基于数据表形式的 JDF 工作机制构建

刘 恒，刘诗德，苏 强
（信息工程大学，郑州 450052）

摘要：针对 JDF 工作机制问题，以 JDF 树形数据结构、节点继承关系和资源推动机制为出发点，提出了一种新的执行方式。首先根据 JDF 文档，抽象树形结构，定义树形结构遍历方式；然后以资源为控制点，设计节点数据表，实现 JDF 工作路径的完整描述；最后，利用二者间的内在联系，构建了基于数据表形式的 JDF 工作机制。

关键词：数据表；树；节点；JDF 工作机制
中图分类号：TS865　**文献标识码**：A　**文章编号**：1001-3563(2013)09-0111-04

Construction of JDF Working Mechanism Based on Datasheet

LIU Heng, *LIU Shi-de*, *SU Qiang*
(Information Engineering University, Zhengzhou 450052, China)

Abstract: A new way to achieve JDF workflow was put forward considering JDF tree data structure, node inheritance relationship, and resource-driving mechanism as starting points. First of all, tree structure was abstracted according to the JDF document, and the traversal method was defined. Then, node datasheet was designed relayed on JDF resources and a complete description of JDF working path was constructed. Finally, JDF working mechanism based on datasheet was built by taking advantage of intrinsic link between them.

Key words: datasheet; tree; node; JDF working mechanism

JDF 是 CIP4 为印刷数字化工作流程定义的一种基于 XML 语言的与生产设备无关且包含生产管理全过程的数据格式[1]。在现行的 JDF 工作流程中，以 XML 语言编码的 JDF 文档作为工作传票，运用节点元素设计工作模块，将产品信息和控制信息定义在节点中；以树形结构实现工作流程，依据树根叶节点实现对产品不同信息、参数和过程的描述；以资源状态控制工作机制，通过资源连接关系，决定将被执行的后续节点，其数据结构清晰、信息描述准确和数据交换方便的特性有利于实现对产品以及如何生产产品的描述和控制，也使 JDF 技术成为实现印刷数字化工作流程统一、高效、集成化的可行方式。

现有 JDF 数字化工作流程已经在国外有广泛的应用，在国内也表现出了盎然生机，印刷人急切渴望应用 JDF 技术来提高效益。针对 JDF 的资源推动机制，国内外有不少的厂商实现了 JDF 技术，然而由于核心技术保密性，对于其中的具体方法和实现过程还不甚了解，基于此，笔者提出基于数据表形式的 JDF 工作机制的构建方法，希望在 CIP4 的框架下，实现 JDF 机制构建方式的公开探索。

虽然 JDF 有固定的执行机制，但其具体的执行方式和数据组织形式却是千差万别，一样的结果有着截然不同的内在过程。首先将 JDF 节点树进行拆分，形成不同的子树，将子树定义为链表形式，根据子树下标进行多重循环，实现子树的分层遍历执行；然后将子树内容定义为数据表形式，依据指针域关键字索引和数据表间的储存地址的对应关系和资源链接，实施查找和执行，最终完成工作流程。

1　定义数据结构

1.1　定义子树序列

树形结构是一类重要的非线性数据结构，其结构是以递归方式定义，直观看来，树是以分支关系定义的层次结构，在任意一颗非空树中，有且仅有一个特定的根节点，其余结点可分为互不相交的有限子树，

收稿日期：2013-02-26
作者简介：刘恒(1988-)，男，四川遂宁人，信息工程大学硕士生，主攻数字出版技术。

即“树中有树”[2]。而在 JDF 作业文档中，上下节点之间存在逻辑继承关系，兄弟节点之间不存在制约关系，此特性与树形结构非常吻合，故将 JDF 文档的节点对应为树结点，抽象为树形结构。如将一个 JDF 文档抽象为如图 1[4]所示的树形结构，并将树命名为 T。为了建立清晰且明确的对应关系，实现 JDF 文档可执行节点的查找和执行，按照子树继承关系，遍历树结点，以从上至下，从左至右的顺序，采用阿拉伯数字作为下标，对子树进行命名。

