

面向语义 Web 的软件模式的描述

冯新亚, 刘奎, 钱萌, 程一飞
(安庆师范学院 计算机系, 安徽 安庆 246011)

摘要: 软件模式用于捕捉开发面向对象软件的经验, 可以被复用。文中分析了应用 XML, RDF(S) 和 DAML + OIL 三者描述软件模式的区别, 并提出应用 DAML + OIL 来描述软件模式以及软件模式之间的关系, 使软件模式的描述具有精确的语义, 从而使计算机可以理解 Web 上的软件模式, 为以后研究基于语义 Web 和具有推理能力的搜索奠定了基础。

关键词: 软件模式; 语义 Web; XML; 资源描述框架; DAML + OIL

中图分类号: TP311.5

文献标识码: A

文章编号: 1673-629X(2007)01-0079-03

Representation of Software Patterns Using Semantic Web - Oriented

FENG Xin-ya, LIU Kui, QIAN Meng, CHENG Yi-fei
(Department of Computer Science, Anqing Teachers College, Anqing 246011, China)

Abstract: Software patterns are an attempt to capture expertise in building object-oriented software, and are software reusability. Analyses the difference of means for representing software patterns by XML, RDF(S) and DAML + OIL. And present that make use of DAML + OIL to describe software patterns and their relation, so patterns has more semantic, and for search based on semantic.

Key words: software pattern; semantic Web; XML; RDF; DAML + OIL

0 引言

随着软件技术的不断发展, 人们希望软件的开发过程中可以复用一些已有的部件。这个领域内近些年提出的一个重要思想是软件模式。这个思想认为在系统设计这一层次上, 软件开发可以抽象成一种模式, 模式描述了系统所面临的问题和解决此问题的方案, 并可以复用^[1,2]。因此, 基于软件模式的软件开发可有效地提高软件开发的效率。互联网的出现, 尤其第三代网络——语义 Web 的出现, 使人们可以充分利用网上的海量资源, 并希望利用互联网的优势, 把已有的软件模式集中管理, 并把它们放在互联网上供更多的人使用。同时, 大家还可以上传自己挖掘出来的软件模式, 这样就可以形成一个庞大的模式库。但是由于目前软件模式的描述缺乏丰富的语义, 人们很难通过网络找到自己所需要的软件模式, 更难实现软件模式的自动选取。所以, 为了更好地自动地复用现有的软件模式, 如何描述软件模式就成为一个研究的重难点。

Web 的创始人 Tim Berners Lee 于 1998 年提出了语义 Web 的概念^[3]。根据 Tim Berners Lee 和 W3C 的定义, 语义 Web 是建立在 RDF 与其它定义的标准基础之上, 对 Web 上的数据所进行的一种抽象表示。语义 Web 是一个

包含 7 个层次的网(体系结构如图 1 所示), 它包含了文档和文档的一部分, 描述了事务间的关系, 且包含语义信息, 以利于使现有的 Web 信息具有计算机可理解的语义, 从而可对 Web 上的资源实现有效的访问和精确的搜索。

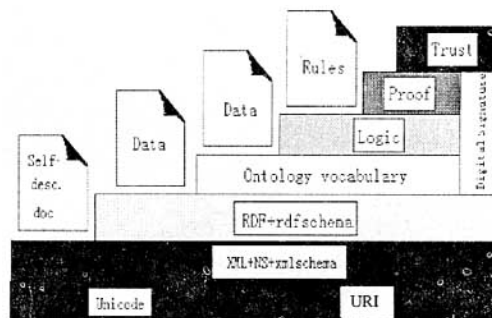


图 1 语义 Web 结构

文中分析了基于 XML, RDF(S), 和 DAML + OIL 来描述软件模式之间的区别, 提出了采用 DAML + OIL 的方法来描述软件模式以及模式之间的关系^[4], 使软件模式的描述具有精确的语义, 并且为以后研究基于语义的和具有推理能力的软件模式搜索奠定了基础^[2]。

