中 文 信 息 学报
第 14 卷 第 4 期 JOURNAL OF CHINESE INFORMATION PROCESSING Vol. 14 No. 4

面向置标文档的文档转换技术研究①

李景春　武港山　王强　张福炎

（南京大学软件新技术国家重点实验室　南京　210093）

（南京大学计算机科学与技术系　南京　210093）

摘要：文档系统间的转换是文档内容共享和协作的必然途径，转换根据不同应用目的包括失真，不失真和增值三种方式。置标文档是用标签(Tag)进行文档结构描述的文档。本文介绍了一种面向置标文档的文档转换增值技术，给出了一种文档转换描述语言，用户可以利用它来定义转换信息从而实现文档间复杂的转换。

关键词：文档转换；失真；增值；置标文档

中图分类号：TP391.12

Research on Markup Document-oriented Document Transformation Technology

LI Jing-chun　WU Gang-shan　WANG Qiang　ZHANG Fu-yan

(State Key Laboratory for Novell Software Technology　Nanjing　210093)

(Department of Computer Science and Technology, Nanjing University　Nanjing　210093)

E-mail: ljc@nju.edu.cn

Abstract: Document transformation among different document system is a necessary approach to content sharing and cooperation. Based on the application background, the transformation can be divided into three categories: distortion, non-distortion and increment. Markup document uses the tag to describe the structure. This paper introduces technology of markup document-oriented document transformation, and presents a document transformation describe language which can be used for transformation information definition, with this definition, user can accomplish complex transformation between different document.

Keywords: document transformation; distortion; increment; markup document

收稿日期：2000－01－03；修改稿收到日期：2000－05－08

基金项目：江苏省应用基础研究项目(BJ95006)

作者李景春，男，1973 年生，博士研究生，主要研究领域为多媒体文档及相关技术. 武港山，1968 年生，博士，副教授，主要研究领域为多媒体文档及相关技术. 王强，1976 年生，硕士研究生，主要研究领域为置标语言及相关技术. 张福炎，1939 年生，教授，博士生导师，主要研究领域为计算机图形学和 CAD，多媒体应用技术.

一、引言

由于应用背景的不同，文档的组织方式和表现方式差别很大，所以不同文档系统间的信息交换是文档内容共享和协作的必然途径[4,6]，如一个文档系统可能需要通过 Internet 从用户或其他文档系统中获得文档内容，建立自己的文档库，同时一般浏览器或其他文档系统需要向该系统请求文档信息时，也需要把文档库中内容以适当的格式提供出来。

文档不同描述形式之间的转换研究在商品化软件中时常可以见到，如 Microsoft 的 Word 文档编辑软件中就支持多种格式之间的相互转换。信息转换根据不同目的包括失真，不失真和增值三种方式。相对于文档信息的转换而言，失真方式是指文档目标描述形式的能力有限，无法反映源文档所描述所有内容，如：带格式的文档转换成纯文本文档，格式信息有可能丢失。增值转换相对于文档转换而言是指通过对文档内容的分析，增加了文档内容的描述信息。如文献[3,6]中的文本格式到超文本格式转换都属于增值转换。置标文档是文档的一大分支，它使用标签来标志文档的结构信息，如 HTML、SGML、XML 文档[1,2]。本文主要针对置标文档讨论基于用户定义转换信息的文档间的增值转换问题。

二、文档转换系统结构

文档转换系统的功能是将置标文档同其它描述形式下的文档进行相互转换。转换系统主要由两大功能模块构成：文档分析器和文档转换器。其结构如图 1 所示。

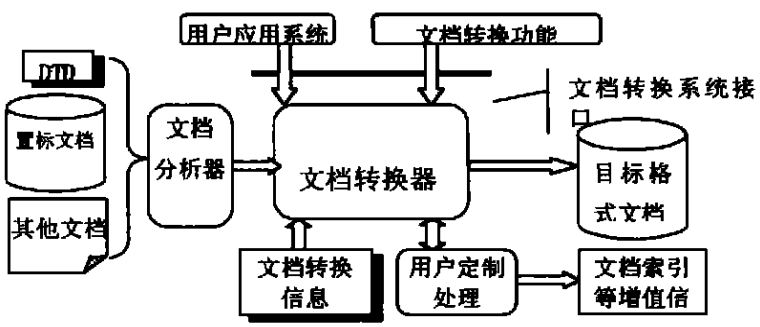

图 1　文档转换系统结构框图

2.1　文档分析器

文档分析器用于对输入的文档进行简单的结构分析，主要面向置标语言描述的文档（SGML 或 XML 文档），它将文档按置标的定义，进行分割，并将相应的信息传送给转换器去处理。对于非置标语言文档，则采用不同的功能模块对其文档内容结构进行分析，以便转换。

