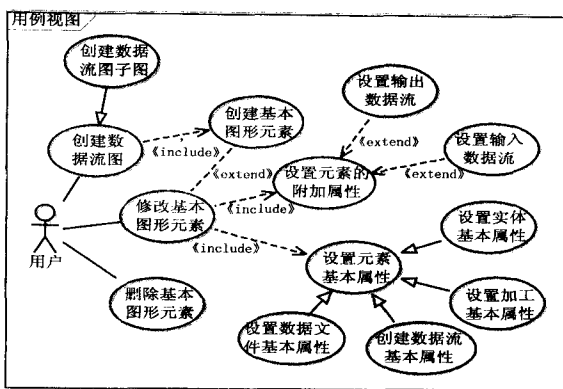


图 2 系统用例图

主要用例描述:

创建数据流图:初始状态下,创建顶层图,只有创建了加工图形元素后,可以为加工创建数据流图子图,依次为 0,1,2...层图。

创建数据流图子图:与顶层图用法类似,图中加工的编号和当前图的层号有关,其它元素除数据流外都需要记录当前的层号。

修改基本图形元素:创建数据流图元素对象后,需要设置它们的属性,包括各个元素的基本属性(编号、名称、说明),并把他们保存至数据流图文件。

设置元素的附加属性:数据流图各个元素的属性不同,除数据流外的三种元素包含流入的数据流和流出的数据流^[9]。

2 数据流图的一致性维护

2.1 数据流图的原则

每一层的数据流图都需遵循以下原则^[10]:

1. 一个加工的输出数据流不应与输入数据流同名,即使它们的组成成分相同。
2. 保持数据守恒。也就是说,一个加工所有输出数据流中的数据必须能从该加工的输入数据流中直接获得,或者说是通过该加工能产生的数据。
3. 每个加工必须既有输入数据流,又有输出数据流。
4. 所有的数据流必须以一个外部实体(数据的起点)开始,并以一个外部实体(数据的终点)结束。
5. 外部实体之间不应该存在数据流。外部实体之间的数据流可理解为计算机系统外的活动。

2.2 数据流图的一致性维护算法

在分层数据流图的显示中,数据流的正确显示是保证数据流图一致性的关键^[11],数据流两端的元素在显示层的显示标记非常重要。外部实体因为其特殊性,在各层数据流图中均须显示,因此一端连接外部实体的数据流全部需要显示;数据存储在第一次出现写入和读出双向数据流时的所在层开始,在以后的子层

中都需要显示,与之相连的数据流也必需显示;根据加工的编号和当前的显示层号可确定相连的数据流是否显示。这个算法的前提是每一个数据流只有一个起点元素和一个终点元素,且其它元素均不保存流入的数据流信息和流出数据流信息,需要时可通过遍历数据流元素得到。这样才能保证父子图的数据流平衡和一致。

以数据流两端为加工元素为例,数据流在子图中定义后,在父图中是否显示的具体算法如下:

假设第 s 层图中有 n 个加工 $P_x (1 \leq x \leq n)$, P_i 的子图记为 D_i , P_j 的子图记为 D_j 。

假设加工 P_n 的子图为 D_n , P_n 分解后的加工为 $P_n.m$, 起点或终点为 $P_n.m$ 的数据流集合为 FN , 其中 $P_n.m$ 之间的数据流集合为 FTN , 则继承自上一层父图的数据流集合为 $FHN = FN - FTN$ 。

F_i 连接 P_i 和 P_j 就是指 F_i 的起点为 P_i , 终点为 P_j 。若 $F_i \in FHN$, 有下列情形之一的, 则 F_i 连接 P_i 和 P_j :

- (1) F_i 连接 P_j 和 $P_i.k$;
- (2) F_i 连接 $P_j.r$ 和 $P_i.k$;
- (3) F_i 连接 $P_j.r$ 和 P_i ;
- (4) F_i 连接 P_i 和 P_j 。

3 类的定义

系统中使用了用户接口类、数据管理类和实体类, 用户接口类包括主界面 DFDToolFrm、四元素的创建与修改界面 (ProcessDlg、EntityDlg、DataFlowDlg、DataFileDlg)、数据字典生成界面 DDDlg 等, 数据管理类包括存储和处理数据流图信息的 DFDiagram 等, 实体类由 DFDItem、Process、DataFlow、DataFile、Entity、DD 等组成。