1 基于 XML 的软件模式的描述

XML 是 W3C 于 1998 发布的, 是一种专门为 Internet 所设计的标记语言, 是国际标准化组织在 SGML 基础上制定的一系列规则的集合。这些规则说明了如何定义标记, 这些标记又可以将文档分为多个部分和子部分。

收稿日期: 2006-04-27

基金项目: 安徽省教育厅自然科学研究项目(2006kj155c); 安徽省高校青年教师资助计划项目(2006JQL207)

作者简介: 冯新亚(1961-), 男, 安徽安庆人, 讲师, 研究方向为计算机体系结构。

XML 是由嵌套的标记元素构成的自描述标记文本,它是数据表示和交换的主要标准。由于 XML 的自描述性和可扩展性,它不仅能够有效描述传统的结构化数据,更为重要的是:它也是描述半结构化数据的理想工具。XML 允许用户创建文档类型定义 (DTD) 或 XML 模式 (XML Schema), 以实现文档有效性的检查和验证。XML 最大的特点是格式和文档的分离,具有很好的结构性。用户可以定义自己的标签,让 XML 进行自解释。

由于 XML 这些特点,所以运用 XML 描述软件模式具有一定的优势^[4,5]。

(1)清晰的结构层次。比如用刻面分类法,从各个方面来描述软件模式的时候,使用 XML 描述软件模式的各个方面,属性与值的层次结构非常清晰。

(2)良好的扩展性。由于 XML 的标签不是固定的,用户可以自己定义。因此采用 XML 描述软件模式的各个方面的时候,具有很好的扩展性。

(3)很好的交互性。XML 作为一种标准交换语言,不同的人使用的标签可以不同,使用时只要就标签的语义达成共识就可以实现交换。

软件模式的描述一直是软件工程领域的一个研究重点。根据文献[1],软件模式主要由 4 个部分组成:模式名称、问题域、解决方案和效果域。这 4 部分用模式名称与类型、目的、别名、动机、适用性、结构、参与者、协作图、效果、实现、代码示例、已知应用和相关模式 13 个元素组成的统一模板来描述软件模式。显然,这是一种半形式化的描述方法,其描述结果是一种半结构化的数据,符合 XML 的特点。所以文献[2]采用 XML 描述软件模式,并给出了描述软件模式的 E-R 图(如图 2 所示)。

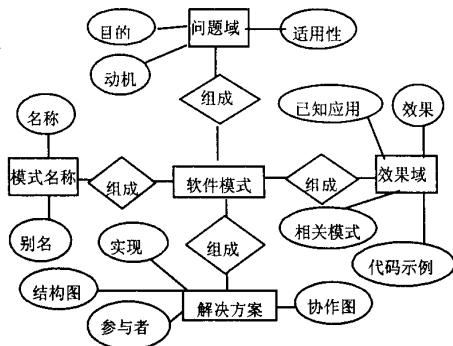


图 2 描述模式的 E-R 图

软件模式 Proxy 的 XML 的描述如下:

```
<? xml version="1.0"? >
<! DOCTYPE SoftWare patternBase c: \spb \xml \softwarepat-
tern. dtd>
<SoftWare_ Pattern>
  <SoftWare_ Pattern_ Name>
    <Name>Proxy</Name>
    <Also_ Known_ As>Surrogate</Also_ Known_ As>
  </SoftWare_ Pattern_ Name>
  <Problem>
```

```
<Intent>Provide a proxy for another object to control access
to it</Intent>
  <Motivate>.....</Motivate>
  <Applicability>.....</Applicability>
</Problem>
.....
<Solve_ Way>
  .....
</Solve_ Way>
</SoftWare_ Pattern>
```

