

ワークフローモデル Workflow Base に基づく 分散 DTP 支援環境の構想

国島丈生

岡山県立大学情報工学部情報通信工学科
kunishi@c.oka-pu.ac.jp

近年のインターネットの発達により、学術論文誌や国際会議への電子投稿が広く行われはじめている。ところが、これに伴い、電子投稿された論文を紙に出力する際に、出力できない、判読不能な状態で出力される、などといった出力トラブルが頻発しはじめている。この問題は、印刷関連技術のさまざまな要素が複雑に絡み合って発生する。そのため、印刷の専門家でない学術分野では、分散環境上で知識の統合を行い、それを各研究者の環境に適用して論文作成・印刷工程を支援する、など、学術分野の実情に即した独自の解決策を考える必要がある。本報告では、我々が現在研究を進めている、ワークフローとデータベースの統合モデル Workflow Base の応用の一つとして、学術分野における分散 DTP (Desktop Publishing) 支援環境の構想について報告する。

An Environment for Supporting Distributed DTP on Workflow Model *Workflow Base*

Takeo Kunishima

Faculty of Computer Science and System Engineering
Okayama Prefectural University

In recent years, many transactions or conferences tend to accept submission in electronic ways, such as by sending PostScript files by e-mails. However, many troubles have been occurred when printing out the submitted papers. As these troubles originate in various mismatches in printing technologies, intelligent systems for printing technologies in distributed environment must be needed. In this paper, we propose an environment for supporting distributed DTP (Desktop Publishing) especially in the research area, using Workflow Base, an workflow model with database technologies.

1 はじめに

現在、学術分野でもコンピュータによって論文等の原稿を作成し、PostScript プリンタによって出力するという事は当たり前のように行われている。そして、近年のインターネットの発達により、作成した PostScript 等のファイルを電子メールなどにより送付することにより、学術論文誌や国際会議への電子投稿が広く行われはじめている。ところが、これに伴い、電子投稿された論文を紙に出力する際に、出力できない、判読不能状態で出力される、などといった出力トラブルが頻発しはじめている。このようなトラブルは、 $\text{\TeX}$ などのフリーソフトウェア、Word などの商用ソフトウェアのいずれでも発生している。

この問題は、ソフトウェアの種類・バージョン、プリンタの種類、論文の漢字コードなどの要素が複雑に絡み合っている。そのため、その解決には印刷関連技術に関する高度な知識とノウハウを必要とする。しかし、関連技術の急速な進歩発展ともあいまって、計算機関連の研究を行っている研究者でさえ、このような知識・ノウハウを充分には持ち合わせていないことが多い。また、すべての研究者が文書作成環境を統一することは不可能である、などの事情があり、商業印刷業界でとられている解決方策は適用することができない。

したがって、この問題の解決のためには、分散環境上で知識の統合を行い、それを各研究者の環境に適用して論文作成・印刷工程を支援する、など、学術分野の実情に即した独自の解決策を考える必要がある。

本研究では、我々が現在研究を進めている、ワークフローとデータベースの統合モデル Workflow Base [2, 3] の応用の一つとして、このような学術分野における分散 DTP (Desktop Publishing) 支援環境の構想について報告する。

以下、2章では、商業分野および学術分野における DTP の現状と違いについて述べ、3章では、我々が研究を行っているワークフローモデル Workflow Base の概要について述べる。4章では、学術分野の分散 DTP を支援するシステムについて、必要な機能を述べた後、そのた

めのさまざまなモデリングについて提案する。5章でまとめと今後の課題について述べる。

2 学術分野における DTP

2.1 DTP の特徴

DTP はコンピュータ上で印刷物の作成工程をすべて行う手法であり、近年の出版物の大半は DTP によって作成されている。商業印刷の分野では、おおまかな工程としては、原稿(文章)の作成、図・写真データの作成(スキャニング、補正、データ変換など)、レイアウトなどを経て、高解像度出力機による版下の出力を行い、印刷工程に回される。

商業印刷における DTP でも、入稿されたファイルが高解像度出力機によって出力できない、というトラブルが発生することが多い。これを避けるため、さまざまな暗黙の約束ごとのもとに工程が行われることが多い。たとえば、次のようなものが典型例である。