图3为系统类图。

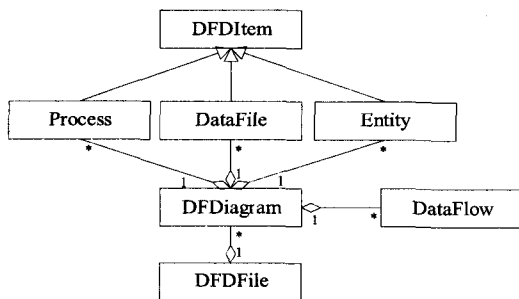


图3 系统类图

下面介绍主要的类和它们的属性与操作。

3.1 类 DFDToolFrm

类 DFDToolFrm 主要完成数据流图的绘制和元素列表的显示。可分为三栏, 左侧使用树形控件显示数据流图所有元素, 中间显示分层数据流图, 右侧显示选

中元素的数据字典。

```
Class DFDToolFrm
```

```
{
    DFDiagram * m_dfdiagram; // 数据流图对象
    void ShowDFD(); // 显示数据流图
    void ShowTreeOfAllItem(); // 显示元素树
    void ShowDD(DFDItem); // 显示元素的数据字典
}
```

3.2 类 DFDiagram

数据流图类 DFDiagram 存放数据流图所有元素, 元素所在层由元素的编号确定。

```
class DFDiagram
```

```
{
    CPtrList * dataflowList; // 数据流链表
    CPtrList * entityList; // 外部实体链表
    CPtrList * processList; // 加工链表
    CPtrList * fileList; // 数据存储链表
    void ShowDFD(int level, CDC * pDC); // 显示 level 层的数据流图, -1 代表顶层, pDC 为显示设备
}
```

3.3 类 DFDItem

这是类 Process、DataFile、DataFlow 和 Entity 的父类, 它们分别表示处理、数据存储、数据流和外部实体, 它们拥有共同性质: 名称、编号和备注说明, m_Pos 对数据流 DataFlow 类无用, 因此泛化为类 DFDItem。

```
class DFDItem
```

```
{
    CString m_name; // 名称
    CString m_ID; // 编号
    CString m_note; // 备注说明
    CPoint m_Pos; // 位置 x 和 y 坐标
    CStringArray * GetDD();
}
```

3.4 类 Process

加工元素实体类, 继承自 DFDItem, 加工可以附加子图, 通过加工列表 subProcessList 获取父子关系, 同时加工的编号也与加工的父子关系关联, 分层数据流图中的加工的显示与否根据加工的编号和当前的层号来判定。

```
class Process : public DFDItem
```

```
{
    CString m_processproc; // 加工的处理逻辑
    int m_redius; // 加工绘制半径
    int m_subDFDFlag; // 1: 有子图, 0: 无子图
    CPtrList * subProcessList; // 子图的加工列表
    int m_level;
}
```

```
CStringArray * GetDD();
void show(bool showflag);
}
```

3.5 类 DataFile

数据存储实体类,继承自 DFDItem,根据 m_Level 确定在分层数据流图中的哪一层显示。

```
class DataFile : public DFDItem
{
    int m_length;//长度
    int m_height;//高度
    CString m_level;//可见层
    CString composition; //文件的组成
    CStringArray m_proForRead;//读文件的数据流
    m_proForWrite;//写文件的数据流
    CStringArray * GetDD();//生成该元素的数据字典

    void Set ( CString com, CString nam, CString l,
    CString not);
    void show(bool showflag);
}
```

3.6 类 Entity

外部实体类,继承自 DFDItem,外部实体在分层数据流图中除顶层图和 0 层图外可以不显示,但为了分层数据流图中数据流更清晰,外部实体一直显示。

```
class Entity : public DFDItem
{
    int m_length;//长度
    int m_height;//高度
    CStringArray * GetDD();//生成该元素的数据字典

    void Set( CString nam, CString l, CString not);
    void show(bool showflag);
}
```