2 基于 RDF(S)的软件模式的描述

XML 所描述的树型结构虽然具有良好的结构 (Well-Formed),但是缺乏丰富的语义,所以仅仅用 XML 描述显然不能实现软件模式的智能搜索。

资源描述框架(RDF)是一个能对结构化的元数据进行编码、交换及再利用的体系框架。RDF 以 XML 语法为基础,提供了描述网络资源以及资源之间关系的模型与语法格式^[6]。RDF 包含了 3 种对象模型:资源、属性和声明。RDF 对信息的描述是由一系列三元组所组成的,每个三元组的结构为(资源,属性,值)。RDF 利用它的三元组表达丰富的语义,是计算机可以理解的,这样就可以支持网络资源的自动处理。

RDFS 是在 RDF 的基础上建立的类型系统,支持从客观世界到抽象世界的映射。用户可以基于 RDFS 定义自己的 Schema,通常属于一个特定的领域。

由于 RDF(S)的种种性质,因此应用 RDF(S)来描述 Web 上的软件模式将具有很多优势^[4,7,8]:

(1)与 XML 相比,更丰富、更全面地描述了软件模式的各种属性和属性的约束。

(2)方便描述软件模式之间的关系和约束条件,这可以提高 Web 搜索的查全率和查准率。

(3)提供了比 XML 的描述更丰富的语义。

下面应用 RDF(S)来描述软件模式 pattern1, pattern2, pattern3 之间的关系。假如 pattern2 和 pattern3 是 pattern1 的子类。关系如图 3 所示,其中 $s = \text{rdfs:subClassOf}$, $t = \text{rdf:type}$ 。

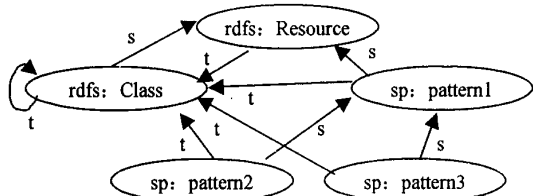


图 3 软件模式之间的关系图

用 RDF(S)描述图 3 如下:

```
<rdf:RDF xml:lang="en"
  xmlns:rdf = http://www.w3.org/1999/02/22-rdf-syntax-
  ns#
  xmlns:rdfs = http://www.w3.org/2000/01/rdf-schema#>
```

```

<rdf:Description ID="pattern1">
  <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/
2000/01/rdf-schema#Resource"/>
</rdf:Description>
<rdf:Description ID="pattern2">
  <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
  <rdfs:subClassOf rdf:resource="#pattern1"/>
</rdf:Description>
<rdf:Description ID="pattern3">
  <rdf:type resource="http://www.w3.org/2000/01/rdf-
schema#Class"/>
  <rdfs:subClassOf rdf:resource="#pattern1"/>
</rdf:Description>
</rdf:RDF>

```

3 基于 DAML + OIL 的软件模式的描述

RDF(S)在表达能力和逻辑严格性方面的不足导致了 DAML + OIL 的出现。DAML + OIL 由早期的 DAML 本体语言和一种基于描述逻辑的本体推理层 OIL 合并而成。DAML + OIL 提供了很多词汇使其具有相当强的表达能力,它不仅继承了 RDF/RDFS 的表达能力,还从描述逻辑(Description Logic)中借鉴了很多表达方式,能够提供足够的约束条件使得其描述客观领域知识的能力非常强大。同时,它还遵循面向对象的思想,按类和属性的形式描述领域知识所包含的结构。

鉴于 DAML + OIL 的特点,应用 DAML + OIL 来描述软件模式,将会更精确地描述软件模式的语义和模式间的关系^[4,9]。

(1) DAML + OIL 系统中的领域与范围允许多范围的定义,这样用户在描述一个新类型的描述时,可以灵活地定义类型。

(2) 易于交互。DAML + OIL 定义了 Equivalentto 和 SamePropertyAs 属性,这样当两个不同模式库中的同功能的模式具有不同名称时,可以使用属性 SamePropertyAs 实现交互。

例如两个软件模式库中代理模式的功能相同,但是模式名不同,分别为 Proxy 和 Surrogate,则在交互的时候,可以描述如下:

```

<rdf:Description about="#Proxy">
  <daml:samePropertyAs rdf:resource="http://software-pat-
tern.org/Surrogate"/>
</rdf:Description>