- MacOS 上で作業が行われる。
- 入稿できるファイルは QuarkXpress, PageMaker, Photoshop, Illustrator などごく一部のアプリケーションで作成されたものに限られる。印刷業者によってバージョンも指定されることが多い。
- 入稿ファイルには TrueType フォントは使わない。また、(原則的には)印刷業者の持っているフォントしか使えない。

このような約束ごとによって制作者側と出力側の環境をできるだけ統一し、出力時のトラブルを回避しようとしている。

2.2 学術分野の分散 DTP の特徴

商業分野と比べ、次のような点が異なる。

- 一般に、製作に関わる人の環境がすべて異なる。プロフェッショナルの DTP の分野では、印刷側が、使用するソフトウェアとそのバージョン・OS・フォントの種類

などを規定し、それらを満たした電子原稿のみを受け付けて出力を行う。しかし、学術分野では、このように環境を統一してトラブルを減らすという解決策がとれない。また、OSに起因する差異(標準的な漢字コードや行末コード、など)、ソフトウェアのバージョンの差異などによるトラブルも起こる。

- TeX, LaTeX, tgif, dvi2ps, dvips, などのフリーウェアを使う場合が多い。
- ユーザが、PostScript などの背景技術については詳しくない場合が多い。
- 出力媒体がさまざまである。(PostScript ファイル、PDF ファイル、HTML ファイル、紙、など)
- 同一のソースを使うが出力を変えたいことがある。(紙出力はモノクロで、PDF ファイルはカラーで、など)

3 Workflow Base

紙面の都合上、ここでは概要を述べるにとどめる。詳細は参考文献 [2, 5] を参照されたい。

活動オブジェクトは仕事の単位に対応しており、 $a = (I, O, P, S)$ と定義される。ここで I は a への入力、 O は a からの出力、 P は a に実行主体であるエージェント、 S は a を実行するための活動オブジェクトの集合(ワークフローテンプレート, WFT)である。直観的には、 a が入力 I を受けとると、 P は必要な仕事を実行し、出力 O を出す。そのとき a は必要なら I を分割して S にその依頼を行ない、実行を監視し、結果を合成して出力 O を生成する。

WFT では、以下のように定義される 2 種類の活動オブジェクト間のフロー(水平フロー($\Rightarrow$)・垂直フロー($\rightarrow$))を用意する。

$$\begin{aligned} a_1 \Rightarrow a_2 &\stackrel{def}{=} O_1 \supseteq I_2 \\ a \rightarrow a_i &\stackrel{def}{=} a_i \in S \text{ ただし } a = (I, O, P, S) \end{aligned}$$

ワークフローは、すべての活動オブジェクトが水平フローまたは垂直フローの推移的閉包に

よって互いに連結されている(ワークフロー制約)ような WFT として定義できる。

WFT の実行モデルは、水平フローと垂直フローにそれぞれ対応した 2 種類のプロダクションシステム P-box と B-box とから構成される。P-box 中のルールは前向きに評価され、一方 B-box 中のルールは後ろ向きに評価される [2]。

WFT を実際の仕事に適用するためには、WFT 中の入力、出力、エージェント等を実体化しなければならない。活動オブジェクト $a = (I, O, P, S)$ に対して、 I に $I' = \{i_1, i_2, \dots, i_n\}$ 、 O に $O' = \{o_1, o_2, \dots, o_m\}$ 、 P に $P' = \{p_1, p_2, \dots, p_k\}$ を割り当てることを考える。ここで I, O の定義域 $\mathcal{M}$ 、 P の定義域 $\mathcal{P}$ はそれぞれ $\sqsubseteq_{\mathcal{M}}$ 、 $\sqsubseteq_{\mathcal{P}}$ で半順序集合であると仮定する。この割り当て $\theta = \{I/\{i_1, \dots, i_n\}, O/\{o_1, \dots, o_m\}, P/\{p_1, \dots, p_k\}\}$ が $I' \sqsubseteq_S I \Leftrightarrow \forall i \in I, \exists i' \in I'. i' \sqsubseteq i (O' \sqsubseteq_S O, P' \sqsubseteq_S P \text{ についても同様})$ という条件を満たすとき、これは型の特化に対応する。この条件は、秘書/{高田}のような、通常想定されるワークフローに対する割り当てに相当している。