3.7 类 DataFlow

为保证数据流的来源和去处的唯一性,软件只在数据流类中保存来源和去处信息,分层数据流图中数据流的显示根据父子图的对应关系进行推导,不但保证子图边界数据流在父图中的显示,也保证了父子图的数据流一致性和平衡性。

```
class Dataflow : public DFDItem
{
    CString m_composition;
    CString m_sourceType;//数据流来源元素类型
    CString m_sourceID;//数据流来源元素编号
    CString m_destinationType;//数据流去处元素类
```

```
CString m_destinationID;//数据流去处字符串
public:
    CStringArray * GetDD();//生成该元素的数据字典

    void show(bool showflag);
}
```

4 数据流图显示算法

数据流图的显示除数据流元素外根据用户对各个元素的定位进行显示,数据流的显示通过获取数据流两端的元素的位置和大小来计算起点和终点,具体的显示算法如下:

S1:显示外部实体。

S1.1 读取外部实体链表,获得外部实体 ent;

S1.2 根据 ent 的名称长短确定元素显示时的大小;

S1.3 读取显示设备指针,根据 ent 的位置属性绘制相应的图形和文字;

S1.4 ent 的 next 节点不为空,转 S1.1,否则,结束。

S2:显示数据存储,同 S1。

S3:显示加工。

S3.1 读取加工链表,获得加工 proc;

S3.2 根据当前显示的数据流图的层号,判断 proc 显示则继续,否则,转 S3.5;

S3.3 根据 proc 的名称长短确定文字在元素中显示时的行数和位置;

S3.4 读取显示设备指针,根据 proc 的中心位置属性绘制相应的图形,并显示文字,文字不能越过加工元素的显示边界,多余文字以省略号代替;

S3.5 proc 的 next 节点不为空,转 S3.1,否则,结束。

S4:显示数据流。

S4.1 读取数据流链表,获得数据流 df;

S4.2 读取 df 的来源类型 stype;

S4.3 如果 stype 为外部实体或数据存储,查找其显示坐标 spos 和大小 ssize,转 S4.5,否则继续;

S4.4 df 的来源元素为加工,读取 df 的来源元素 sproc,根据 sproc 的编号确定当前是否显示,若显示,记录 sproc 的显示坐标 spos 和大小 ssize,继续,否则转 S4.9;

S4.5 读取 df 的去处类型 dtype;

S4.6 如果 dtype 为外部实体或数据存储,查找其显示坐标 dpos 和大小 dsize,转 S4.8,否则继续;

S4.7 df 的去处元素为加工,读取 df 的去处元素 dproc,根据 dproc 的编号确定当前是否显示,若显示,记录 dproc 的显示坐标 dpos 和大小 dsize,继续,否则转

S4.9;

S4.8 根据 spos、ssize、dpos、dsize 绘制数据流,在 dpos 处绘制箭头,若屏幕在相同的位置已绘制数据流,则在前数据流旁增加 df 的名称。

S4.9 df 的 next 节点不为空,转 S4.1,否则,结束。

5 系统动态模型

顺序图用于捕获系统运行中对象之间有顺序的交互,强调的是消息交互的时间顺序^[12]。根据数据流图的显示算法和图形元素类的定义,可得到图4所示的数据流图显示的顺序图模型。