```

(3) DAML + OIL 提供了丰富的约束。软件模式应用在特定的环境(context)中,所以要完整地描述软件模式,应包含大量的约束条件。在应用 DAML + OIL 描述软件模式时,可以用 onProperty 和 hasValue 标签来描述软件模

式的约束。

(4) DAML + OIL 提供了丰富的类的关系。这样比较方便地描述软件模式之间地关系。

例如,应用属性 disjoint 来描述没有共同实例的两个软件模式。一般,软件模式根据它的应用层次不同,分为:体系结构模式(architecture pattern)、设计模式(design pattern)和习惯术语(itom),它们之间没有公共的实例,可以应用 DAML + OIL 描述如下:

```

<daml:Class rdf:ID="software-pattern">
  <rdfs:label> software-pattern </rdfs:label>
  <rdfs:comment> patterns are divided to architecture patterns,
design patterns and itoms. </rdfs:comment>
<daml:disjointUnionOf parseType="daml:collection">
  <daml:Class rdf:ID="architecture-pattern">
    <rdfs:label> architecture-pattern </rdfs:label>
    <rdfs:comment> these patterns for building software ar-
chitecture </rdfs:comment>
  </daml:Class>
  <daml:Class rdf:ID="design-pattern">
    <rdfs:label> design-pattern </rdfs:label>
    <rdfs:comment> these patterns for designing software
system </rdfs:comment>
  </daml:Class>
  <daml:Class rdf:ID="itom">
    <rdfs:label> itom </rdfs:label>
    <rdfs:comment> these patterns for implementing soft-
ware system </rdfs:comment>
  </daml:Class>
</daml:disjointUnionOf>
</daml:Class>

```

4 结束语

语义 Web 的出现与发展为在语义级上进行信息知识的表达提供了一种先进的手段。语义 Web 赋予 Web 上信息更丰富的语义,并使计算机能够理解,实现 Web 服务的智能化。XML 的出现,使得 Web 上的资源(如软件模式)的描述变得更加结构化,但是也看到基于 XML 的描述缺乏更准确的语义,也不方便描述软件模式之间的关系。文中提出了采用 DAML + OIL 来描述 Web 上的软件模式,使 Web 上软件模式具有良好的结构和完备的语义,从而有利于实现软件模式的智能化处理,也为以后进一步研究与实现具有推理能力的模式自动搜索引擎提供了基础。

参考文献:

- [1] Gamma E, Helm R, Johnson R, et al. Design Patterns: Elements of Reusable Object-Oriented Software[M]. 北京:机械工业出版社,2002.
- [2] 刘奎,袁兆山,陈刚,等.基于 XML 的设计模式描述及

(下转第 84 页)

```

function()
{
    输入;
    T //T 为一棵树,即初始查询树
    M=x; //设置 M 的大小
    While(N>M)
    {
        Select(M); //分离出一棵小树,返回小树的结点
        DP(); //对小树使用 DP 算法,返回以树根为序的连接
        顺序
        Merge(); //合并结点
        N=N-M+1; //修改当前查询树结点个数
    }
    DP(); //对最后的查询树使用 DP 算法
}

```

●算法分析:

本算法利用了局部优推导出全局较优的近似算法思想。当查询树结点数增大到一定程度,对整棵树使用动态规划算法会导致状态数目过多、搜索空间过大、花费的时间代价过多。当使用笔者提出的算法,结点数目会在循环的过程中以 M 递减,而每次参与运算的仅仅是 M 个结点,有效地缩减了搜索空间。同时每个局部的较优也保证了全局的较优。

假设使用 DP 算法对一个结点数为 M 的树进行连接顺序选择,时间复杂度为 $T(M)$ 。每循环一次查询树结点减少 M 个,故使用本算法总时间复杂度为: $T(M) * \frac{N}{M}$ 。在这里忽略了分离一个小树的算法的时间复杂度,这是合理的,因为该算法时间复杂度仅在线性级别。与普通算法的复杂度相比,分树连接的动态规划在时空上得到了很大的改进。虽然不能保证找到最优的解,但它缩减了搜索空间,效率很高。