この割り当てに基づいて、活動オブジェクト間および WFT 間に半順序関係 $\sqsubseteq$ が定義できる。すなわち、ワークフローの汎化・特化階層が構成できる。これらの各種情報がワークフロー用のデータベース、つまりワークフローベース (Workflow Base) に格納される。

Workflow Base におけるワークフロー演算子としては、WFT を対象とするもの、活動オブジェクトを対象とするもの、WFT 集合を対象とするものの 3 種類が考えられる [4, 5]。このうち、WFT を対象とするもの、WFT 集合を対象とするものについては関係度数に準じた体系を提案している。

4 学術分野の分散 DTP 支援

本章では、2 章の現状考察に基づき、学術分野での分散 DTP を支援するシステムの現段階

での構想について述べる。

4.1 実現したい機能

システムとして支援したい機能は、大きく分けて2つある。

1. 分散環境に起因するトラブルの原因追跡、およびトラブルの防止。
2. 新しいメディアに対する標準的工程の提示および支援。

1. は、PostScript に関連する出力トラブルなど、分散 DTP の工程の途中で発生するトラブルに関する対処支援を目的としている。具体的には、たとえば次のような機能が必要であろう。

- トラブルが起こったときの原因追跡。
- 事例収集によるトラブルの未然防止。
- システムのバージョンアップ等による影響のシミュレーション。

2. の最近の例としては、PDF ファイルを生成する工程の支援というものが考えられる。PostScript ファイルを Adobe Acrobat で処理してやれば PDF ファイルは作成できるが、その際、なるべく英文 Type1 フォントを含めるような PostScript ファイルの作り方をしたほうがよい、など、それまでに行っていた PostScript ファイルの生成方法や設定を変更したほうがよい。しかし、 $\text{\TeX}$ およびその周辺ツールの設定は必ずしも簡単ではなく、なんらかのガイドによる支援が必要なことが多いと考える。

また、関連技術の進歩がひじょうに速いため、新たに登場してくるメディアに対応する設定の更新も支援できるとよいと考えている。

以降の節では、上記のような機能を実現するための基本的な考察として、DTP の工程やその中で発生してくるメディアのモデル化手法、トラブルの起こったケースの記述方法、トラブルを回避するような工程の設計手法などについて述べる。

4.2 メディアオブジェクト

文書ファイル、そこから生成した PostScript ファイルや PDF ファイル、文書に使用する画像ファイルなど、DTP の過程で使用・生成されるデータをメディア (media) と総称することにする。本稿のシステムでは、これらメディアをオブジェクトとして扱うことにする。以降、このオブジェクトをメディアオブジェクトと呼ぶ。

メディアには、テキスト、画像、バイナリなどさまざまな種類があり、さらに

- テキスト……ブレインテキスト、 $\text{\LaTeX}$ ソースファイル、HTML ファイルなど。
- 画像…… GIF、JPEG、TIFF など。
- バイナリ……PostScript、PDF など。

というように、さらに細分化することができる。また、DTP の過程を考える場合、同じ EPS (Encapsulated PostScript) であっても、それを作成したソフトウェアによって区別して扱わなければならないことがある。たとえば、tgif 2.16PL12-JP で生成された EPS ファイルは、CID ベースの PostScript プリンタに出力するときパッチをあてなければならない、といったことがある。

そのため、メディアオブジェクトを型付けし、型階層によって管理すると都合が良い。本稿では、メディアオブジェクトの型として、MIME の media type 宣言 [1] に creator (そのメディアを作成したソフトウェア) に関する記述を加えたものを考える。

- 第一階層……MIME の media type:
text, image, audio, video, application(text, image, audio, video 以外の基本型), multipart(複合型), paper
- 第二階層……MIME の subtype:
(例) text/plain, text/html, image/jpeg, application/postscript, application/pdf, など。
- 第三階層……creator type:
(例) application/postscript/tgif2.16pl12-jp, application/pdf/acrobat3.0j, など。

