钟元生 （南昌江西财经大学计算机系 330013）

An Object Oriented Method to Modeling Webapps: OOHDM

面向对象的Web建模技术OOHDM

摘要：随着WebApp的复杂度增加，越来越多的人认识到必须采取措施，在系统实现前加强建模工作。OOHDM就是一种较好的Web建模方法。本文以网上杂志的设计为例介绍之。

关键词：Web 信息系统 面向对象 模型

1 OOHDM概述

随着基于web的信息系统(web-based applications, WebApps)越来越普遍和复杂，人们越来越希望能用系统的建模理论来指导开发实践，以开发易维护与复用程度高的WebApp [1]。OOHDM [2] [3] [4](Object Oriented Hypermedia Design Method)是一种较成功的超媒体设计理论。

使用OOHDM，开发WebApp时可以采用熟悉的面向对象软件工程经验。OOHDM将WebApp看作是建立在对象模型之上的导航视图。OOHDM为导航及用户接口提供了相应的标准构件(construct)，OOHDM将导航对象(Navigation objects)看成是概念对象(conceptual objects)的视图，用导航上下文(Navigational Contexts)组织导航空间，导航设计独立于界面设计，采用清晰的规格说明来描述WebApp。

OOHDM将WebApp开发分为四步，即：概念设计、导航设计、抽象界面设计及实现，下面分别描述。

2 概念设计

OOHDM的概念模型设计，与普通面向对象的软件工程中类似，因此，这儿仅给出它的一些专用概念。

概念模型：OOHDM的概念模型包括类、关系与子系统。图2是某在线杂志的简单概念模型。其中有故事，故事可以是短文、翻译与访谈，访谈是问答式集合。每个故事有一个作者，而访谈也与一个人相关。为简便计算，模型强调了类行为的关系。

概念对象：概念模型涉及两类对象：

· 结点：导航模型的对象，由类导出。

· 动作：封装具体的行为，为WebApp提供计算支持，如数据库的存取或具体应用领域的计算方法。

观察：类的属性可能有多种类型，每种称为一个观察(perspective)，表示对真实世界中实体的一种观察方式。多个观察用符号“[p1, ... ,pn]”表示，pi为观察的实例变量，其中有"+"号为默认观察，所有实例中必须出现它，但不一定出现其他观察。例如，图2中短文Illustration属性可以是相片(photo)或录像(video)，相片为缺省观察。

关系：概念模型的“关系”表示WebApp领域的一些重要特征，在具体实现类之前，OOHDM一般不将关系隐藏在类属性中，而应详细说明。建立在OO环境的WebApp，常用类的“方法”实现“关系”的存取。

行为模型：用来描述WebApp的各种各样的计算活动，如对象数据库的动态查询、在线修改、基于启发式的搜索等等。概念模型所需的行为种类依赖于WepApp要求的目标特征。只需浏览功能的普通WebApp，其类只要链接行为。相反，复杂WebApp，如大公司信息系统的超媒体网络部分，则要在概念类中定义很多“方法”实现复杂行为。