面向置标语言的文档分析器实现方法有两种，一种是依据文档的类型定义（DTD），对置标文档进行全面解析。这种方式中，文档分析器需要分析文档类型定义内容，并对文档的正确性进行检查。另一种是直接按照置标语言的语法分析文档的内容，无视文档类型定义的内容，直接根据语法对文档的内容进行分割。前一种方法虽然实现比较麻烦，但可以保证文档能够正确的切割，因为一般的置标文档，在缺乏文档类型定义时存在二义性。但这种情况在 XML 文档中不会出现，XML 规范中不允许出现省略结束标志的情况。

对于不同的输入文档，文档分析器必须采用不同的文档分析方法。特别是对于纯文本文档，分析器可以采取基于模式的分析的方法，如中文文档中的章、节等都是比较规整的，因而可

以通过简单规则来分割文档内容。也可以采用用户参与交互的方式，这种方式比较简单，但效率低下。

2.2 文档转换器

文档转换器的功能是根据用户指定的转换情报，将文档分析器传送过来的文档解析信息转换成目标文档的格式。文档转换首先要分析文档转换情报的内容，并形成内部可执行的格式，然后才能对文档内容进行转换处理。

经过文档分析器处理后，输入的文档可以简单地描述为置标文档格式：

〈开始标志〉被标志的正文部分〈结束标志〉

这种结构是可以嵌套描述的，文档的转换处理就是针对这三个部分进行处理，即：

·对开始标志的处理

·对正文部分的处理

·对结束标志的处理

文档的转换处理可以有多种输出内容，对于没有区分逻辑结构、布局结构等结构化信息的文档，在文档转换时，必须增加额外的处理模块自动生成或分类这些结构信息。

文档转换系统中支持用户定制的处理模块功能，以便允许用户生成一些特殊需要的信息。如根据用户提出的关键字信息来生成整个文档库的检索信息，根据一般布局结构在文档转换结束后，统一自动生成布局结构。

三、文档转换描述语言

文档转换信息是用户用文档转换描述语言创作的文档转换处理的描述信息，文档转换信息由表 1 所列的 5 个部分组成，而各个部分都是用文档转换描述语言记录的文档转换语句。文档转换语句分为定义语句和处理语句两大类，前者用于[DOCONV] 和[ENTITY] 分区，后者用于[COMMON] [ELEMENT] [FINAL] 分区。

表 1 文档转换信息的组成

分区名	功 能	省 略
[DOCONV]	定义文档转换的基本信息，如：描述语言版本号等	不可
[ENTITY]	定义文档转换信息中使用的文件名，或对源文档内容进行文字替换的信息	可
[COMMON]	文档转换处理前的共同处理信息，定义局部变量等。	可
[ELEMENT]	文档转换中各个部分对应的处理信息	不可
[FINAL]	文档转换处理结束时，各种善后处理过程	可

3.1 定义语句

定义语句可以定义转换输出必须的文档以及源文档中文字的自动转换部分。它的定义语法为：

1. 输出文档定义语句

filedef〈文档参照名〉:〈选择项〉〈文档名〉;

其中：选择项的定义为：

- FIX：定义的文档名可以直接使用，是完整的文档名。
- AUTO：文档名是变换过程中自动生成的，一般作为临时文档使用。
- API：文档名由用户通过交互手段定义，这一种在自动转换过程中不太方便，因而，可用于其它文档转换环境。

● DIR:文档转换过程中需要使用的目录名。

2. 字符串转换语句

strdef〈原字符串〉:〈目标字符串〉;

3.2 处理语句

处理语句的功能是对经过文档分析器解析的源文档内容,进行相应的处理,因为经过解析后,源文档内容被分割成许多子块,以置标语言语法的形式,分别输入开始置标、中间文字和结束置标三个部分。在转换定义信息中,可以对输入的不同部分定义相应的处理。一种置标可以对应多个处理语句。处理语句的定义语法为:

功能名 [处理对象] [/[开始置标对应的处理信息]
[/[置标中间文字对应的处理信息]
[/[结束置标对应的处理信息]]]] ;