●对 M 大小的讨论:

本算法总时间复杂度为: $T(M) * \frac{N}{M}$, 随着 M 的增大, $T(M)$ 增大, $\frac{N}{M}$ 减小。当 M 增大到 N 或者减小到 1, 算法都将退化为普通 DP 算法。由此看出, 选择一个合适大小的 M 对本算法至关重要。笔者经过粗略的分析, M 在 3 ~ 10 之间是比较适宜的, 且根据查询树形状的不同, M 有不

同的适用值。

●算法适用条件:

当 N 较小时, 使用前述的 DPP 或者 DPAP 算法在较短的时间内即可得到较优的连接顺序。若使用本算法, 反而将问题复杂化。

当 N 较大时, 前述基本算法将不再适用。利用本算法则可以在较短的时间内得到一个较优的解。

4 结束语

通过对现有的选择连接算法的比较分析, 针对其不足之处, 笔者提出了一种优化策略, 并给出新算法的实现过程及性能分析。与普通算法相比, 该算法在较大规模的 XML 查询中能有效地缩减搜索空间, 提高效率。

参考文献:

- [1] Garcia-Molina H, Ullman J D, Widom J. Databases System Implementation[M]. 北京: 机械工业出版社, 2002.
- [2] Wu Y, Patel J M, Jagadish H V. Structural Join Order Selection for XML Query Optimization[C]//In: Casati F. Proceedings of the 19th IEEE ICDE International Conference on Data Engineering. Los Alamitos, Bangalore, India: IEEE Computer Society, 2003: 443-454.
- [3] Amer-Yahia S, Cho S, Lakeshmanan L, et al. Minimization of Tree Pattern Queries[C]//In: Proc. of the 2001 ACM SIGMOD Conf. on Management of Data. Santa Barbara, California, USA: [s. n.], 2001: 21-24.
- [4] 万常选. XML 数据库技术[M]. 北京: 清华大学出版社, 2005.
- [5] Al-Khalifa S, Jagadish H V, Koudas N, et al. Structural Joins: A Primitive for Efficient XML Query Pattern Matching[C]//In: Proceedings of the 18th International Conference on Data Engineering. Los Alamitos: IEEE Press, 2002: 141-152.
- [6] Lee Y K, Yoo S J, Yoon K. Index structure for structured documents[C]//In: Proceeding of the 1st ACM Int'l Conf. on Digital Libraries. New York: ACM Press, 1996: 91-99.
- [7] 刘云生, 伍慧敏. XQuery 查询优化中结构连接顺序选择算法[J]. 计算机应用研究, 2005(7): 87-89.

(上接第 81 页)

- 其存储系统[J]. 合肥工业大学学报, 2005, 28(6): 581-584.
- [3] Berners-Lee T, Hendler J, Lassila O. the Semantic Web[J]. Scientific American, 2001, 284(5): 34-43.
 - [4] 李涛, 董红斌, 彭煜伟. 基于语义 web 的组件描述方法[J]. 计算机工程, 2004, 30(7): 39-40.
 - [5] Fallside D C. XML Schema Part 0: Primer[EB/OL]. 2004-06-05. <http://www.w3.org/TR/2001/REC-xml-schema-0-20010502>.
 - [6] 张维明. 语义信息模型及应用[M]. 北京: 电子工业出版社,

2002: 148-172.

- [7] Hjelm J. Creating the Semantic Web with RDF[M]. [s. l.]: Wiley Computer Publishing, 2001.
- [8] Lassila O, Swick R. Resource Description Framework Model and Syntax Specification[EB/OL]. 1999-02-22. W3C Recommendation, <http://www.w3c.org/TR/1999/REC-rdf-syntax>.
- [9] Horrocks I. DAML+OIL: a Description Logic for the semantic web[J]. IEEE Bulletin of the Technical Committee on Data Engineering, 2002, 25(1): 4-9.