定义 L = 树结构层数，B_i = 非根叶层中各兄弟结点数，其中 $1 \leqslant i \leqslant L-2$，$D$ = MAX（最底层节点中相同父结点的兄弟结点数），$abc\cdots x$ 为子树多重循环下标，其中 $0<a<L$，$0<b \leqslant B$，$0<c \leqslant D$，a 为最外层循环控制数，并以此类推，本例中 $L=4$，$B1=2$，$B2=2$，$D=3$，$0<a<4$，$0<b \leqslant 2$，$0<c \leqslant 3$。

根据上述定义，将图 1 所示树图生成子树序列，最终得到的树序列为 Tree = {T, T1, T2, T11, T21, T111, T112, T113, T12, T211, T212, T22, T3}。并按照上述方法对树序列进行拆分，得出各树的结构图，如 T11 树为图 2 所示，T212 树见图 3。

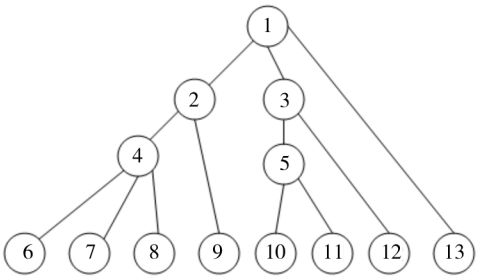

图 1 JDF 文档结构树

Fig. 1 Tree structure of JDF document

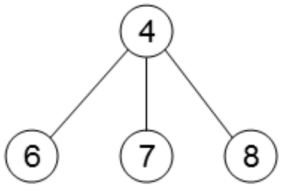

图 2 T11 子树

Fig. 2 T11 subtree

图 3 T212 叶结点

Fig. 3 T212 leaf node

1.2 定义数据结点

为了在数据结构中清晰的表达 JDF 文档的树形结构，以指针为数据元素之间的逻辑关系映像，存储位置为数据元素的物理地址，根据存储位置和指针域的对应关系，将子树序列，即 JDF 作业节点通过链表的形式连接起来[5]，得到简洁的作业结构链。

定义链表结点数据结构为{存储地址(A)，数据域{D}，指针域{P}}，其中存储地址是节点在内存中的物理位置，数据域是包含多个子数据域数组，储存对应的树形结构中所执行的节点对应的相关数据，如由 MIS 自动生成的 ID 号，指针域数组是存储直接后继的存储位置，或是指向数据表的存储地址。结点的结构见图 4。

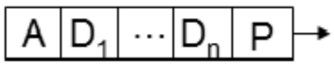

图 4 结点数据结构

Fig. 4 Node data structure

1.3 定义查找过程

定义子树查找函数，按照多重循环模式，对子树序列中子树下标进行匹配，进行遍历查找，当某重循环查找完毕后，删除该层子树，回复到更内一层循环，继续上述工作，直至到树叶结点。遍历过程 f 定义见图 5。

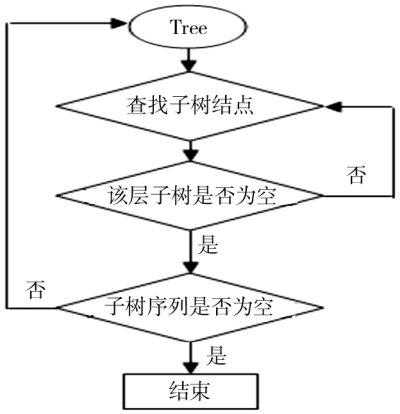

图 5 子树查找过程

Fig. 5 Searching process of a subtree

通过数据结点和子树查找模式的定义，可以仅考虑结点的前驱和后继关系，而忽略执行过程中的数据操作，即将逻辑关系和具体操作分开，同时避免从复杂的“资源机制”方面来考虑 JDF 结构，能减少数据因子，避免如“外来资源”[1]带来的复杂局面，这样结构更加清晰，逻辑更加明朗，有助于对 JDF 执行机制的理解，也便于对过程节点的控制和实施，利用上述定义对 Tree 序列查找生成的链表见图 6。