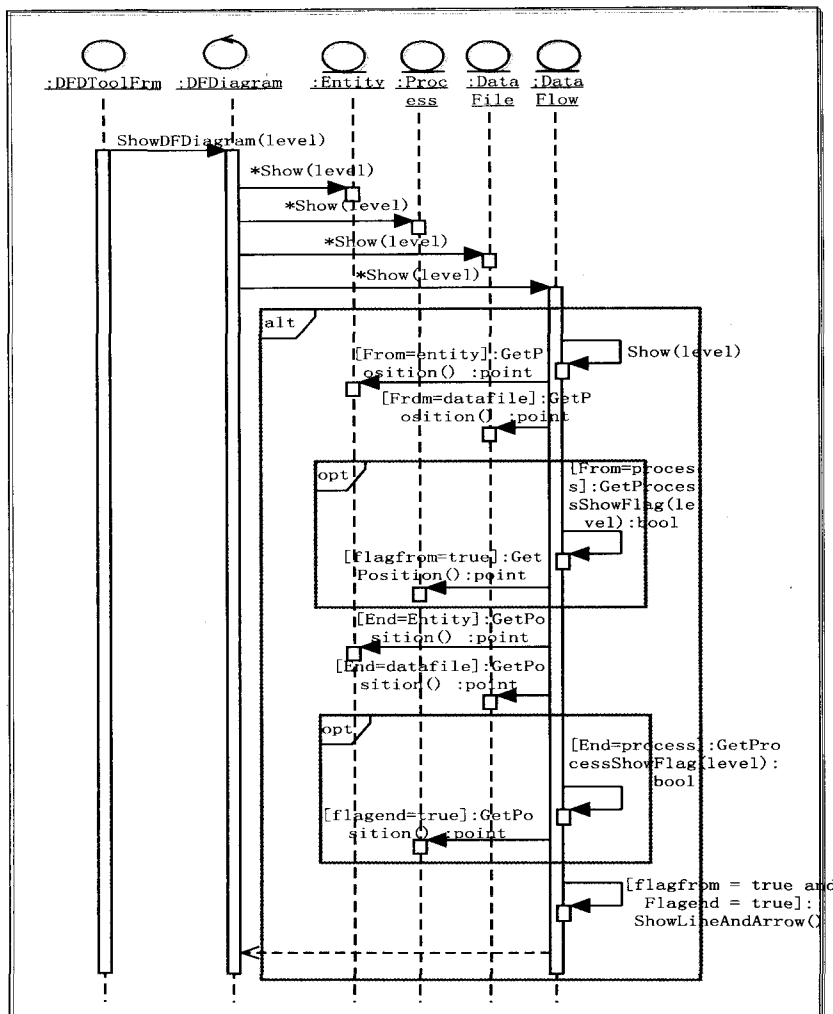


图4 数据流图显示功能的动态建模

6 结束语

数据流图可视化编辑工具能够将数据流图和数据字典有机结合,解决了现有画图软件无法记录数据字典的问题;工具可绘制多层数据流图,解决目前画图软件只能绘制单层数据流图的问题;工具依据有向图的特征及数据流图的特征,自动保持父子图的平衡和一致,解决现有画图软件无法检查数据流图正确性的问

题;工具自动生成数据字典功能解决了用户编辑数据字典时容易和数据流图不符的问题。

根据目前的研究成果,未来可进行数据流图映射为软件结构图并自动生成C语言代码的研究,可为嵌入式程序设计提供支持;并根据数据存储的数据组成和数据项的类型建立数据库文件,为管理信息的分析与设计提供支持。

参考文献:

- [1] 罗磊,曹顺良,王超,等.基于UML的生物实验室信息管理系统建模研究[J].计算机工程与设计,2009,30(3):689-692.
- [2] 秦晓,张衡.数据流图图形编辑工具的主要算法[J].计算机工程与应用,1994(4):22-23.
- [3] 董庆超,王智学,张爱辉,等.基于UML类图模型的一致性检查方法[J].计算机技术与发展,2008,18(10):85-88.
- [4] 方木云,戴小平.基于UML的集成化CASE平台的研究和实现[J].计算机技术与发展,2006,16(2):26-31.
- [5] 白桂梅.结构化与面向对象分析方法之间关系的研究[J].现代电子技术,2009(20):137-139.
- [6] Fernandes J M, Lilius J, Truscan D. Integration of DFDs into a UML-based model-driven engineering approach[J]. Software and Systems Modeling, 2006, 5(4):403-428.
- [7] 许家珩.软件工程-方法与实践[M].北京:电子工业出版社,2008:22-33.
- [8] Gao Xiaolei, Miao Huaikou, Liu Shaoying, et al. The Availability Semantics of Predicate Data Flow Diagram[C]//Lecture Notes in Computer Science. [s.l.]:[s.n.], 2004:970-977.
- [9] Liu Tong, Tang C S. Semantic specification and verification of data flow diagrams[J]. J. of Comput. Sci. & Technol., 1991, 6(1):21-31.
- [10] 钱乐秋,赵文耘.软件工程[M].北京:清华大学出版社,2009.
- [11] 王家华,徐皓冬.智能的数据流图辅助生成系统[J].东北大学学报,1998,18(1):40-43.
- [12] 袁涛,孔蕾蕾.统一建模语言[M].北京:清华大学出版社,2009:21-31.