4P－04　　コンポーネントベース・フレームワーク技術(*C Solution* APF)

帳票編集処理 APF

山田 広佳、斉藤 悦生、吉田 和樹
(株)東芝

1 帳票編集処理 APF とは

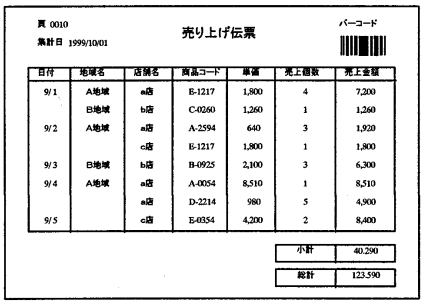
帳票編集処理 APF は、在庫管理、予約管理、受発注管理や勘定業務などにおいて用いられる帳票データの編集処理を確実、迅速に構築するための APF である。

帳票編集処理 APF では、帳票の構造を、Report、Group、Record、Field の４種類のベースコンポーネントの組み合わせで実現する。業務に固有のロジック（業務ロジック）に伴う処理は、カスタムコンポーネントとして実装し、ベースコンポーネントに差し込む。

帳票編集処理 APF は、帳票において要求される様々な処理を支援する。これらは帳票編集処理 APF によって、各機能ごとに処理が細分化される。開発者は、業務ロジックに専念しながら、その細分化された機能に基づいて業務に固有な処理を短期間で実現することができる。

2 適用による効果

ここでは、一般的に用いられる帳票の例を挙げ、帳票編集処理 APF を用いた開発の特徴を説明する。



頁 0010　　集計日 1999/10/01

売り上げ伝票

バーコード

日付	地域名	店舗名	商品コード	単価	売上個数	売上金額
9/1	A地域	a店	E-1217	1,800	4	7,200
	B地域	b店	C-0260	1,260	1	1,260
9/2	A地域	a店	A-2594	640	3	1,920
		c店	E-1217	1,800	1	1,800
9/3	B地域	b店	B-0925	2,100	3	6,300
9/4	A地域	a店	A-0054	8,510	1	8,510
		a店	D-2214	980	5	4,900
9/5		c店	E-0354	4,200	2	8,400
					小計	40,290
					総計	123,590

図１. 簡単な帳票の例

この図は、ある系列店において、特定分野での売り上げを記録する「売り上げ伝票」である。

この帳票では、１ページあたりに印刷される売り上げ件数が８件であるから、９件目からは次のページに出力されなければならない。このとき、ページごとの小計計算も必要となる。

この帳票では、日付と地域名にところどころ空欄がある。ここでは、同じデータが連続する際、２件目以降の印字を抑制している。しかし、ページが切り替われば、連続の有無にかかわらず同じデータの再印字が必要となる。

このように、簡単な帳票を考えても、その印刷対象となるデータの編集には複雑な手順を踏まなければならない。帳票編集処理 APF は、帳票のデータ編集で行われる各種の処理機能を備えるコンポーネントを提供する。

それでは、帳票編集処理 APF を使って「売り上げ伝票」の編集処理を実現する過程を見てみることにする。

この帳票の編集に必要となる処理は、以下のようなものが挙げられる：

・ページ制御(改ページ、ページ番号発番)
・集計処理(ページ小計、伝票総計)
・出力制御(同一内容出力抑制、ページブレイク)

同一内容出力抑制とは、同じ内容のデータが連続した際に２件目以降の出力抑制を行う機能であり、ページブレイクとは、改ページの際に行われる処理(例えば小計出力)を示す。

開発では、まず、帳票編集処理 APF が提供するコンポーネントを念頭に、帳票の論理構造を定義する。

Component-based Framework Technology (*C Solution* APF) : Report Editing

Hiroyoshi Yamada
Toshiba Corporation
3-22, Katamachi Fuchu-shi, Tokyo 183-8512 Japan