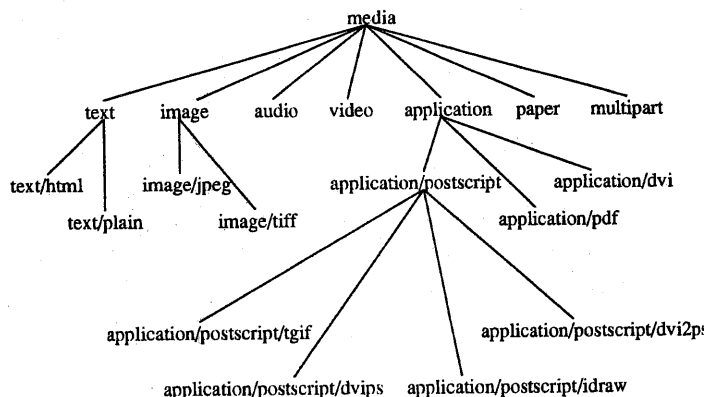


図 1: メディアオブジェクトの型階層

図 1 に型階層の概略を示す。

第一階層に複合型を許すのは、複数のメディアが揃ってはじめて意味のあるメディアとなる場合があるためである。たとえば、EPSF スタイルファイルなどによって $\text{\LaTeX}$ に EPS ファイルを取り込んで生成された dvi ファイルでは、取り込んだ EPS ファイルの相対パス名だけを保持しており、dvi ファイルだけを別のサイトに転送しても正常に出力できない。このような場合に、dvi ファイルと取り込んだ EPS ファイルを multipart によって“グループ化”し、転送したいことがある。また、paper は紙メディアオブジェクトを表す型であり、第二階層以下の型は持たない。

なお、本稿では、メディアオブジェクトに対するメソッドは特に定義しない。

4.3 活動オブジェクト

Workflow Base における活動オブジェクトは、入力オブジェクト集合、出力オブジェクト集合、エージェント、副活動オブジェクト集合から定義され、直感的には、「エージェントが入力オブジェクト集合から出力オブジェクト集合を生成する。そのとき、副活動オブジェクトに相当する作業が“子供”の作業として行われる」という意味を表す [2]。

本稿のシステムに Workflow Base を応用す

る場合、入力オブジェクトおよび出力オブジェクトがメディアオブジェクトであると考えてよい。すなわち、活動オブジェクトはメディアオブジェクトからメディアオブジェクトへの変換を表すと考えられる。

分散 DTP 環境下では、活動オブジェクトには次のような種類がある。

- 変換 (conversion)……単一または複数のメディアオブジェクトから単一メディアオブジェクトを得る。
- 構成 (composition)……複数のメディアオブジェクトから単一の複合型メディアオブジェクトを得る。
- 分解 (decomposition)……単一の複合型メディアオブジェクトから、それを構成している複数のメディアオブジェクトを得る。
- 転送 (transfer)……単一のメディアオブジェクトを異なる環境に転送する。転送するメディアオブジェクトは基本型でも複合型でもよい。また、転送の方向も自分の環境から他の環境、他の環境から自分の環境いずれでもよい。(環境については後述)

DTP プロセスを表すワークフローで用いられる活動オブジェクトは、すべて上記の活動オブジェクトのいずれかから特化によって導出される。

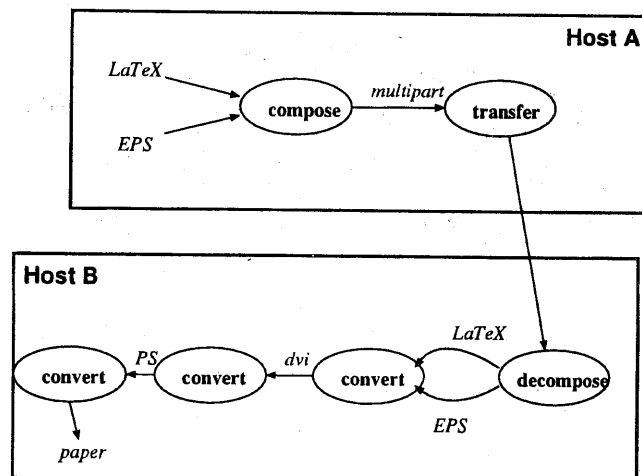


図 2: DTP プロセスのワークフローの例

図2に、 $\text{\LaTeX}$ を用いた分散DTPプロセスを表すワークフローの例を示す。この例では、ホストAで作成した $\text{\LaTeX}$ ファイルと(図のための)EPSファイルをいったん複合型にまとめてからホストBに転送し、そこで複合型をほどいてからdviへ、ついでPostScriptに変換し、ローカルプリンタに出力している。