3 导航设计

3.1 有关概念

导航模型：导航模型是建立在概念模型之上的视图。不同用户可以按照各自特征构建不同的导航模型，每个导航模型分别提供一个对概念模型的“主观”视图。

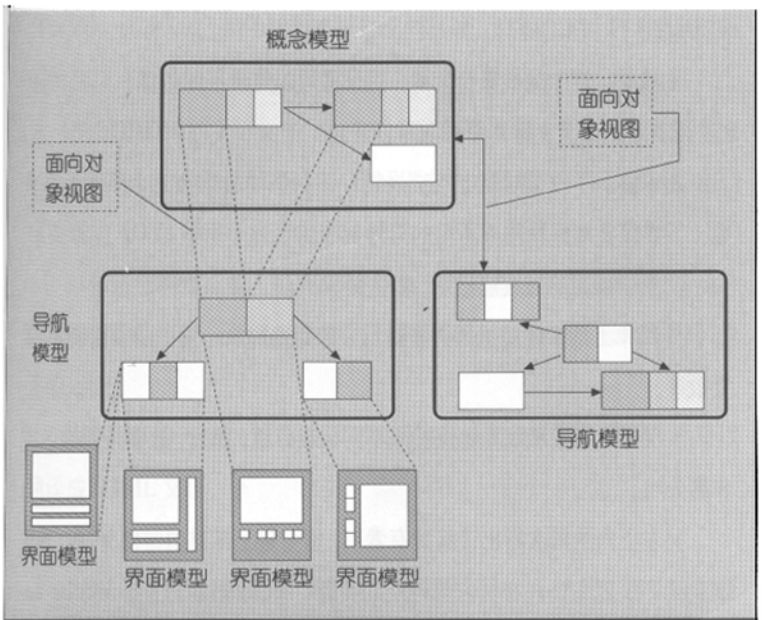

图1 概念对象、导航对象与界面对象之间的关系

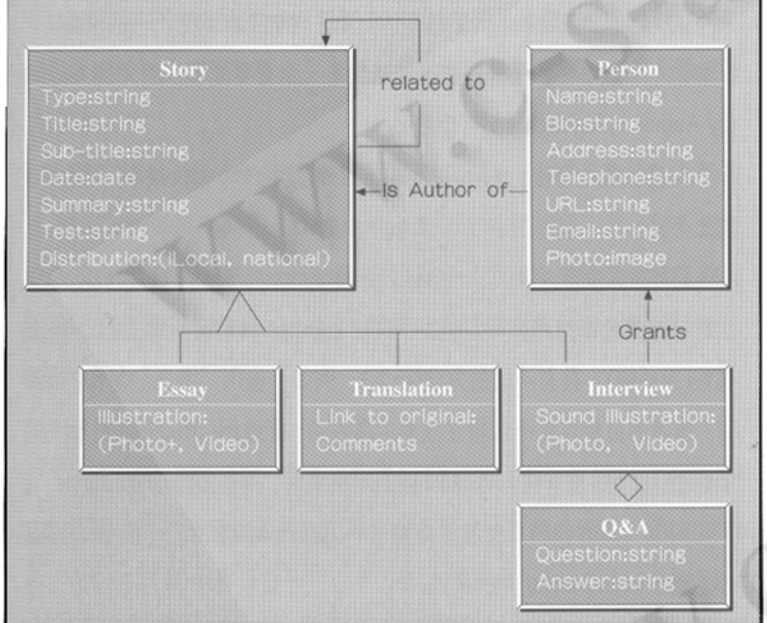

图2 一个在线杂志的概念模型

导航对象：即结点与链接，其属性可能是几个不同概念对象的属性"剪切"，可以有自己的行为，实现除浏览与导航以外的功能，如修改及计算。根据用户的特征与任务，可以用对象间的视图机制规定个性化导航。导航对象不被用户直接感觉，而经过界面对象来存取。

导航上下文：导航空间的对象的收藏集合，代表导航空间的结构。

导航转换：导航转换描述WebApp的动态性，规定用户导航时导航空间的改变方式。

3.2 设计导航类

OOHDM的预定义导航类包括：结点、链接与存取结构。

(1) 结点规划。OOHDM使用一个查询语言，将定义结点为相关的不同类的属性组合。其通用语法为：

NODE name [FROM className: varName] [INHERITS FROM nodeClass]

attr1: type1 [SELECT name1] [FROM class1:varName1, classj: varNamej WHERE Lexpression]

attr2: type2 [SELECT name2] .....

.....

attrn: typen [idem]

END

其中：?name是新建结点的类名；?className是概念类名；?nodeClass是超类名；?attr1是结点类的属性名；typei是属性i的类型；?namei是查询表达式的主题。expression是逻辑表达式，规定类的实例应满足的条件。

(2) 链接规划。链接定义为概念模型中关系的视图。其定义语法与结点定义类似。链接连接导航对象，可能是一对一或一对多。遍历链接的结果通过将导航语义定义为链接行为的结果来表示，或者通过与状态图相似的OO状态转换机制来表示。

(3) 存取结构。存取结构也定义为类，并且为超媒体应用提出可选的导航方式。

(4) 实例。考虑图2中的在线杂志。以定义一个story结点类为例说明之，设story结点类的属性包括作者姓名(author)及其自传(bio)和指向作者的链接锚(toAuthor)。按前述语法，定义如下：

NODE Story [FROM Story:St] [INHERITS FROM Person]

author: String [SELECT Name] [FROM Person:Pr WHERE Pr Is Author of St]

author_bio: String [SELECT Bio] [FROM Person:Pr WHERE Pr Is Author of St]

.....(其他从概念类Story中保留的属性)

toAuthor: Anchor (Is Author of)