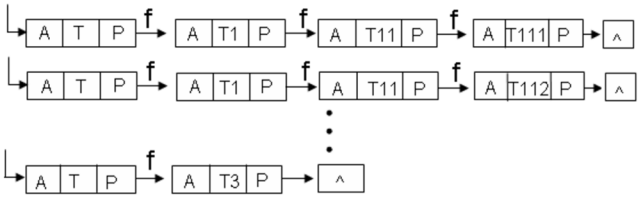

图 6　Tree 序列查找链表

Fig. 6 Searching linked list of the Tree sequence

2　定义数据表

在 JDF 印刷流程中,包含两类最基本的元素:节点和资源[1],其中,节点分为 4 类,分别是意图节点、过程组节点、组合过程节点、过程节点,资源分为输入资源和输出资源[4]。JDF 是以不同节点构成 JDF 流程的结构体系,而节点内部的执行则是以资源推动为工作机制,所以,要完整的执行整个 JDF 流程,不仅要明确定义其内部结构,还要将结构体系和具体的资源以及操作参数相连接。

运用子树序列和数据结点,根据 f 查找函数,生成查找链表,按照链表数据,进行内存分配,以保证上一结点的指针指向下一结点存储地址。以图 1 为例,生成数据表见表 1,2,3,其中 0xdFFTT#代表子树序列

表 1　JDF 文档结构表

Tab. 1 JDF document structure table

存储位置	树名称	ID	指针域
0x0dFFTT#1	T	1	0x0dFFTT#2,3
0x0dFFTT#2	T1	2	0x0dFFTT#4, 9
0x0dFFTT#3	T2	3	0x0dFFTT#5
0x0dFFTT#4	T11	4	0x0dFFTT#6,7,8
0x0dFFTT#5	T21	5	0x0dFFEE#10,11
0x0dFFTT#6	T111	6	0x0dFFEE#6
…			
0x0dFFTT#13	T3	13	0x0dFFEE#13

表 2　JDF 文档节点数据表

Tab. 2 Node datasheet of JDF document

存储位置	Type	Index	Types	指针域
0x0dFFEE#1	Intent	0	Product	0x0dFFWW#1
0x0dFFEE#2	Intent	1	Part Product	0x0dFFWW#2
0x0dFFEE#3	Intent	2	Part Product	0x0dFFWW#3
0x0dFFEE#4	ProcessGroup	11	DigitalPrinting Gathering Stitching	0x0dFFWW#4
0x0dFFEE#5	ProcessGroup	21	DigitalPrinting Gathering	0x0dFFWW#5
0x0dFFEE#6	Process	111	DigitalPringting	0x0dFFWW#6
0x0dFFEE#7	Process	112	Gathering	0x0dFFWW#7
0x0dFFEE#8	Process	113	Stitching	0x0dFFWW#8
0x0dFFEE#9	Combined	12	Cutting Folding	0x0dFFWW#9
……				

表 3　JDF 文档资源表

Tab. 3 Resources table of JDF document

储存位置	Input	Class	Status	Output	Status
0x0dFFWW#1	Part Product	…	Waiting	Product	Unavailable
0x0dFFWW#2	DigitalPrinting Gathering Stitching	Waiting	Part Product	…	
	ProcessGroup Cutting Folding	Combined	Waiting		
0x0dFFWW#3	DigitalPrinting Gathering	ProcessGroup	Waiting	Part Product	…
	Stitching Cutting Folding	Combined	Waiting		
0x0dFFWW#4	DigitalPrinting Gathering Stitching	ProcessGroup	Waiting	ComponentSheet	…
0x0dFFWW#5	DigitalPrinting Gathering	ProcessGroup	Waiting	ComponentSheet	…
0x0dFFWW#6	DigitalPrintingParams	Parameter	Available	ComponentSheet	…
	Media	Consumable	Available	…	
	RunList	Parameter	Available	…	
0x0dFFWW#7	GatheringParams	Parameter	Available	ComponentSheet	…
	Component	Quantity	Available		
0x0dFFWW#8	StitchingParams	Parameter	Available	ComponentSheet	…
	Component	Quantity	Available		
		……			