其中:功能名和处理对象的定义如表 2 所示。

表 2 处理语句中功能名和处理对象定义

功能名	处理对象	说 明
PUT	文档名或 文档参照名	向目标文档中输出处理信息。
ORDER	无	对转换过程中定义的变量进行操作,可进行一般表达式的运算。
NOP	无	无任何动作。
RUN	文件名以及 执行参数	执行用户指定的执行文件。可完成用户定制的功能。
CALL	文件名以及 执行参数	调用用户指定执行文件中的接口。可完成用户定制的功能。
FILE	文档名或 文档参照名	对目标文档进行各种操作。如:拷贝、移动以及删除等。

处理语句的三个分割符“/”分别表示对开始置标、正文部分和结束置标的处理。由于[COMMON] 和[FINAL] 分区是在文档转换之前/后执行的处理,所以没有置标信息,系统缺省执行第一个分割符后的转换信息。

每个处理信息中可以定义要处理的常量,给变量赋值或进行判断或循环的控制等。转换系统定义了一些系统常量用于处理上的方便,如:置标名、文档内容字符串以及属性名和属性值变量等,用户也可以定义变量参与转换过程的处理。

转换过程中还可以执行用户自定义的处理模块,RUN 和 CALL 两项操作就是实现这样的功能,RUN 负责可执行文件,CALL 负责动态连接库类的文件。

3.3 文档转换例

由于文档转换的具体处理可以由用户通过转换信息来定义,且转换信息定义功能非常灵活,因此可以完成比较复杂的转换要求。下面举一个文档转换的例子。

下图是两个置标语言描述的文档的结构图。

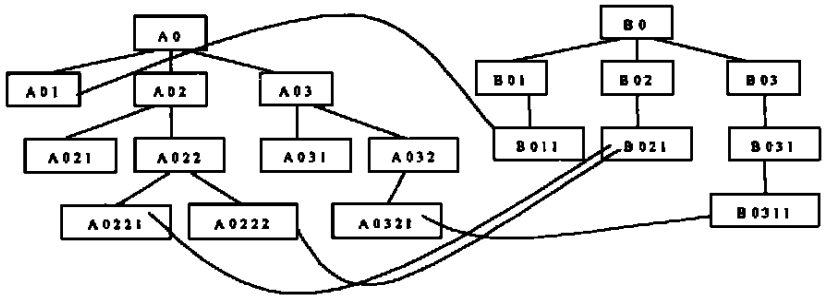

图 2　文档转换例

图中的箭头表示两个文档转换是的相互对应的内容, 其余内容不用转换。转换信息的为:

文档 1→文档 2

```
[DOCONV]
        ver = 1.0
[ENTITY]
filedef DOC: FIX  ". \output-doc.odm"
[ELEMENT]
〈A0〉
        put DOC    /"〈B0〉"
                   /
                   /"〈/B0〉";
〈A01〉
        put DOC    /"〈B01〉〈B011〉"
                   /$TEXT
                   /"〈/B011〉〈/B0〉";
〈A02〉
        nop;
〈A021〉
nop;
〈A022〉
nop;
〈A0221〉
        put DOC    /"〈B02〉〈B021〉"
                   /$TEXT
                   /;
〈A0222〉
        put DOC    /
                   /$TEXT
                   /"〈/B021〉〈/B02〉";
〈A0321〉
        put DOC    /"〈B03〉〈B031〉〈B0311〉"
                   /$TEXT
              /"〈/B0311〉〈/B031〉〈/B03〉";
```

文档 2→文档 1

```
[DOCONV]
ver = 1.0
[ENTITY]
filedef DOC: FIX  ". \output_doc.xml"

[ELEMENT]
〈B0〉
     put DOC   /"〈A0〉"
               /
               /"〈/A0〉";
〈B011〉
     put DOC   /"〈A01〉"
               /$TEXT
               /"〈/A01〉";
〈B021〉
order          /*para=0
               /*para++;
put DOC        /"〈A02〉〈A022〉"
               /if(*para==0)
       "〈A0221〉", $TEXT, "〈/A0221〉"
     else
       "〈A0222〉", $TEXT, "〈/A0222〉"
     endif;
               /"〈/A022〉〈/A02〉";
〈B0311〉
put DOC        /"〈A03〉〈A032〉〈A0321〉"
               /$TEXT
               /"〈/A0321〉〈/A032〉〈/A03〉";
```