END

注意概念模型中人名与自传是person类的属性；toAuthor属性的值，是用链接类Is Author of参数化的一个锚。当定义story结点类的界面显示时，可以将锚(toAnchor)作为可视按钮显示在作者名字(Author)上端。尽管author与toAuthor两个属性似乎有相同的行为，但是只有锚的行为响应界面事件(如点击)。图3为在线杂志的一个导航类规划实例。

3.3 设计导航上下文

导航上下文是结点、链接、上下文类以及其他嵌套的导航上下文的集合,它定义用户探索WebApp超媒体空间的方式。如果上下文的元素能变化为用户导航的序列,则这个上下文称为是动态的。如果允许建立与修改本对象,所有源自这些对象的上下文也是动态的。

(1) 定义导航上下文的常见方法

· 基于简单类：所有元素属于相同类C，并满足属性P，记为：context={e|p(e), e ∈ C}。

· 基于类组：所有元素是基于简单类的上下文。所有可能的值为有限可枚举的。例如，stories by type是一个基于简单类的上下文的上下文组，对于每个可能的类型（type）存在一个上下文，包括其"Type"属性等于type的故事。该上下文组可以表示为：Group={Context theme}，Contexttheme={P|P.type=type, P ∈ Story}

· 基于链接：链接的集合。例如，所有"张三"写的故事可以表示为：context={p| Is Author of(Bob Woodward, p),p ∈ Story}。