4.4 環境

同じプログラムを用いる場合でも、そのプログラムの動作するOSや設定が異なるために、あとの工程に対して影響をおよぼすことがある。たとえば、同じapplication/postscript/dvipskという型を持つメディアオブジェクトであっても、それが作成されたOSがUNIXであるかMacOSであるかWindows95であるかによって最終出力が異なってくる場合がある。また、同じUNIXホスト上であっても、ユーザの設定が違っていれば異なるメディアオブジェクトが得られ、最終出力が異なってくる可能性がある。

このような、ホストやユーザの違いによる差異をシステムに取り入れるため、環境(environment)という概念を導入する。

まず、活動オブジェクトを、それを実行す

るホスト(サイト)を明示的に記述するよう拡張する。すなわち、活動オブジェクト $A = (I, O, P, H, T, S)$ とする。ここで、 I は入力オブジェクト集合、 O は出力オブジェクト集合、 P は(その活動を実行する)エージェント、 H は(その活動が実行される)ホスト、 T はその作業を行うのに必要なプログラム(O に紙メディアが含まれる場合はプリンタの機種も含む)、 S は A の副活動オブジェクト集合を表す。

このとき、環境 $E(P, H)$ を、共通のエージェント P およびホスト H を持つような活動オブジェクトの集合として定義する¹。すなわち、 $E(P, H)$ は、エージェント P がホスト H 上で実行する活動オブジェクト全体の集合である。同じ型を持つメディアオブジェクトであっても、それらが異なる環境下で作成されていれば、以降の工程で異なる結果を得ることがある。

UNIXのNFSなどを用いて、異なるホスト上でも全く同じように作業を行えるようにしている場合がある。このような場合、二つの環境 E_1 と E_2 は等価であるという。

また、あるメディアオブジェクト o が、ホスト H 上でユーザ P が管理しているとき、 o は環境 $E(P, H)$ の上に置かれるという。

¹引数の P, H は、特に混乱を生じない場合は省略できる。

4.5 トラブルの記述

DTP における出力トラブルは、多くの場合、環境と上で用いたプログラムの組み合わせが原因となって、それより後の工程で(ある特定の環境に限り)引き起こされることが多い。たとえば、オリジナルの `tgif-2.16pl12-jp` で作成された EPS ファイルを $\text{\LaTeX}$ ファイルに取り込むと、古い PostScript プリンタでは問題なく出力されるが、比較的新しい PostScript プリンタではうまく出力されないことがある。

そこで、トラブルを引き起こす活動オブジェクトから成る集合をトラブル・ワークフローテンプレートと呼ぶ。これは、ワークフローテンプレート的一种になっている。たとえば、上に示した例では、トラブル・ワークフローテンプレートは図 3 のようになる。トラブルが発見され次第、対応するトラブル・ワークフローテンプレートが Workflow Base に格納され、次節で述べるアルゴリズムなどに用いられる。

4.6 トラブルフリー・ワークフローを求めるアルゴリズム

前節で述べたトラブルの起こらないワークフローをトラブルフリー・ワークフローと呼ぶ。本節では、入力・出力となるメディアオブジェクト、その置かれる環境が与えられたときに、入力から出力を得るトラブルフリー・ワークフローを求めるアルゴリズムについて述べる。

DTP のワークフロー中で行われるメディアオブジェクトの変換は一般に不可逆である。たとえば、 $\text{\LaTeX}$ ファイルから dvi ファイルを生成すると、dvi ファイルから元の $\text{\LaTeX}$ ファイルを再度生成することはできない²。したがって、DTP の過程ではフィードバックは存在しないと考えてよい。また、入力オブジェクトのメディア型と出力オブジェクトのメディア型を決定すると、その変換は、メディア型の第二階層レベルでほぼ一意に定まる。たとえば、入力が $\text{\LaTeX}$ 、出力が PostScript である場合、