文档 2 向文档 1 转换时, B021 置标的内容要分成两个部分转换成文档 1 中的记录形式。转换的规则是: B021 中的第一段转换成 A0221 的内容, 第二段转换成 A0222 的内容。

文档转换信息中最重要的部分是 ELEMENT 分区, 它直接决定了文档转换的失真度大小。我们在生成中参考源文档和目标文档的一般逻辑结构来减少失真度。初始, 根据两种文档的一般逻辑结构来自动生成要素间的近似匹配关系, 如同名匹配, 接着, 用户可以自行定义这种匹配关系, 我们确定了以下的基本原则来保证要素匹配的合法性:

1. 不允许出现一对多或多对一的匹配, 即一个节点最多只能匹配一次。

2. 节点的匹配不能跨子树。如果一对节点 A 和 B 已匹配, 那么 A 的后代节点的对应节点必须在 B 的子树中, 反之亦然。

对于第一个约束条件的检验比较简单, 只需判断用户指定的两个节点是否已经匹配即可。

对于匹配是否跨子树的判断则要复杂一些。对用户指定的两个节点，需要分别向祖先和后代两个方向搜索已有匹配。但我们的算法是从根向下逐层进行的，匹配到某一个节点时其子孙一定都没有匹配，因此判断范围可以缩小到该节点的祖先，复杂度大大降低。

四、应用实例

在江苏省应用基础研究项目(面向 Internet 的电子出版的光盘文档库存储技术[5])中采用了上述的文档转换技术。把 HTML 和 SGML 文档作为数据源的原因是此类文档结构性较好且网上资源丰富。图 3 给出了从 SGML 或 HTML 文档转换到光盘文档的处理过程。

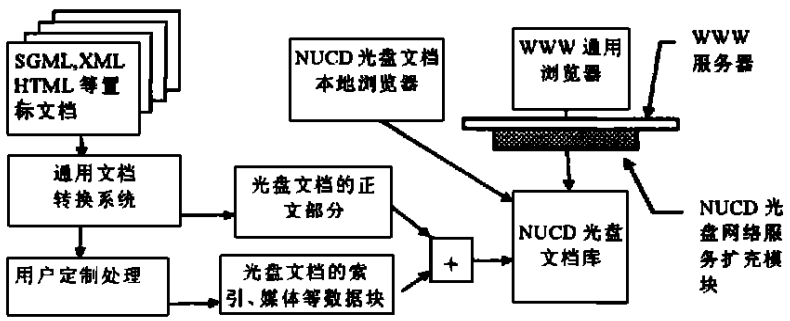

图 3　基于转换的光盘文档库的生成过程

在光盘文档格式的转换过程中，用户可以将特定的文字作为关键词抽取出来，以便生成基于关键词的检索数据，同时利用转换系统中能够嵌入用户私有处理的功能，在转换处理过程中直接生成关键词的检索数据和全文检索数据。

用户的文档转换要求用转换描述语言来记录，用户可以用它描述文档中要处理的标志以及处理方式等信息，转换描述语言由转换系统来解释、执行。由于转换信息独立于转换系统，因此用户可自由地设计各种不同的处理以适应不同的文档转换要求。比如，面向印刷处理可转换成 SPDL 描述，按照我们的光盘文档库要求将 HTML 或 SGML 文档转换为光盘文档等。

五、相关工作比较

加州大学伯克利分校研究开发的多媒体文档系统 Ensemble[7] 中，使用 Tree Transformation 来进行文档的转换。该系统中定义了自己的转换规则描述语言。转换规则由模式(Pattern)和处理动作(Action)两部分组成。模式类似 Lisp 语言中的表结构，这种形式的模式可以很方便地反映出节点的上下文特征，但是由于缺乏断言这样的手段，功能不够灵活和强大。Action 部分也是用表结构表示的，不过因为文档转换的结果是针对特定应用的固定的文档结构，所以规则中的 Action 部分较为简单。IBM 公司也在 1999 年底推出了一个可视化的 XML 转换工具 Visual XML Transformation。其主要功能是进行 XML 文档间的转换，其转换要求非常严格，灵活性较差。

本文所论述的文档转换技术在减少失真度的同时，又支持用户定制的处理模块功能，以便允许用户生成一些增值信息。除了可对置标文档进行有效处理外，通过增加抽取功能，系统也可用于平面文档如纯文本文档到置标文档的转换。

(下转 64 页)