· 基于链接组：所有元素是基于链接的上下文。如上下文组Stories by Author可以表示为：

Group={Context Author},ContextAuthor={P|Is Author of(Author,p), p ∈ Story}

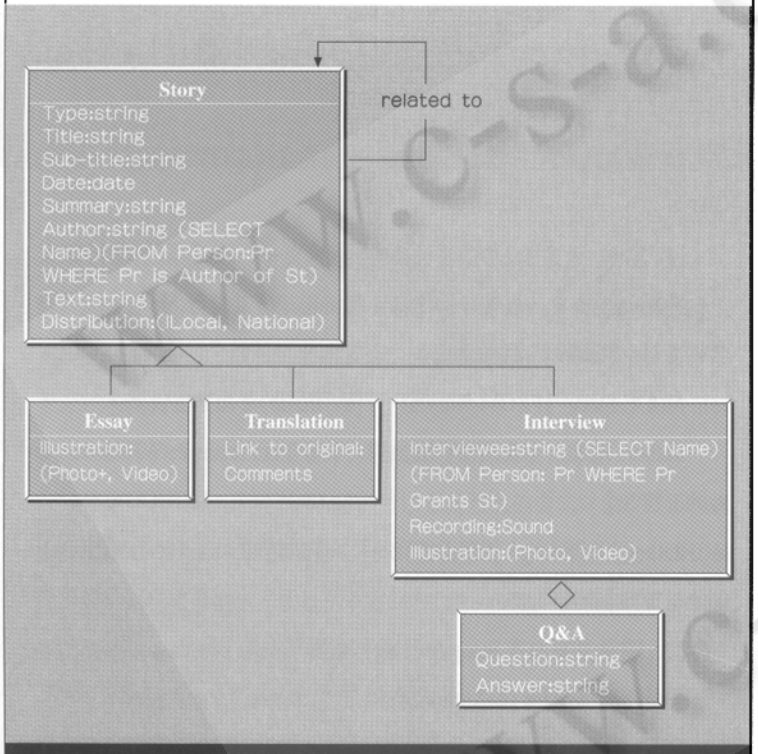

图3 在线杂志的一个导航类规划

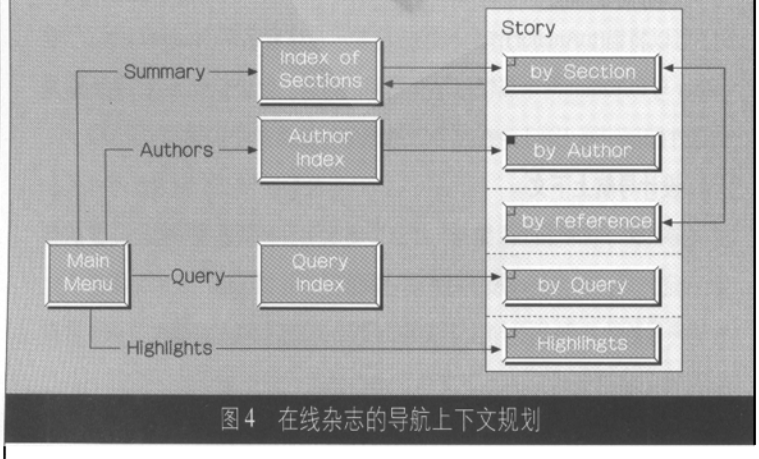

图4 在线杂志的导航上下文规划

· 枚举：逐个列举集合元素，且元素可以属于不同的类。

(2) 导航上下文基本符号

图4显示了一个导航上下文规划。该杂志的故事按几个不同标准分组。下面简要地解释图中所用的符号：

· 索引由粗的虚线盒指示，如主菜单。

· 简单上下文由虚线圆边框指示，如highlights，指出某期杂志的重要故事。

· 上下文组表示同类结点按照不同标准分组。例如，故事按照地区或作者分组。

· 左上角的黑盒指出该组拥有索引，索引是上下文的一部分。上下文本身还可规定在允许在其内部导航的类型：sequential(顺序),circular sequential(循环顺序),index(索引)，或index sequential(索引顺序)等。

· 组间虚线表示不允许直接从组中一个上下文转换到其他组的另一个上下文。例如，在看某地区的故事时，允许导航到本地区下一故事或同一作者的下一故事，但不允许直接导航到下一个重要故事。因为by Section /by Author与highlights间有虚线。

· 箭头表示链接，表示读者可以沿着链接到一个相关故事，到达相应的上下文。

(3) 结点装饰。结点用一个称之为Incontext特殊类集合装饰(见图5)，这种特殊类允许结点在特定的上下文中导航时，看起来不同，呈现不同的属性(包括锚)，具有不同的方法(行为)。例如，遍历"张三"写作的故事时，作者的自传可能不包括在故事的属性中，而遍历"Highlights"时则可能包括。图6给出一个InContext类的规格说明，用于装饰"Highlights"上下文内的"Story"结点。注意"前一个"(previous)与"下一个"(next)链接都是属于该InContext类的属性。

4 抽象界面设计

4.1 界面对象

抽象界面设计规定界面对象，界面对象起用户交互与导航对象中介的作用。界面对象将激活导航以及其他的应用功能，规定发生什么界面转换以及转换时间。它也指出哪些是不影响导航状态的本地界面转换，因此不需要存取web服务器。OOHDM用抽象数据视图(abstract data view, ADV)来描述超媒体应用的用户界面。ADV被视作界面对象。

4.2 抽象数据视图与抽象数据对象

ADVs是界面与状态的抽象，但不是实现。ADVs有一个状态与接口，接口可由消息控制。一个ADV由定义其感觉特性的属性(与嵌套的界面对象)的集合以及与它能操作的事件(如用户生成的事件)组成。属性值可以定义成常数，从而定义特殊的显示式样，如：位置、颜色与声音。保留变量

"PerceptionContext" 用于指出感觉空间的修改，如某给定时刻可感觉对象的集合。

在OOHDM的上下文中，导航对象如结点与索引充当抽象数据对象(Abstract Data Objects, ADOs)，ADOs不支持外部用户产生的事件。他们相关联的ADVs则用于规定用户显示。

从体系结构观点看，ADVs是ADOs的观察者，所以在界面与应用对象之间的通信协议应遵循观察者设计模式(Observer Design Pattern)[5]描述的规则。

在Web OO环境如Java中，可以很容易地实现ADV。ADVs也可以翻译成HTML文档。

4.3 ADV的抽象机制

ADV设计方法使用了不同的抽象与写作机制。一方面，可以通过低层ADVs的聚合或合成方法写作ADVs，因此允许用嵌套的可感觉对象来建立用户界面。例如，假定关于油画的应用中，油画的ADV能由图象、文字以及按钮三个ADVs合成（见图7）。另一方面，ADVs也可按泛化/特化层次组织，这种层次为定义界面对象的层次体系提供一个强大的框架(见图8)。

图8中AnchoredText是一个界面对象，将一个锚点集合加入到更通用的TextField对象中。Description是一个特化按钮，用于加入更个性化的行为（行为没有显示在图中）。当用一个环境支持某种特定种类的界面对象来实现WebApp时，可以用它们作为生成设计规格说明的基本ADVs。