存储位置，0x0dFFEE#代表节点数据存储位置，0x0dFFWW#代表资源存储位置。

3 执行分析

首先，进行数据结构定义与设计。定义结点数据结构，并抽象 JDF 文档为节点树，并遍历子树，得出子树序列。并根据 JDF 文档内容，生成文档结构表，文档节点数据表，文档资源表。

其次，根据 3 组表之间的指针关系，查找生成数据表执行链。以图 1 为例生成的一条执行链表见图 7。

图 7 树执行链表

Fig. 7 Linked list of tree execution

然后，根据定义的子树查找过程，在节点数据表中基于 Index 的值进行循环查找，以生成“倒序”执行过程。本例中，最先执行包含 T111，T112，T113 子树的链表，然后进行查找并执行 T11，T12，再到执行 T1，T2，T3 过程，最后执行 T 过程，完成整个工作。

与此同时，在每执行完一条链表时，需要对其输出资源状态进行更改，而且，当本层结点执行完毕后，即查找到上层结点时，需要对上层节点的输入资源状态进行更改，如本例中对 0x0dFFWW#6，7，8 3 个过程执行完毕后，更改其输出资源状态为 Available，同时查找到上层节点为 T11，并将其输入资源状态更改为 Available。如此循环，直至整个过程执行完毕。

4 结论

利用数据表形式来定义和描述 JDF 工作机制是对实现该机制的具体构建方法的详细阐述和一种实现形式，运用 3 种相互衔接的数据表和子树序列依据特定的查找过程，最终实现 JDF 的工作机制的构建。该方法还存在如下优点。

1）将执行结构和执行数据分开定义，有利于结构分析，过程管控。文档结构表 1 是对 JDF 文档的整体结构的描述，文档节点数据表 2 是对节点类型的描述，文档资源表 3 是对资源推动机制的描述和控制，各表分开定义，又通过指针连接，结构明朗，思路清晰。

2）数据表结构简单，易于储存和查看，同时能修改、添加、删除数据项，便于对执行过程的实现，通过对数据项的修改，可以方便 MIS 对资源状态等的管控，同时易于实现 JMF 循环消息的控制。

3）以链表方式实现 JDF 工作机制，方式简单易行，同时对故障节点的查找和修复方便直观。

4）整个数据表存储紧凑，节省空间，只需根据指针域内容进行遍历查找，时间复杂度低并且有利于计算机程序对 JDF 工作机制的理解和实现。

5）以数据表的形式实现避免了以热文件形式的不安全性[6]，提高了执行过程的鲁棒性。

综上所述，依据数据表形式能够实现 JDF 工作机制的构建。

参考文献：

[1] 罗如柏，周世生. 基于 AOV 网的 JDF 印刷工作流程建模[J]. 北京工业大学学报，2011，37(1)：7-12.
LUO Ru-bai，ZHOU Shi-sheng. Activity on Vertices Based JDF Workflow Modeling[J]. Journal of Beijing University of Technology，2011，37(1)：7-12.

[2] 严蔚敏，吴伟民. 数据结构[M]. 北京：清华大学出版社，2008：118.
YAN Wei-min，WU Wei-min. Data Structure[M]. Beijing：Tsinghua University Press，2008：118.

[3] 周世生，罗如柏，赵金娟. 印刷数字化与 JDF 技术[M]. 北京：印刷工业出版社，2008：121.
ZHOU Shi-sheng，LUO Ru-bai，ZHAO Jin-juan. Digital Printing and JDF Technology[M]. Beijing：Printing Press，2008：121.

[4] CIP4. JDF Specification Release 1.4a[EB/OL]. http://www.cip4.org/documents/jdf_specifications/JDF1.4a.pdf.

[5] 李崇. 基于数据结构的链表创建方法探究[J]. 现代电子技术，2009(2)：117-119.
LI Chong. Exploration of Link Table Creation Based on Data Structure[J]. Modern Electronic Technique，2009(2)：117-119.

[6] 刘文峰. 基于 JDF 数字化工作流程的 JMF 研究[J]. 印刷工业，2009(2)：77-80.
LIU Wen-feng. Research on JMF Based o-n JDF Digital Workflow[J]. Printing Industry，2009(2)：77-80.