² `dvi2tty` を用いて plain text に変換した後 `plain2` を用いて $\text{\LaTeX}$ を生成することはできるが、生成された $\text{\LaTeX}$ ファイルは元のファイルとは異なるものになる。

$\text{\LaTeX} \rightarrow \text{dvi} \rightarrow \text{PostScript}$ という変換過程に定まる。

このことを利用すると、トラブルフリー・ワークフローは容易に求めることができる。以下にそのアルゴリズムを示す。入力となるメディアオブジェクトを o_1 (型 t_1)、出力となるメディアオブジェクトを o_2 (型 t_2)、 o_1, o_2 の置かれる環境をそれぞれ E_1, E_2 とする。

1. Workflow Base より、入力・出力の型がそれぞれ t_1, t_2 であるようなワークフローのうち、最汎なものを取り出す。これを W とする。
2. $\forall a_i \in W$ に対して、 a_i の子孫かつ E_1, E_2 に含まれるものをそれぞれ $E_1(a_i), E_2(a_i)$ とする。
3. $E_1(a_i) \cup E_2(a_i)$ から一つずつ要素を選びだし、ワークフローをつくる。環境の変更が途中で起こる場合は、適宜、構成・分解・転送を加える。生成されたワークフロー全体の集合を S とする。
4. $W \in S$ かつ、 W の部分集合としてトラブル・ワークフローテンプレートを含むものを S から取り除く。

5 おわりに

本稿で述べた支援機能は、4.1 節で述べた機能の一部しか実現できていない。トラブルの防止についても、たとえば次のような状況を反映するモデルにはなっていない。

- 同じバージョンのソフトを使っている、ローカルにパッチをあてるなどして対策を講じている、というような情報が取り込めていない。
- スタイルファイルのバージョンなど、システムであらかじめ提供されている資源の情報が取り込めていない。

新しいメディアに対する工程の支援についても、現状では十分な支援ができていない。たとえば、論文として作成した $\text{\LaTeX}$ ファイルから HTML ファイルを生成したり、印刷用

{(application/tgif, application/eps/tgif, P , H , tgif-2.16pl12-jp, $\emptyset$),
(application/postscript, paper, P' , H' , newer-PS-printer, $\emptyset$)}

図 3: トラブル・ワークフローテンプレートの例

の PostScript ファイルとオンライン公開用の PDF ファイルを同時に生成したりするようなことを考えると、4.6 節で示したアルゴリズムと同様にして工程を示すことは可能である。しかし、現実には、オンライン公開用のファイルではカラー化を行うなど、ソースに手を加えなければならないことがある。本稿では、このような作業のモデル化は検討できていない。

今後、以上のような課題をふまえ、研究を進めていきたい。問題が具体的、かつ、いま問題となっているものであるため、なるべく実装を速く進めていきたいと考えている。

謝辞

日頃より熱心にご討論いただく、岡山県立大学情報工学部横田一正教授、岡山理科大学理学部劉渤江助教授、および横田研究室の皆様に感謝します。なお、本研究の一部は文部省科学研究費重点領域研究「メディア統合および環境統合のための高機能データベースシステムの研究開発」による。

参考文献

- [1] N. Freed and N. Borenstein. *Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*, Nov. 1996. RFC2046.
- [2] T. Kunishima and K. Yokota. Flexible Workflow Frameworks for Supporting Collaborative Work. In *Proceedings of International Symposium on Cooperative Database Systems for Advance Applications*, pp. 412–419, Kyoto, Dec. 1996. World Scientific.
- [3] T. Kunishima and K. Yokota. An Agent-Based Coordination Model on Workflow Databases. In *Proceedings of 4th International Conference on Object-Oriented Information Systems*, pp. 361–371. Springer-Verlag, Nov. 1997.
- [4] 国島, 横田. ワークフローモデル Workflow Base におけるワークフロー質問言語. 情報処理学会第 55 回全国大会予稿集, 第 4 巻, pp. 177–178, Sept. 1997.
- [5] 国島. ワークフローモデル Workflow Base における演算子群. 平成 9 年度科学研究費重点領域研究「高度データベース」東京ワークショップ講演論文集, pp. 22–29, June 1997.