概括来说，ADVs允许表达：

· 界面结构化方式，用聚合与泛化/特化作为抽象机制。ADVs表达实现界面的静态布局结构，定义导航对象与其他有用的界面对象(如菜单条、按钮与菜单)的界面显示。

· 它们与导航对象静态相关的方式。用配置图作为图形工具表达这些关系。

· 响应外部事件时它们如何行动，特别是如何触发导航以及当用户与应用交互的界面转换。为表达典型的多媒体数据同步问题，在ADV图增加了结构与行为两个嵌套，以及一个象Petri网一样的符号。总之，ADVs规定界面的组织与行为，其实际的物理显示或属性以及ADV在屏幕上的布局，则在实现阶段完成。

5 结束语

OOHDM中，概念、导航与界面三个模型的相对独立与互相分离，为提高系统可维护性及复用性打下了良好的基础。因为导航与抽象界面设计使用的建模构造是很相似的，所以能获得在两个活动之间的无缝变换，并方便对导航与抽象界面模型的增量构建。

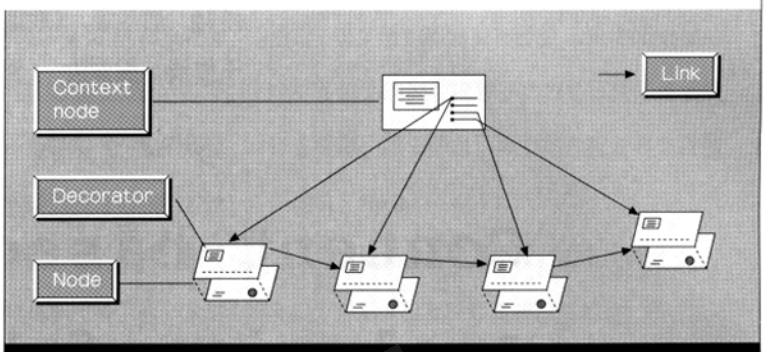

图5 基本结点和装饰器一起构建上下文中的一个结点

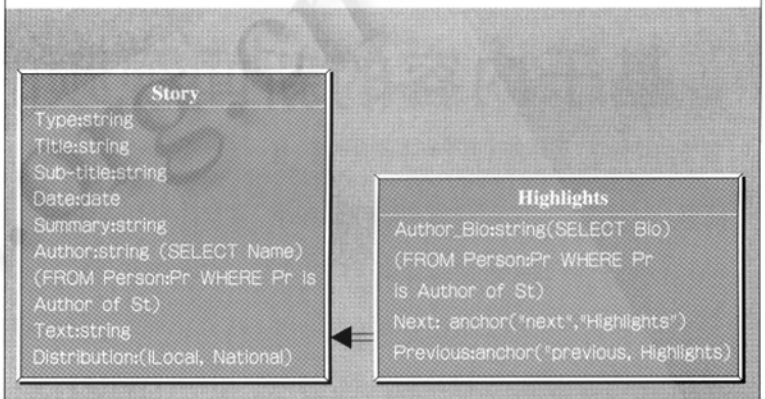

图6 InContext类规格说明举例

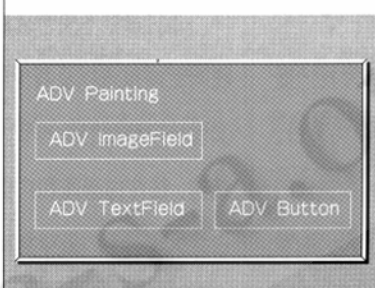

图7 ADV使用聚合

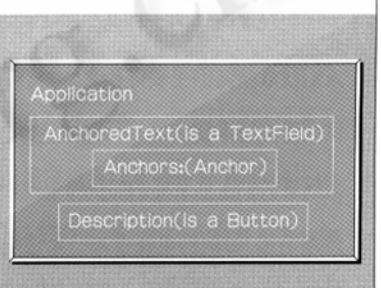

图8 ADV使用继承

参 考 文 献

1 Pressman, R. S., What a tangled Web we weave, IEEE Software, Vol.17 Issue: 1, Jan.-Feb. 2000: p18 -21。

2 Daniel Schwabe, Gustavo Rossi An object oriented approach to web-based application design, Theory and Practice of Object Systems 4(4), 1998. Wiley and Sons, New York, ISSN 1074-3224), p1-34 。

3 Daniel Schwabe, Patricia Vilain: The OOHDM notation, http://sol.info.unlp.edu.ar/notacaoOOHDM/, 2000年10月。