「売り上げ伝票」は、大きく３つの構成要素に分解される。

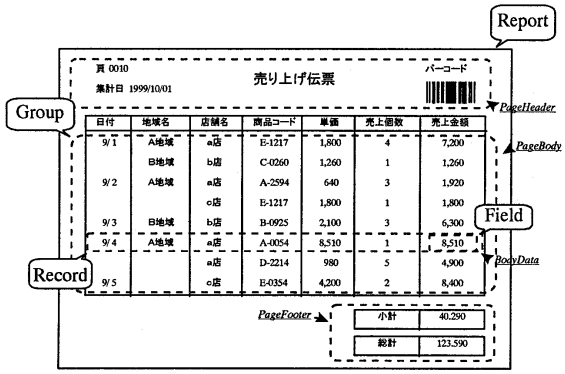


図２．帳票の論理構造

基本となるデータは、帳票の中央にリスト形式で記述されている。これは、一つの情報のまとまりであり、Group として定義する。この Group は、各行の Record よりなっており、また、この各 Record はそれぞれ個々の Field より構成されている。

また、これに付随するデータとして、上方にはページヘッダがあり、下方にはページフッタがある。これらはそれぞれ単一の Record として考えられ、その構成要素として個々のフィールドを持っている。

これを帳票編集処理 APF のコンポーネントの組み合わせで表現すると、下図のようになる。

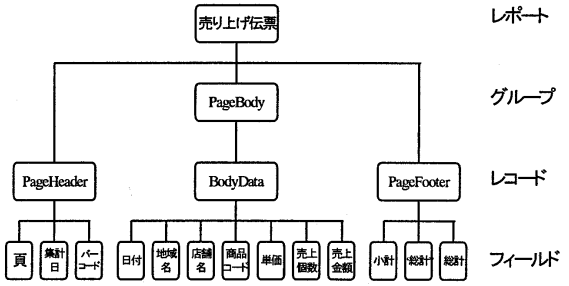


図３．帳票論理構造の木構造表現

帳票編集処理 APF では、このコンポーネントの組み合わせを XML で記述する。

次は、XML 記述での先頭部分を抜き出したものである。このようにそれぞれのコンポーネントに対応してタグを記述することによって、帳票の構造を簡潔に定義することが可能になっている。

```
<?xml version="1.0" encoding="Shift_JIS"?>
<report customize="sample.ReportCustom"
 name="UriageDenpyo"
 input="sample.InputCustom"
 line="8"
 form="Uriage">
 <record name="PageHeader">
      <field id="1"
             type="転記"
             name="頁"
             input="PageNO"
             format-type="NUMBER"
             format="###0">
      </field>
      <field id="2"
             type="転記"
             name="集計日"
             input="Date"
             format-type="DATE"
             format="yyyy/MM/dd">
      </field>
      <field id="3"
             type="転記"
             name="バーコード"
             input="BarCode"
             format-type="PLAIN">
      </field>
 </record>
```

図４．帳票論理構造の XML 表現

この XML の記述に基づき、帳票編集 APF はコンポーネントを前出の木構造に組み上げる。

編集の対象となるデータはこの木構造に投入され、該当するレコードで処理される。

ページヘッダとページフッタのレコードは、改ページの都度投入され、そこで編集作業が行われる。ページヘッダでは処理中のページ番号が格納され、ページ本体のレコードが投入される都度金額の集計が行われ、ページフッタではそこまでの集計金額が設定される。これは、上記の木構造であらわされた各ベースコンポーネントとそれに差し込まれたカスタムコンポーネントにより行う。

このように、帳票の論理構造はベースコンポーネントの組み合わせで、業務に依存する機能はカスタムコンポーネントでそれぞれ独立して実現される。従って、カスタムコンポーネントは帳票の論理的な構造に基づいて作りこむ事ができる。出力を行う帳票に変更があっても、該当する項目の変更だけを行うことにより柔軟に対応できる。

3 まとめ

この適用例の通り、帳票編集 APF を利用すると、印字位置などの物理構造やデータの処理順序とは切り離して編集処理を行えることから、低コストで、柔軟に、帳票データの編集処理を構築できる。